# Simultaneous Iterative Learning and Feedback Control Design

Anil Philip Chinnan

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2015

# ABSTRACT

## Simultaneous Iterative Learning and Feedback Control Design

Anil Philip Chinnan

Iterative learning controllers aim to produce high precision tracking in operations where the same tracking maneuver is repeated over and over again. Model-based iterative learning control laws are designed from the system Markov parameters which could be inaccurate. Chapter 2 examines several important learning control laws and develops an understanding of how and when inaccuracy in knowledge of the Markov parameters results in instability of the learning process. While an iterative learning controller can compensate for unknown repeating errors and disturbances, it is not suited to handle non-repeating, stochastic errors and disturbances, which can be more effectively handled by a feedback controller. Chapter 3 explores feedback and iterative learning combination controllers, showing how a one-time step behind disturbance estimator and one-repetition behind disturbance estimator can be incorporated together in such a combination.

Since learning control applications are finite-time by their very nature, frequency response based design techniques are not best suited for designing the feedback controller in this context. A finite-time feedback controller design approach is more appropriate given the overall aim of zero tracking error for the entire trajectory, even for shorter trajectories where the system response is still in its transient phase and has not yet reached steady state. Chapter 4 presents a combination of finite-time feedback and learning control as a natural solution for such a control objective, showing how a finite-

time feedback controller and an iterative learning controller can be simultaneously synthesized during the learning process. Finally, Chapter 5 examines different configurations where a combination of a feedback controller and an iterative learning controller can be implemented. Numerical results are used to illustrate the feedback and iterative controller designs developed in this thesis.

# Table of Contents

# LIST OF FIGURES

# LIST OF TABLES

# ACKNOWLEDGEMENTS

I would like to take this opportunity to acknowledge all of the people without whom this thesis would not be possible. You have all played an integral role in my journey and I would like to express my deepest gratitude and appreciation to each of you.

The journey to this point began over a decade ago during my undergraduate education in Electrical Engineering at Binghamton University, State University of New York, where Dr. Mark Fowler first inspired me to pursue a PhD because he saw something in me, which I had yet to see in myself. I would like to thank Dr. Fowler for his words of encouragement that inspired me to take my education as far as possible to maximize my ability to give back to society through scholarly contributions.

At Columbia, Dr. Richard Longman took me under his wing and truly introduced me to the world of academic research. Your thirst for knowledge is contagious to say the least and I thank you for the limitless motivation you have provided since that first meeting in your office, before you formally became my research advisor. Along with Dr. Longman, Dr. Minh Phan, a former student of Dr. Longman, was a key source of technical insight and valuable feedback for all aspects of my research. It is often said that an advisor has the ability to make or break your research experience. Luckily, I had two phenomenal advisors and I humbly thank both of you for all of your help.

I would also like to extend thanks to the other members of my dissertation committee, Dr. Daniel Ellis, Dr. Xiaodong Wang, and Dr. John Wright, for being involved in the completion of my dissertation and participating in my defense. In particular, Dr. Ellis deserves special thanks for being my Electrical Engineering advisor and helping to facilitate the entire process.

An exhaustive list of family and friends who have believed in me and supported me through this phase of my life could easily fill the pages of a book. First, I would like to

thank my grandparents, two of whom I know are smiling down on me from above, for giving me the greatest family. Next, thank you to all of my aunts and uncles for all of the times you have treated me more like a son than a nephew. This especially applies to Thampy Chachan, Lissy Ammachi, and Chinnamma Aunty who have helped my parents, and in turn my sister and I, in ways that words simply cannot do justice. To all of my cousins and their children, you have each helped me in your own ways and I wish I could acknowledge you all individually. There are three cousins that have been more like brothers and sisters to me, Shabu Chetai, Joe Chetai, and Mini Chechi, and I cannot emphasize how grateful I am to have you in my life. In terms of friends, I would like to thank my buddies from Elmont who I grew up with as well as my friends from St. Mary's who I have grown so close to over the years, for everything you have done for me.

Most important of all, I must thank God above, my two living Gods, my mother, Leela Chinnan and my father, Philip Chinnan, my sister and partner in crime, Ashley, my nephew, David and niece, Sofia, and my fiancé and better half, Tina Mathew. Mom, thank you for showing me how to love. You are the most innocent and loving person I know. Dad, thank you for teaching me how to be a man and not be afraid to strive for perfection. You are the most intelligent and honest person I know. Ashley Chechi, thank you for being the definition of what an older sibling is supposed to be. You have always been the wind beneath my wings and so much of my success in life I owe to you. David and Sofia, thank you for unknowingly demonstrating the fundamental goal of my research and helping me communicate it during my defense. Last but not least, my muthé Tina, thank you for being you. I honestly do not know how I could have done this without you and just hope I was able to be there for you during your PhD pursuit in the same way you were there for me. I love you all and will always keep you deep in my heart.

# DEDICATION

*This work is dedicated to my three lifelong best friends, my mother, Leela Chinnan, my father, Philip Chinnan, and my sister, Ashley. There is no me without you and I hope that I have made each of you proud.*

# PREFACE

All of the work presented in this thesis is based on original, independent research conducted by the author A. Chinnan, in collaboration with R.W. Longman of Columbia University and M.Q. Phan of Dartmouth University. The majority of this thesis is unpublished.

A version of Chapter 2 has been published [A. Chinnan and R.W. Longman, "Designing Iterative Learning Controllers from Limited Pulse Response Data," *Advances in the Astronautical Sciences*, Vol. 130, 2008, pp. 283-302]. The lead investigator was A. Chinnan, who was responsible for all areas of analysis, development, numerical simulation, and manuscript preparation. Collaboration with R.W. Longman involved the early stages of concept formation and contributions to manuscript edits.

A version of Chapter 3 has been published [A. Chinnan, M.Q. Phan, and R.W. Longman, "Feedback and Iterative Learning Control with Disturbance Estimators," *Advances in the Astronautical Sciences*, Vol. 152, 2014, pp. 747-766]. The lead investigator was A. Chinnan with responsibility for a majority of the analysis and development, in addition to all areas of numerical simulation and manuscript preparation. The early stages of concept formation and development were collaborative primarily with M.Q. Phan. Both M.Q. Phan and R.W. Longman made contributions to manuscript edits.

All other chapters are unpublished with A. Chinnan being the lead investigator in each and having responsibility for all areas of analysis, development, numerical simulation, and thesis preparation. Collaboration with M.Q. Phan involved the early stages of concept formation and development, in addition to thesis edits. Collaboration with R.W. Longman primarily involved supervisory support, technical feedback, and guidance.

# CHAPTER ONE: Introduction

When designing and implementing a control scheme that aims to optimally combine iterative learning control (ILC) and feedback control (FBC) to meet specific control objectives, the designer can approach the problem in two different ways. One approach would be to develop the ILC and FBC independently, using methodologies established and commonly used in the respective fields, and then form an optimal combination of the separately designed controllers for implementation. The second approach is to simultaneously design a feedback and learning combination controller that can be tuned for any designer specified optimal behavior. The primary advantage of the simultaneous design approach is that it naturally facilitates division of the control burden and allows the designer to explicitly prioritize the control emphasis and objectives as desired. The ideal division of control burden is generally problem specific and will vary from one learning control application to the next. So, there are significant potential gains which can be realized through the ability to optimally tune a combination controller to operate principally as a feedback controller, or principally as a learning controller, or as a truly complimentary scheme involving both types of control actions.

## 1.1: Historical Background

The academic origins of ILC can be traced to the late 1970s when Masaru Uchiyama introduced the concept on high-speed motion control of a robotic arm to follow a desired trajectory through iterative trials.[1] Though this seminal academic work spurred some industrial interest, dissemination of the learning control concept was initially limited because the publication was only available in Japanese. The concept of ILC began to

truly flourish and receive broad interest by the mid-1980s with further development motivated primarily by robotics applications.[2-4] These contributions from Suguru Arimoto, Giuseppe Casalino, Giorgio Bartolini, and John Craig paved the way for ILC to become a very active research area with numerous publications. Over the years several books,[5-8] special issues,[9-11] and surveys[12-19] have been produced, in addition to a significant number of journal and conference contributions annually.

The learning control concept was initially referred to in a myriad of ways. These include iterative control, iterative learning control, repetitive control, betterment process, virtual reference, or simply as training. It should be noted that some of these terms are commonly used to represent other types of controllers and they can mean something different entirely. For example, repetitive control (RC) shares a lot of the same characteristics as ILC, but focuses on a slightly different problem where there is no resetting between operations, as in the case of ILC. The term virtual reference refers to the modified reference trajectory that results when a learning signal is used to update the actual reference trajectory to reduce the output tracking error.[20] The virtual reference methodology is intuitive and logical because it tries to solve the inverse problem of finding the optimal input signal, to produce an output approaching the desired output, by modifying the reference signal. This control approach is considered in this thesis through a specific type of controller configuration in Chapter 5, where the learning signal is used to supplement the reference signal, effectively creating a virtual reference.

Though the development of ILC originated in robotics where repetitive motion applications are abundant, situations where ILC can be applied are widespread. ILC is ideally suited for any repetitive control process which starts from the same initial conditions. It offers the potential for improved controller performance through iterative

learning. Learning control can bring the tracking error down to the repeatability of the hardware, which is the true limiting factor in terms of tracking error performance.[21] In manufacturing and assembly applications that require a manipulator to perform some task repeatedly with high accuracy and speed, ILC can be leveraged to maximize the throughput. For satellites and spacecraft that perform high precision, iterative scanning maneuvers, ILC can be implemented to mitigate flexibility effects so that improved accuracy at higher rates can be achieved.

As industrial interest and applications for ILC began to expand, so did research trying to combine or use results from other related fields to advance the theoretical developments of ILC. One such related field is adaptive control and there are many theoretical works found in the literature that utilize adaptive control concepts and apply them to learning control.[16],[22-28] Owens and Munde provided an adaptive approach for ILC that included error feedback into an adaptive control law to effectively utilize the most recent error data.[23] The use of the most recent error data, as opposed the data from previous trials for example, makes sense particularly in reactive control implementations which incorporate feedback, since the current performance is most accurately reflected in the current data. Error feedback can also be used to stabilize an unstable plant that would otherwise be inoperable or dangerous to work with. Another branch of control theory that researchers have tried to combine or use results from to gain further insight into ILC is robust control.[29-33] This area of controls is known to involve approaches such as $\mathcal{H}_\infty$ to address things like stochastic affects, repeating and non-repeating disturbance rejection, and other similar issues. Error feedback has also

been used specifically to improve overall robustness to non-repeating disturbances in learning control applications.[34]

In addition to adaptive control and robust control, another related control field which has had significant impact on the development of ILC is optimal control.[35-38] In fact, optimal learning control is widely considered to be one of the main ILC research areas. In 1995, Amann *et al.* focused on developing a controller that tries to incorporate feedback into the learning control framework by using optimal feedback and feedforward actions.[36] Three years later, the same authors proposed a predictive optimal ILC, which minimized a cost function that weights incremental inputs and errors according to the iteration in which they occurred.[37] This norm-optimal controller, as it is often called, fundamentally is trying to determine the optimal way to incorporate error feedback into the controller implementation in a recursive fashion. The steepest gradient method was used by Choi and Jang to optimize performance of iterative learning control in feedback systems using an performance index function.[39] There has also been research which tries to merge both adaptive control and optimal control approaches in the context of learning control.[40] The design of a feedback and iterative learning combination controller that exhibits a certain optimized learning behavior was also studied previously.[41] That contribution makes use of independently developed ILC and FBC to implement a combination controller which attempts to achieve optimal performance in a loose sense, where optimality is attributed primarily to ILC performance only. These publications show that there has been notable interest since the early years of ILC to develop a control scheme that optimally utilizes both feedback and learning control actions in the controller implementation.

Along with these references, several more contributions have been made closely studying the striking similarities and clear distinctions between feedback and learning control from different perspectives. There is a collection of work that considers the equivalence between feedback control and ILC under certain conditions.[42-47] In theory, one can argue that ILC operates in the repetition-domain in an analogous fashion as traditional feedback operates in the time-domain. However there are some critical shortcomings to suggesting a true equivalence and it is generally accepted that ILC algorithms have significant practical value which traditional feedback control does not offer. Learning control that is either based on or assisted by feedback control has been studied extensively for various applications.[48-52] Among these is research into the use of feedback to improve robustness and transient behavior in the presence of model uncertainty for ILC based on an inverse process model.[48] Such applications effectively use low-pass filtering, by suppressing higher frequencies when the model uncertainty becomes large, to robustly achieve monotonic convergence of the learning process. Also included are interesting applications in the areas of bioengineering,[49] to control human limbs, and process control,[50] for chemical reactors. Frequency domain analysis of the stability and convergence of learning control that is assisted by feedback is also available.[52] While the use of frequency domain analysis is routine in FBC, there are important limitations associated with analysis of ILC in the frequency domain.

For time domain analysis, considerable work investigating continuous time ILC, with specific incorporation of feedback,[53] and discrete time ILC, with improved convergence also through the use of feedback,[54-55] is available. The analogous frequency domain analysis is traditionally done using the Laplace transform for continuous time systems or

the z-transform for discrete time systems. Numerous contributions involving such frequency domain analysis of continuous time and discrete time ILC systems exist throughout the literature.[56-59] The limitations of such analysis on ILC are inevitably encountered when the practical aspects of ILC are discussed. For continuous time systems analyzed using Laplace transforms, results on performance, convergence, and stabilization, through incorporation of error feedback, in the frequency domain have been established.[57] The practical applicability of such results are limited, however, by the simple fact that iterative learning requires past experience to be stored as data. This means that the continuous time data must first be sampled, so the results using such an analysis can only be expected to estimate real implementations. For discrete time systems, the strict use of the z-transform requires that signals be defined over an infinite time horizon. This places another limit on applicability since all practical ILC applications are finite duration, so the results of analysis based on the z-transform are similarly considered approximations.[58] Despite this limitation, some authors use z-transforms assuming notional signals of infinite length and then discuss the inferences that could be made from the theoretical results about practical, finite duration ILC systems.[56],[58-59]

Additional studies comparing different ILC update rules in the frequency domain can also be found in the literature.[60-61] The first study assumes the ILC is applied to a system that already has a FBC solution implemented, so it is effectively utilized in combination with the existing feedback.[60] The second contribution similarly performs a comparison of different ILC update rules, but considers control strategies that utilize learning based on the prior iteration, the current iteration, and a combination of both.[61] This research concluded that the control strategy which combines learning based on

both the previous and current iterations was the best approach among those considered. It also found that the overall robustness of the ILC can be notably improved through the proper incorporation of feedback. While these types of research efforts have helped to establish a strong knowledge base related to feedback and learning control, there is still a lack of understanding of how to simultaneously design a truly optimal combination controller incorporating both feedback and learning control. This is especially true for practical applications of finite duration, where the ILC system does not have sufficient time during a cycle to settle into steady state and frequency response techniques are not suitable. It is the primary objective of this thesis to establish an end-to-end simultaneous design approach which is sufficiently flexible for use in all practical ILC applications.

## 1.2: Description of Thesis Content

The majority of ILC laws investigated in this thesis make use of a Toeplitz matrix of Markov parameters which are unique to the system being controlled.[62] This use of a Toeplitz system matrix implicitly assumes that the system is linear, time-invariant (LTI), however the use of linear, time-varying (LTV) systems can also be easily accommodated in a similar matrix based framework.[63] In a digital control application involving either a LTI or LTV system, depending on the sampling rate and time duration of the desired trajectory, the size of the system matrix can get quite large. This implies that there is a requirement for a rather large number of Markov parameters, i.e. a long unit pulse response history. In practice, one cannot simply measure the unit pulse response directly at the sample times and obtain a long accurate history of the response. Methods such as Observer Kalman Filter Identification (OKID) can be used to reliably extract the desired number of Markov parameters using a long and rich excitation input history.[64] Nevertheless, there will always be limits on the number of parameters one can obtain

with a desired accuracy. ILC is unusual in control theory because it asks for zero error at every time step and zero error for all frequency components up to Nyquist. In most practical systems, such a demand is likely to cause instability and one must use extra care to avoid this. In Chapter 2, a practical approach to generate convergent ILC laws with accurate knowledge of only a limited number of Markov parameters is developed by relaxing some of the more aggressive learning control objectives, such as achieving zero error at every time step.

While the goal of achieving zero error at every time step is difficult, if not impossible, for many practical situations, this goal becomes even more ambitious in the presence of general disturbances. Disturbances should be expected during each repetition, in most or all applications, from natural imbalances and imperfections that commonly persist in all types of system hardware. It is known that ILC can automatically compensate for unknown but repeating disturbances. Though ILC can potentially excel in applications involving such a disturbance environment, it suffers from a strict limitation on performance due to its inability to generate real-time corrective control actions to compensate for non-repeating errors and disturbances. Use of FBC, or current cycle feedback as it is known in the ILC literature, can be specifically considered for handling non-repeating disturbances and becomes necessary when real-time error corrections are desired. FBC can be designed, with increased complexity, to handle both repeating and non-repeating stochastic disturbances in time. However, it lacks the ability to learn and improve performance through previous interactions with its operating environment and must be strictly causal. Therefore, FBC alone is typically not considered to have the potential to approach system hardware repeatability limits.

Earlier work developed a unified ILC formulation and investigated approaches to produce learning control laws that aim to incorporate both FBC (in the form of current cycle feedback) and ILC (in the form of previous cycle feedback).[65] For many practical systems, however, earlier works restricted the structure of the FBC gain matrices $R$ and $G$ to be of lower sub-triangular structure to ensure causality. That restriction is relaxed in Chapter 3 to allow the FBC gain matrices to be of lower triangular structure and still maintain causality, which can be particularly useful, especially in applications involving a well-conditioned plant. In addition, Chapter 3 provides further improvement of such a feedback and learning combination controller through the incorporation of disturbance estimators. Although ILC can handle repeated unknown disturbances in principle, the incorporation of disturbance estimation turns out to be especially advantageous. Both the feedback and learning performance can be improved with such disturbance estimators. In general, disturbance handling of both repeating and non-repeating disturbance is desired, with various works having investigated one or both of these within the context of ILC.[66-68]

As stated earlier, while ILC can compensate for repeating disturbances, FBC is necessary to address stochastic or non-repeating disturbances. Conventional feedback controllers, such as proportional (P), proportional-integral (PI), proportional-derivative (PD), and proportional-integral-derivative (PID), are commonly designed using frequency response based techniques. However, since practical ILC applications are finite-time by their very nature, such steady state frequency response based design approaches are not best suited for designing the feedback controller. The design of finite-time FBC was investigated, for the purposes of combining it with ILC, in a prior study with the goal of achieving satisfactory performance in both the transient and steady states.[69] While

strict causality must be followed for FBC, learning control can be non-causal since ILC is driven by previous cycle feedback. In Chapter 4, a novel approach that enables the design of a finite-time feedback and learning combination controller that can be tuned for a desired optimal behavior is presented. The approach is cost function-based and provides the designer full freedom to determine how the control burden is shared between FBC and ILC.

For practical applications in which ILC can be considered, Chapter 4 shows that the incorporation of FBC will be advantageous in the majority of implementations. The benefits of complementing ILC with FBC include additional tracking and disturbance rejection capabilities that ILC lacks. While previous work presents approaches to combine FBC and ILC in some optimal way, there is no attempt to thoroughly investigate the various configurations in an exhaustive fashion. Such a comprehensive assessment was conducted for RC, where the effects of placing the repetitive controller inside or outside the feedback loop was explored.[70-72] A detailed evaluation of different configuration options for feedback and learning control combinations is presented in Chapter 5. This evaluation directly results in the development of some novel designs for the controller implementation and evaluation techniques for the purposes of performance prediction and comparison. The approaches are cost function-based but differ with regards to the configuration used and control variable that is penalized in the cost.

## 1.3: Mathematical Formulation of Iterative Learning Control

The general learning control problem formulation utilized throughout this thesis is formally introduced here and will be referred to in subsequent chapters as necessary. The majority of the formulation will be applicable directly in each chapter. Modifications to the formulation will be primarily related to the expected disturbance and shall be

specified within the individual chapters where appropriate. If no modifications are specified, then the setup introduced here is directly applicable. To develop the general problem formulation, begin with the discrete time representation of an *n*-th order state space system model expressed as

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + w(k) \\ y(k) &= Cx(k) + Du(k) + m(k) \end{aligned} \tag{1.1}$$

where $A \in \mathbb{R}^{n\times n}$, $B \in \mathbb{R}^{n\times r}$, $C \in \mathbb{R}^{q\times n}$, and $D \in \mathbb{R}^{q\times r}$ are matrices describing the appropriate system dynamics. There are two distinct types of plant models which result from Eq. (1.1), one with direct feedthrough involving a nonzero $D$ and one without direct feedthrough, where $D$ is zero. Of these two cases, the case where $D = 0$ is the more common because the input typically does not influence the output instantly when applied in real systems. The index $k$ denotes the time $t = kT$, where $T$ is the reciprocal of the sampling frequency $f_s$, and is often referred to as the time step. If the model represented by Eq. (1.1) is generated by discretizing a continuous-time plant fed by a zero-order hold (ZOH), then it is known that the discretization process normally introduces a one-step delay from input to output. The models considered for this work are limited to systems with a one-step input-output delay. From a practical perspective, digital controllers nominally require a one-step delay from input to output to compute and apply the desired control actions. The vectors $x(k)$, $u(k)$, and $y(k)$ denote the system state, input, and output, respectively. In addition, it is assumed that the process disturbance $w(k)$ and measurement disturbance $m(k)$ are generally allowed to vary from iteration to iteration in an unknown, non-repeating fashion.

While the representation given in Eq. (1.1) is good for time-domain consideration, the general learning control problem formulation must also facilitate repetition-domain

analysis and design. Performance in both of these two fundamentally distinct domains must be thoroughly evaluated to guarantee overall stability and desirable transient behavior. In ILC there are two types of transient effects which persist in the output. One is the well-known time-domain transient or natural response commonly seen in all applications when a system is forced from equilibrium. The other is unique to the repetition-domain and the description proposed by Phan and Longman in 1988 is ideally suited to evaluate this effect [73]

$$\begin{bmatrix} y_j(1) \\ y_j(2) \\ y_j(3) \\ \vdots \\ y_j(p) \end{bmatrix} = \begin{bmatrix} CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \ddots & \vdots \\ CA^2B & CAB & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & CB & 0 \\ CA^{p-1}B & \cdots & CA^2B & CAB & CB \end{bmatrix} \begin{bmatrix} u_j(0) \\ u_j(1) \\ u_j(2) \\ \vdots \\ u_j(p-1) \end{bmatrix} + \begin{bmatrix} \Delta_j(1) \\ \Delta_j(2) \\ \Delta_j(3) \\ \vdots \\ \Delta_j(p) \end{bmatrix} \tag{1.2}$$

$$\underline{y_j} = P\underline{u}_j + \underline{\Delta}_j \tag{1.3}$$

This is often referred to as a *lifted system* representation because the input and output vectors, $\underline{y_j}$ and $\underline{u_j}$ respectively, are lifted into column vectors related to one another through $P$, a $p \times p$ matrix map representing the lifted system. Subscript *j* denotes the run or iteration number and an underbar is used to denote a column vector. In this thesis, it will generally be assumed that the product *CB* is not zero. This is a natural assumption if the system comes from feeding a differential equation with an input coming through a ZOH for discretization. If the product is zero, simple modifications can be made to the above mathematics. The result in Eq. (1.2) is obtained by assuming the standard convolution sum solution to the general state space problem presented in Eq. (1.1) above. This repetition-domain formulation is particularly useful in learning control applications and is routinely found in the ILC literature.[74-80] It facilitates analysis of the system by transforming a problem with two independent variables (time step $k$ and

repetition number $j$) into a problem with a single repetition variable $j$, as shown above. In this manner, learning control research contributions that utilize the lifted system are similar to those based on 2-D system theory.[81-85] Notice the lifted system representation is formulated around a static plant model $P$ whose matrix dimensions are solely dependent on the number of times steps, here $P \in \mathbb{R}^{p \times p}$, in the desired trajectory. The general disturbance vector $\underline{\Delta}_j$ in Eq. (1.3) can be thought of as being influenced by two different elements, a repeating disturbance $\underline{d}$ and a stochastic disturbance $\underline{d}_j$ which changes each repetition

$$\underline{\Delta}_j = \underline{d} + \underline{d}_j \tag{1.4}$$

The vector $\underline{d}$ represents the total effects of the initial conditions and repeating portions of both the process disturbance $w(k)$ and measurement disturbance $m(k)$, which can, theoretically, be combined and modeled as a single unknown repeating disturbance at the output. The vector $\underline{d}_j$ represents the total effects of the random portions of both $w(k)$ and $m(k)$, which can similarly be lumped together and modeled as a cumulative unknown stochastic, non-repeating disturbance.

The repeating disturbance portion of $\underline{\Delta}_j$ can be eliminated from the formulation by introducing a backwards difference operator $\delta_j$ that takes the difference of its operand, $\delta_j \underline{\Delta} = \underline{\Delta}_j - \underline{\Delta}_{j-1}$, in repetition. So it is not difficult to see $\delta_j \underline{\Delta} = \delta_j \underline{d}$ and consequently

$$\delta_j \underline{y} = P \delta_j \underline{u} + \delta_j \underline{d} \tag{1.5}$$

Now, with the tracking error vector defined as $\underline{e}_j = \underline{y}^* - \underline{y}_j$, where $\underline{y}^*$ is the desired output or reference trajectory, it should be clear that

$$\delta_j \underline{e} = -\delta_j \underline{y} \tag{1.6}$$

$$\delta_j \underline{e} = -P\delta_j \underline{u} - \delta_j \underline{d} \tag{1.7}$$

Using Eqs. (1.5) - (1.7), it is possible to look at the output and tracking error as state space models in the repetition-domain as follows

$$\begin{aligned} \underline{z}_{j+1} &= I\underline{z}_j + P\delta_{j+1}\underline{u} \\ \underline{y}_j &= I\underline{z}_j + \delta_j \underline{d} \end{aligned} \tag{1.8}$$

From inspection of this state space formulation, it is clear that the controllability of the system is determined entirely by the static plant model $P$ and the system is completely observable in the repetition-domain. Given the relationship between output and error in Eq. (1.6) it is easy to show how a similar model can be realized for the tracking error and yield the same conclusions regarding controllability as drawn for the output.

The limitations on performance in learning control applications, however, are set by both the static plant model $P$ and the overall randomness from repetition to repetition of the observed stochastic disturbances. If the plant model is full rank and square and there are no stochastic disturbances present, then zero tracking error is theoretically possible in learning control applications. It should be noted that the condition number of the plant model also sets limitations on performance, since ill-conditioned plants are generally known to be difficult to control and zero tracking error may not be achievable. In implementations where the plant model is not full rank, error free control is not possible even if there are no disturbances present or if only repeating disturbances persist. Similarly, in the presence of stochastic disturbances, the performance will be limited by the magnitude of the randomness from iteration to iteration, even if the plant model is full rank and square. Is such a situation, particularly when the stochastic disturbances vary in repetition to the extent that learning from previous cycles is difficult or practically

impossible, substantial improvement can be achieved through the incorporation of feedback control, as will be demonstrated in this thesis.

# CHAPTER TWO: Practical Approach to Designing Basic Iterative Learning Controllers

As stated in Chapter 1, the majority of ILC laws investigated in this thesis make use of a Toeplitz matrix of Markov parameters representing the unit pulse response unique to the system of interest. Multiplying this matrix times the input produces the convolution sum solution for each time step of the output. The size of this matrix is determined by the number of sample times in the duration of the desired trajectory, and can therefore be very large. This implies that one can easily need a large number of Markov parameters to fill the matrix. In practice, one cannot simply directly measure the unit pulse response at the sample times and obtain a long accurate history of the response, because there is not sufficient energy in an initial pulse of unit height and one time step duration. Instead, one normally inputs a signal of long duration that is rich in frequency content for all frequencies up to Nyquist. From the resulting input-output data one can compute the unit pulse response history using time-domain methods such as the OKID algorithm,[64] or one uses standard frequency response methods that take the inverse discrete Fourier transform (IDFT) of the transfer function computed as the ratio of the discrete Fourier transforms (DFT) of the output to that of the input.[86] As expected, due to measurement noise and sensor limitations, there will always be limits on the number of parameters one can obtain with a desired accuracy. It is the purpose of this chapter to investigate the implications of this in the performance of practical ILC systems and purpose a novel approach to generate convergent basic ILC laws with accurate knowledge of only a

limited number of Markov parameters. The research contributions of this chapter have already been published and can be found in the literature.[87]

One might assume that for time steps for which the parameters have become small, one can simply substitute zero for the small but inaccurate parameter values. One expects that at some point this should become true, but ILC is unusual in control theory in that it asks for zero error at every time step, and this suggests caution in making assumptions. As an example, thinking in terms of frequency response, the zero error requirement every step can be restated as asking for zero error for all frequency components visible in the number of steps in the trajectory, all the way up to Nyquist frequency. It should be noted that frequency response concepts are not generally applicable to the finite-time ILC problem. Neglecting this fact, asking for zero error at all frequency components from zero to Nyquist is an unusual requirement in control theory. In classical control one often determines stability by examining the phase margin when the magnitude response goes through zero decibels. In so far as frequency response techniques apply to ILC, one realizes that this approach is not of much help since there are repeated zero dB crossings for the fundamental and all harmonics up to Nyquist frequency.

This chapter is organized as follows. First, the general mathematical formulation of the ILC problem introduced in Chapter 1 is appropriately modified for use here. Asking for zero tracking error every time step will very often result in what can be thought of as an internal instability.[88-89] How to avoid this problem is then presented following where one asks for zero tracking error every other time step.[90] Four general classes of effective ILC laws are then presented, including the Euclidean norm contraction mapping

law, partial isometry law, quadratic cost ILC, and a pseudoinverse control law.[90-95] These ILC laws are presented in the form needed to avoid the internal instability and are used throughout this chapter when evaluating stability and performance. Then the ability to obtain accurate unit pulse response histories is examined both by OKID and by IDFT of the ratio of DFT's of output divided by input. This provides a feel for the limitations on the number of Markov parameters that can be accurately obtained. The numerical studies that follow examine the use of a limited number of Markov parameters in generating an ILC law in three different ways:

I. First, a fixed number of time steps in the desired trajectory, nominally requiring a fixed number of Markov parameters, is considered. The later Markov parameters are then replaced by zero to study the minimum number of parameters necessary to maintain asymptotic stability of the learning process, and how few parameters are needed to maintain monotonic decay of the Euclidean norm of the error while converging to zero error. In order to have a simple way to refer to this, this process will be called *truncation*.

II. A variant of truncation is also studied. A window used in RC to improve stabilization when the number of Markov parameters that could be used in the real time computation of the repetitive control law is limited is incorporated here.[96] Of course, RC is studying stability in the time-domain while ILC is in the repetition-domain, but ILC and RC are very similar in many respects. Here the number of Markov parameters used is again truncated, but then the parameters are passed through the window before using them in the ILC law.

III. The third type of investigation works in the other direction, expanding the number of time steps in the ILC law. Each control law makes use of a Toeplitz matrix of

Markov parameters having the property that every entry on the diagonal is the same, every entry on the sub-diagonal is the same, and so on. The Euclidean norm ILC law preserves this property in the ILC gain matrix, but the other ILC laws considered do not. Nevertheless, the central rows or columns of the learning gain matrix come close to satisfying this property, while the property is violated more strongly as one gets close to the edges of the matrix. Suppose that a certain number of accurately known Markov parameters have been obtained, and from this the ILC gain matrix for an ILC problem having the corresponding number of time steps has been generated. Next, assume a useful ILC law for a larger number of time steps is desired. To do this it is natural to expand the ILC gain matrix, stretching it by simply repeating the central column elements along their respective diagonal, and supplying zeros in the upper right and lower left corners of the matrix where no information is available. Such an expansion of each of the ILC laws is studied to determine what is needed to have the ILC process remain asymptotically stable in repetitions no matter how much expansion is performed. Similarly, what is needed to preserve good learning transients, in the sense that monotonic decay of the Euclidean norm of the error with iterations, is also studied. In order to have an easy way to refer to this type of design approach, it will be called *expansion*.

Note the distinction between truncation and expansion. In truncation, the Toeplitz matrix of Markov parameters is assumed to start with the dimension dictated by the number of time steps needed in the problem, and zeros are introduced in this matrix for unknown or poorly known Markov parameters. Then this modified Markov parameter matrix of full size is used in computing the ILC gain matrix. In expansion, the Markov

parameter matrix is of a smaller size dictated by the number of Markov parameters one considers to be known and accurate, presumably less than the number of time steps in the ILC problem of interest. The ILC gain matrix is computed from this smaller size Markov parameter matrix, and then this matrix is stretched to get to the needed size. In order to meet the needed dimension of the ILC gain matrix, zero entries are introduced in the gain matrix at this last stage of the process for expansion, whereas the zeros are introduced in the Toeplitz matrix in the first stage of the process in truncation. These are two competing approaches to reaching the desired dimension.

## 2.1: Problem Setup

The general learning control formulation presented in Eqs. (1.1) – (1.8) is modified slightly here for use in this chapter. For simplicity, the process disturbance $w(k)$ and measurement disturbance $m(k)$ are assumed to be deterministic. This implies that the same disturbance will be experienced at every repetition, so

$$\underline{\Delta}_j = \underline{d} \tag{2.1}$$

The implication of using Eq. (2.1) to model the disturbance environment, as opposed to that of Eq. (1.4), means that basic learning control alone is sufficient to address the disturbance rejection needs for this problem setup. This can be shown by first modifying the output given in Eq. (1.3) to reflect the fact that only a repeating disturbance is present here

$$\underline{y}_j = P\underline{u}_j + \underline{d} \tag{2.2}$$

Now, the difference operator can be applied to Eq. (2.2) in the same manner as done in Chapter 1, to obtain the following result

$$\delta_{j+1}\underline{y} = -\delta_{j+1}\underline{e} = P\delta_{j+1}\underline{u} \tag{2.3}$$

Note that the entries in the lifted system matrix map $P$, introduced in Eq. (1.2), are the Markov parameters corresponding to the unit pulse response of the system. The pulse response of a system is the reaction to a unity input at time step zero, with zero input thereafter and zero initial conditions. It can be seen from Eq. (2.3) that proper application of a basic ILC law is sufficient to address the disturbance rejection needs for the disturbance environment modeled here. A general linear ILC law has the form

$$\underline{u}_{j+1} = \underline{u}_j + L\underline{e}_j \tag{2.4}$$

or equivalently, $\delta_{j+1}\underline{u} = L\underline{e}_j$, where $L \in \mathbb{R}^{p\times p}$ and is designed using appropriately specified learning gains. Combining these equations produces the difference equation for the error history as a function of iteration number

$$\underline{e}_{j+1} = (I - PL)\underline{e}_j \tag{2.5}$$

From Eq. (2.5) it can be concluded that ILC using learning gain matrix $L$ will converge to zero tracking error for every time step of the desired trajectory, for all possible initial error histories, if and only if all eigenvalues of the coefficient matrix in this equation have magnitude less than unity

$$\left|\lambda_i(I - PL)\right| < 1 \ \ \forall i \tag{2.6}$$

This necessary condition for convergence can be restated simply as the spectral radius of matrix $I - PL$ is required to be less than one. It is possible that a convergent process have bad error transients during the learning process even if Eq. (2.6) is satisfied. The more restrictive condition is that the maximum singular value of this matrix be less than unity

$$\left|\sigma_i(I - PL)\right| < 1 \ \forall i \tag{2.7}$$

To ensure monotonic decay of the Euclidean norm of the error history vector, decaying by at least a factor given by this maximum singular value each iteration, Eq. (2.7) must be satisfied. Hence, a designer often wants to satisfy this condition in particular whenever possible.

## 2.2: Addressing the Internal Instability Issue

Now, suppose that the designer has an initial run with input $\underline{u}_1$ and corresponding output $\underline{y}_1$. If the designer wants to have zero error in the next iteration, they can specify the desired change $\delta_2 \underline{y}$ on the left of Eq. (2.3) and then try to solve for the required change $\delta_2 \underline{u} = P^{-1} \delta_2 \underline{y}$. The resulting input history is unique and any ILC law that converges to zero tracking error must converge to the same input history. However, the $P$ matrix can be ill-conditioned. As established by previously mentioned research, there is one particularly small singular value of $P$ for every zero outside the unit circle in the discrete time transfer function of the system.[90] When a differential equation is fed by a ZOH and the output sampled synchronously, there is an equivalent difference equation that has the identical input-output relationship. Generically, this difference equation has a *z*-transfer function with one less zero than pole, with the discretization process introducing the required number of additional zeros always on the negative real axis of the *z*-plane. If the differential equation had a pole excess of three or more, and the sample time is fast enough, then it has been shown that will be one or more zeros outside the unit circle.[97] Hence inverting the system produces an unstable difference equation. Corresponding to this, there is one very small singular value of $P$ for every zero outside the unit circle. One cannot precisely call this instability because the ILC

problem is finite-time and hence the time domain signal for the input does not go to infinity. However, the control action can grow each time step, oscillating from positive to negative and back as the steps progress. At every sample time the error is zero, but the error between sample times for the differential equation fed by the ZOH, is also growing every time step and oscillating in sign. Hence, although the ILC can produce zero error at the sample times, the solution is not practical.

One way to address this problem is by modifying the statement of the ILC problem. Instead of asking for zero error every time step, one asks for zero error every other time step at the even numbered steps, but still allows the input to change each step. Then one deletes the odd numbered rows of $P$ to produce the deleted $P$ matrix $P_D$

$$P_D = \begin{bmatrix} CAB & CB & 0 & 0 & 0 & 0 & \cdots & 0 \\ CA^3B & CA^2B & CAB & CB & 0 & 0 & \cdots & 0 \\ CA^5B & CA^4B & CA^3B & CA^2B & CAB & CB & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{P-1}B & CA^{P-2}B & CA^{P-3}B & CA^{P-4}B & CA^{P-5}B & CA^{P-6}B & \cdots & CB \end{bmatrix} \tag{2.8}$$

Then the learning law is updated to $\underline{u}_{j+1} = \underline{u}_j + L\underline{e}_{Dj}$ where $\underline{e}_{Dj}$ is the error history vector containing only the errors at even numbered times steps, and $L \in \mathbb{R}^{p \times p/2}$ now, assuming that $p$ is even. There error propagation in Eq. (2.5) becomes $\underline{e}_{Dj+1} = (I - P_D L)\underline{e}_{Dj}$, and equations Eqs. (2.6) and (2.7) are modified by substituting $P_D$ for $P$. The next section presents the four types of ILC laws considered in this chapter, which incorporate the row deletions developed here.

## 2.3: Iterative Learning Control Laws

***Contraction Mapping:*** The Euclidean norm contraction mapping learning law is

$$\underline{u}_{j+1} = \underline{u}_j + sP_D^T \underline{e}_{Dj} \tag{2.9}$$

where *s* is a scalar gain. Let the singular value decomposition of $P_D$ be $U\Sigma V^T$ where $\Sigma^T = [S \quad 0]$ and *S* is the diagonal matrix of $p/2$ singular values, while $U \in \mathbb{R}^{p/2 \times p/2}$ and $V \in \mathbb{R}^{p \times p}$ are both unitary matrices. Then

$$\underline{e}_{Dj+1} = (I - sP_D P_D^T)\underline{e}_{Dj} \quad ; \quad (U^T \underline{e}_{Dj+1}) = (I - sS^2)(U^T \underline{e}_{Dj}) \tag{2.10}$$

Note that matrix *U* does not change the error history Euclidean norm, so the learning law converges to zero error every other time step provided one picks *s* so that $1 - s\sigma_i^2$ is smaller than one in magnitude for all singular values $\sigma_i$. Matrix $P_D$ is automatically full rank for a system with time delay through the system of one time step, and can be modified appropriately otherwise. The slowest part of the system to learn is associated with the smallest singular value. The error at the intermediate steps as well as the error between steps no longer grows exponentially when using the deleted $P$ matrix.[90]

***Partial Isometry:*** The learning rate of ILC law given in Eq. (2.9) can be improved at high frequencies by replacing $P_D^T$ by its singular valued decomposition with all singular values replaced by unity:

$$\underline{u}_{j+1} = \underline{u}_j + sVU^T \underline{e}_{Dj} \tag{2.11}$$

The $S^2$ in Eq. (2.10) is then replaced by *S*, so that the small singular values are no longer squared and the factor $1 - s\sigma_i$ attenuates the error faster.

***Quadratic Cost ILC:*** A quadratic cost function can be used to compute the change in the control action for iteration $j+1$

$$J_{j+1} = \underline{e}_{Dj+1}^T Q \underline{e}_{Dj+1} + \delta_{j+1} \underline{u}^T R \delta_{j+1} \underline{u} \tag{2.12}$$

The symmetric weight matrices $Q$ and $R$ allow one to directly influence the transients during the learning iterations, and are positive definite. Using Eq. (2.3), it is possible to minimize this cost to obtain the update law

$$\underline{u}_{j+1} = \underline{u}_j + (P_D^T Q P_D + R)^+ P_D^T Q \underline{e}_{Dj} \tag{2.13}$$

The superscript + indicates the Moore-Penrose pseudoinverse. By proper choice of the structure of the weight matrices, one can again prevent any updates of the components of the control action on the space spanned by the null space of $P_D$ and therefore prevent any instability in the control action with time step.

***The Pseudoinverse ILC Law:*** Convert Eq. (2.3) to deleted form $\delta_{j+1}\underline{e}_D = -P_D \delta_{j+1}\underline{u}$, set the desired error in the next iteration to be zero, and use the pseudoinverse solution to get the control update that has minimum Euclidean norm of the change in the control. The result is the control law

$$\underline{u}_{j+1} = \underline{u}_j + sP_D^+ \underline{e}_{Dj} = \underline{u}_j + sV \begin{bmatrix} S^{-1} \\ 0 \end{bmatrix} U^T \underline{e}_{Dj} \tag{2.14}$$

where an overall gain *s* has been inserted allowing one to adjust the size of the control adjustment made each iteration.

## 2.4: Obtaining Pulse Response Data

There are various approaches to obtaining the Markov parameters needed to fill the $P_D$ matrix. One way is to identify a state space difference equation model from data, and then use the $A$, $B$, $C$ matrices to compute the parameters. Doing this involves making choices of model order, decisions about what modes are noise modes and what modes are system modes, etc. Various methods for identifying the $A$, $B$, $C$ matrices such as the Eigensystem Realization Algorithm or the Ho-Kalman algorithm are designed to start

from pulse response data, and one can simply use the direct computation of the pulse response from data avoiding the system identification. Since there is very little energy in a unit pulse for one time step, it is not practical to directly measure the pulse response and have a substantial response history before the output disappears below the noise level. The traditional method of finding the pulse response, is to input a long rich signal, take the DFT of the input and the DFT of the output, take the ratio that represents the transfer function, and then take the inverse transform. The time domain alternative is the OKID algorithm. Both methods are investigated in this work for comparison purposes. The aim is simply to get some general understanding of how noise influences the computed parameters, so there is no real focused effort to optimize the parameters involved in the problem. Now, consider a third order continuous time system represented by the following transfer function

$$Y(s) = G(s)U(s) = \left(\frac{\alpha}{s+\alpha}\right)\left(\frac{\omega_0}{s^2 + 2\zeta\omega_0 s + \omega_0}\right)U(s) \tag{2.15}$$

where $a = 8.8$, $\omega_o = 37$, and $\zeta = 0.5$. The equivalent discrete time system representation, analogous to Eq. (1.1), can be found by assuming the input comes through a ZOH. For the purpose of numerical simulation, a 100 Hz sample rate is used in this investigation. The deterministic input is chosen as white zero mean Gaussian noise of unit variance with $2^{14}$ time steps and the initial condition is taken as zero.

Figure 2-1 gives the Markov parameters versus time step using our perfect knowledge of the model matrices. Then simulated noisy data were generated with various noise levels, and the DFT approach and the OKID approach were used. Note that the DC gain of the above system is unity, so it is reasonable to consider that the input and the output have the same units. Plant noise was considered to be additive to

the input function. The standard deviation of the input function was specified in picking the deterministic input numbers. So, it is possible to generate additional random numbers of the needed standard deviation to produce the desired signal to noise ratio for the process noise. Here, standard deviation of the deterministic signal is divided by that of the random noise distribution. Note that the presence of process noise ensures a noise floor that will limit the number of Markov parameters that can be accurately identified. To set the measurement noise level, the chosen deterministic input function was applied to the system, and the root mean square (RMS) of the resulting output function was computed and used as the standard deviation of the output signal. Then random noise was added to produce any desired signal to noise ratio for measurement noise. Ten thousand Monte Carlo runs were made and used to compute the standard deviation of the identified Markov parameters. Figures 2-2 and 2-3 give the relative error of the computed Markov parameters versus time step using the DFT method, and Figures 2-4 and 2-5 give corresponding results using OKID. The number of observer Markov parameters is set to 20 and the row dimension of the Hankel matrix is set to 40. The first few Markov parameters using OKID are not plotted since the algorithm, as implemented, allows a direct feed through term which made the uncertainty in these few terms large. The relative error is computed as the computed standard deviation of the identified Markov parameter divided by the actual value of the Markov parameter. Figures 2 and 3 show results using a DFT and IDFT with all $2^{14}$ points. In all cases, the relative error grows as the Markov parameters decay. Using DFT, the worst result has a standard deviation of the identified Markov parameter that is 40 times the size of the Markov parameter itself. The OKID results are significantly better. It seems clear that even with some optimization of the Markov parameter identification process, the relative

error will eventually become large. It is also true that if the system is asymptotically stable, the Markov parameters eventually get small, and one might expect that their values could be replaced by zeros and still have a functioning ILC law.

## 2.5: Truncating and Expanding ILC Laws

***Truncation:*** Consider a problem in which the number of Markov parameters one has with confidence is less than the number of Markov parameters needed to fill the $P_D$ matrix, i.e. the number of time steps in the desired trajectory. A logical approach is to simply enter a value of zero for any unknown parameters. With this modified $P_D$ matrix, it is possible to substitute it into any of the four ILC laws, Eqs. (2.9), (2.11), (2.13), or (2.14), presented earlier. As demonstrated through prior research, one can send the Markov parameters through an accelerated exponential window to improve the identification process.[96] If it is desired to keep $M$ Markov parameters, the $k^{\text{th}}$ parameter is multiplied by $\exp[(-k)/(M-k+1)]$ in order to decrease the influence of any step discontinuity from the last Markov parameter used to the zeros that follow. Note that a possible disadvantage of doing this over the expanding approach below, is that this approach could demand manipulating large matrices to compute the laws given in Eqs. (2.11), (2.13), and (2.14).

***Expansion:*** Note that if the gains in $L$ correspond to a linear time invariant rule, then $L$ will have the Toeplitz property that each element within any given diagonal has the same value. This becomes a bit more complicated since odd numbered rows are deleted here, so for the Euclidean norm law the concept of a diagonal takes on the stair case form as follows

$$\text{a)}\ \begin{bmatrix} d_1 & - & - \\ d_2 & - & - \\ - & d_1 & - \\ - & d_2 & - \\ - & - & d_1 \\ - & - & d_2 \end{bmatrix} \qquad \text{(b)}\ \begin{bmatrix} c_1 & c_7 & c_{13} \\ c_2 & c_8 & c_{14} \\ c_3 & c_9 & c_{15} \\ c_4 & c_{10} & c_{16} \\ c_5 & c_{11} & c_{17} \\ c_2 & c_{12} & c_{18} \end{bmatrix} \qquad \text{(c)}\ \begin{bmatrix} c_1 & c_7 & 0 & 0 & 0 \\ c_2 & c_8 & 0 & 0 & 0 \\ c_3 & c_9 & c_7 & 0 & 0 \\ c_4 & c_{10} & c_8 & 0 & 0 \\ c_5 & c_{11} & c_9 & c_7 & c_{13} \\ c_6 & c_{12} & c_{10} & c_8 & c_{14} \\ 0 & 0 & c_{11} & c_9 & c_{15} \\ 0 & 0 & c_{12} & c_{10} & c_{16} \\ 0 & 0 & 0 & c_{11} & c_{17} \\ 0 & 0 & 0 & c_{12} & c_{18} \end{bmatrix} \tag{2.16}$$

Notice that Eq. (2.16a) has the aforementioned stair case form. Given the Euclidean norm gain matrix, it is possible for example, to expand it by two rows and one column. This is done by introducing two initial zeros in the new column followed by repeating the original last column shifted downward by two rows. Repeat this to expand further. It may be simpler to just imagine the full original $P^T$ matrix, introduce zeros for the appropriate Markov parameters in the upper right hand corner, and then delete every other column.

The other three ILC laws manipulate the $P_D$ matrix to form the gain matrix, for example by using singular valued decomposition, and this does not preserve the Toeplitz property of all elements being the same on any given diagonal. Nevertheless, the central part of the learning gain matrix still has this property, and only near the edges of the matrix do the entries on a given diagonal start to deviate. Starting by considering a $P \in \mathbb{R}^{100\times 100}$ for the system given in Eq. (2.15) sampled at 100 Hz sample rate, and then deleting the odd numbered rows, Figure 2-6 plots the $d_1$ and $d_2$ diagonals as in Eq. (2.16a) for the partial isometry law, Figure 2-7 does the same for the quadratic cost law, and Figure 2-8 for the pseudoinverse law. Given the $L$ matrix for one of these laws computed from a given number of Markov parameters, one then expands the matrix to apply to a larger number of time steps by repeating the entries in the central part of the

matrix. In Eq. (2.16b), consider the first and last columns as part of the edges as in Figures 2-6 to 2-8 for each of the step diagonals, and the middle column as corresponding to the middle part of these plots. Then to stretch the matrix $L$ by four rows and two columns, convert to the matrix in Eq. (2.16c) where the middle column is repeated two times. Note that it may desirable to want to do a combination of truncation and expansion, by using a larger initial $P$ matrix than the number of parameters available allows, but not so large that one has difficulty performing the singular value decompositions or other computations needed.

Table 2-1 presents results from a similar investigation. It considers the pseudoinverse ILC law, but similar results apply to partial isometry and quadratic cost. Starting again with a $P \in \mathbb{R}^{100\times100}$ for Eq. (2.15), the odd rows were deleted, and the $\mathbb{R}^{100\times50}$ gain matrix generated. The first entry on the table uses only the entries in $L$ that correspond to $d_1$ and $d_2$ in Eq. (2.16a) and sets all of the remaining gains in $L$ equal to zero. Note that these values that correspond to $d_1$ and $d_2$ are not identical for all entries on the diagonal for this law. It also gives the maximum eigenvalue magnitude from the left hand side of Eq. (2.6) and the maximum singular value from the left hand side of Eq. (2.7). The second row includes the first entry to the right of each $d_1$ and the first entry to the left of each $d_1$. The third row in addition uses the entries to the right and the left of each $d_2$. One observation is that once a relatively small number of diagonals have been included, the learning gain matrix is able to stabilize the system and produce monotonic decay. This fact is the basis for the ability to do expansion as discussed here. It should be noted that when the sample rate is increased, the number of necessary diagonals seemed to increase substantially faster for the Euclidean norm and the quadratic cost

laws, than for the partial isometry and the pseudoinverse laws. A second observation is that when doing the expanding, it is natural to expand by multiples of 4 diagonals added to the initial 2 diagonals. The numerical investigation of expanding presented below expands according to this rule.

Figures 2-9 and 2-10 study the minimum number of Markov parameters needed to create a Euclidean norm ILC law that can be expanded indefinitely, while maintaining stability with respect to Eq. (2.6) or maintaining monotonic decay as described by Eq. (2.7), for Eq. (2.15) with a 100 Hz sample rate. The value of the $a$ in Eq. (2.15) was varied from 3 to 8.8 and 13.4, resulting in the three values of bandwidth for the 3 dB down point as indicated. Increasing the bandwidth decreases the number of parameters needed. Increasing the sample rate seems to make a linear increase in the number of Markov parameters needed for both stability and monotonic decay. There are actually 6 plots in each figure and the three corresponding to applying the window discussed above all lie approximately on top of each other. The use of the window is seen to very substantially reduce the number of Markov parameters need to start the expansion. The same was true when the windowed Markov parameters were used in computing the quadratic cost ILC. However, neither the partial isometry nor the pseudoinverse laws were helped by using the windowed parameters. The same trends were observed for a second order system when the real part of the continuous time roots were kept at -8.8 and the imaginary parts of the roots were moved from zero to 8.8 and 17.6. On the other hand, for a first order system the number of parameters needed was two, independent of the bandwidth of the system and the sample rate.

## 2.6: Numerical Studies of Truncating and Expanding ILC Laws

Numerical simulations are performed to examine how well the process of expanding ILC laws works, and how well truncating works. We will say that an ILC law obtained for the number of Markov parameters equal to the dimension of $L$ is expandable, if both the stability and the monotonic decay conditions, inequalities Eqs. (2.6) and (2.7) are satisfied when $L$ is expanded to any higher dimension. The third order system of Eq. (2.15) is used. Figures 2-11 and 2-12 give results for truncation, and will be discussed in more detail below. Figures 2-13 and 2-14 treat expansion and consider 50 Hz and 100 Hz sample rates for the zero order hold input to Eq. (2.15), while using noiseless values for the Markov parameters. There can be 8 curves on a plot and one needs some guidance to be able to interpret the curves. Subplots a, b, and c in Figure 2-13 are three different views of the same plots, done to different amplifications. There are 4 ILC laws with the plot key for each below the figure title. For the pseudoinverse there is one curve in Figure 2-13a that starts with dimension 14 and a second curve starting with dimension 18, where the dimension in each case is that of the original $L$ matrix, corresponding to the number of Markov parameters used. The dimension on the horizontal axis is then the dimension of $L$ after expansion. Note that the solid line starting at 14 maintains the maximum singular value less than unity for a while as the matrix dimension is expanded but then goes above one. The solid line starting at 18 does not go above one, and corresponds to the first dimension for which monotonic decay is maintained when expanding the learning gain matrix to any desired final dimension. The corresponding two curves for partial isometry start with 2 and 4 Markov parameters, for the Euclidean norm 6 and 10, and the quadratic cost 6 and 10. In order to be able to distinguish the different curves, Figure 2-13b zooms in so that one can see that the partial isometry law

is less than unity when using 6 Markov parameters, but not for 2. Then Figure 2-13c zooms in further to establish when these ILC laws are expandable. Figures 2-13d, e, and f are analogous but look at the spectral radius, i.e. stability.

Figure 2-14 is the same as Figure 2-13 except that the sample rate is changed to 100 Hz. This time the last number of Markov parameters before expandability is 26, 6, 34, and 30, and the first number of Markov parameters that produce an expandable control law is 30, 10, 38, and 34, for the pseudoinverse, partial isometry, Euclidean norm, and quadratic cost ILC laws respectively. Note that the partial isometry law needs much fewer Markov parameters to make an expandable ILC gain matrix than the other laws.

Figures 2-15 and 2-16 are analogous using the same initial dimensions for expansion, but the Markov parameters obtained using OKID were employed, with plant noise having a signal to noise ratio of 20. As might be expected, the pseudoinverse ILC law is the most adversely affected by introduction of noise, and partial isometry seems to be least affected. Figure 2-17 goes back to noiseless Markov parameters and investigates 1000 Hz sample rate. This time only the pseudoinverse and partial isometry laws are plotted, because the Euclidean norm and quadratic cost required so many more Markov parameters in order to be expandable. The largest number of Markov parameters before expandability is 50 and 34, and the first number that is expandable is 62 and 38, for the pseudoinverse and the partial isometry laws respectively.

Now return to the truncation method of generating the ILC laws as treated in Figures 2-11 and 2-12. For comparison with expanding as detailed in Figure 2-14, these plots use noiseless Markov parameters and a 100 Hz sample rate. They also start with the same number of Markov parameters as used in Figure 2-14, one of which was just

below the value needed for expandability, and the other just above. The top curves in Figures 2-11 and 2-12 correspond to the pseudoinverse. There are 6 other curves on the plot which are not distinguishable and all are very near one throughout. By zooming in we find that these other laws go above one on both the stability and the monotonic decay plots except for the partial isometry law, which happens to be identical whether one uses the truncation method or the expansion method. Hence, the truncation method requires more Markov parameters in order to work. In particular, the number of parameters needed for the pseudoinverse increased from 30 to 46, for the partial isometry increased from 10 to 38, and the quadratic cost went from 34 to 38. Of course, the Euclidean norm stayed at 38.

Numerical simulations of the learning process for each ILC law were made. The command trajectory was created using the following polynomial

$$y^*(t) = \pi\left(5\left(\frac{t}{t_p/2}\right)^3 - 7.5\left(\frac{t}{t_p/2}\right)^4 + 3\left(\frac{t}{t_p/2}\right)^5\right) \tag{2.17}$$

This is used, then reflected, and then repeated to form the curve in Figure 2-18 as desired. The polynomial has the characteristic that it has zero velocity and acceleration at the start and end. Figures 2-19 through 2-26 present the absolute value of the errors as a function of time. The ILC law aims to achieve zero error at every even time step, and such time steps form the bottom of the black areas. No attempt is made to get zero error at the odd numbered time steps, and the error for these steps forms the top of the black areas. Figures 2-19 through 2-22 use noiseless Markov parameters, while Figures 2-23 through 2-26 use noisy Markov parameters from OKID with measurement noise exhibiting a signal to noise ratio of approximately 20. The degradation due to noise is

evident in the partial isometry and the pseudoinverse results. In the latter case the zoomed in plot in Figure 2-27 shows the influence of noise.

## 2.7: Summary of Findings

ILC laws are usually based on the Toeplitz matrix of Markov parameters. One can directly compute the Markov parameters from rich input-output data, and this can be desirable in that it does not rely on identifying a model of the system with the associated decisions of what are noise modes and what are system modes. However, plant and measurement noise in the data used will limit how many Markov parameters can be accurately identified. In addition, in some ILC applications the size of the learning gain matrix needed can be sufficiently large that one has numerical difficulties in performing the needed computations, such as a singular value decomposition. In either case, one may be able to generate the learning gain matrix of the size for a certain number of time steps in the trajectory, but need the matrix for a larger number of steps. The expandability concept used here shows how it is possible to do such an expansion, preserving both stability and the desired monotonic decay property, provided one has enough Markov parameters to start with. In particular, the partial isometry ILC law is seen to only need a rather small number of parameters before the learning gain matrix can be expanded with stability and monotonic decay by the method presented here to any arbitrary larger size.

**Table 2-1. Effects of Truncation on Pseudoinverse Control Matrix**

| Diagonals | Singular Values | Eigenvalues | Singular Values | Eigenvalues |
|---|---|---|---|---|
| 2 | 38.5398 | 1.7544 | 38.5398 | 1.7544 |
| 4 | 19.5287 | 0.7892 | | |
| 6 | 13.3777 | 3.5496 | 13.3777 | 3.5496 |
| 8 | 11.5596 | 1.6708 | | |
| 10 | 3.6286 | 2.4304 | 3.6286 | 2.4304 |
| 12 | 4.5853 | 1.0997 | | |
| 14 | 1.3222 | 1.1008 | 1.3222 | 1.1008 |
| 16 | 1.7658 | 0.3662 | | |
| 18 | 0.5069 | 0.3619 | 0.5069 | 0.3619 |

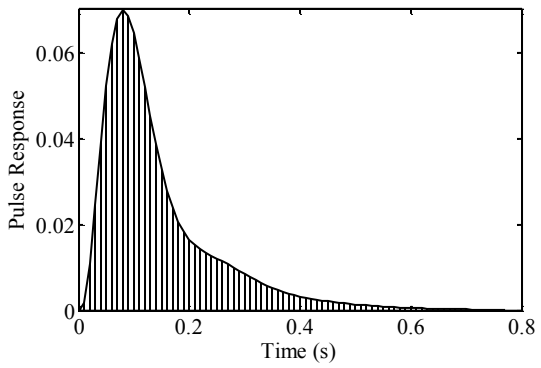



**Figure 2-1. Pulse Response of 3rd Order Plant**

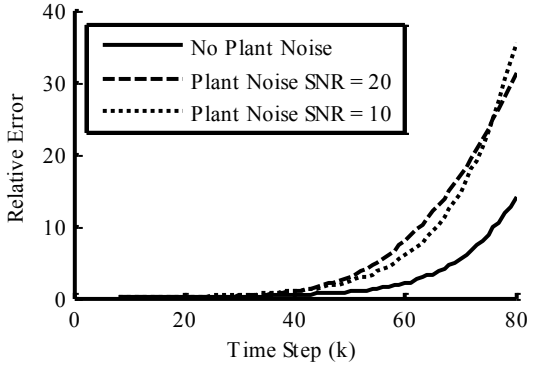



**Figure 2-2. DFT: Measurement Noise SNR = 20**

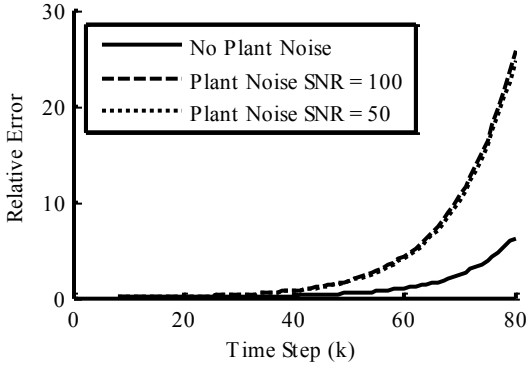



**Figure 2-3. DFT: Measurement Noise SNR = 100**

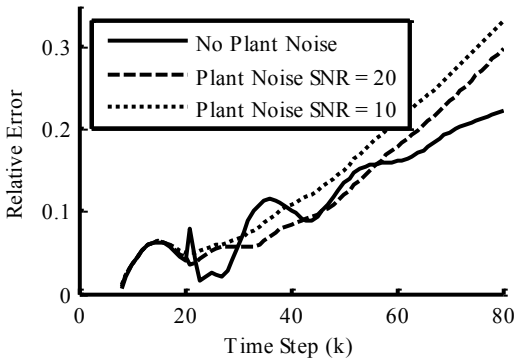



**Figure 2-4. OKID: Measurement Noise SNR = 20**

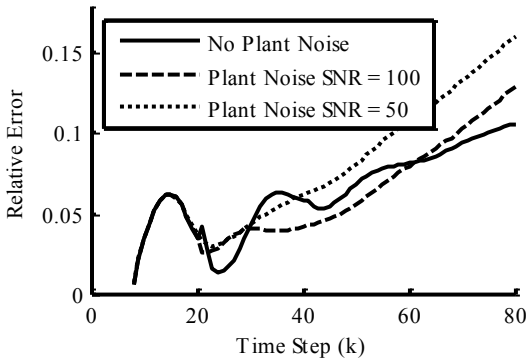



**Figure 2-5. OKID: Measurement Noise SNR = 100**

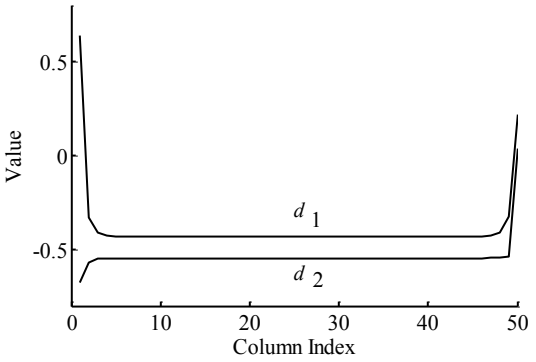



**Figure 2-6. Diagonals for Partial Isometry Law**

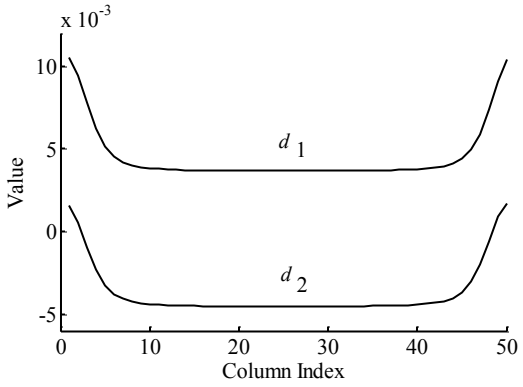



**Figure 2-7. Diagonals for Quadratic Cost Law**

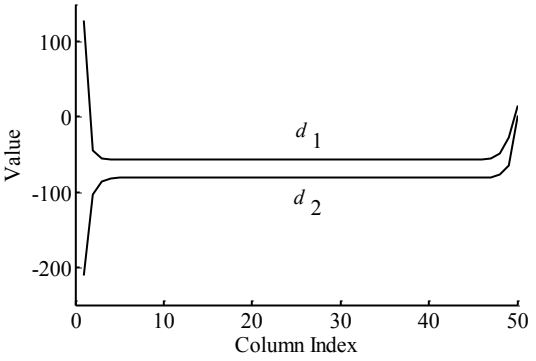



**Figure 2-8. Diagonals for Pseudoinverse Law**

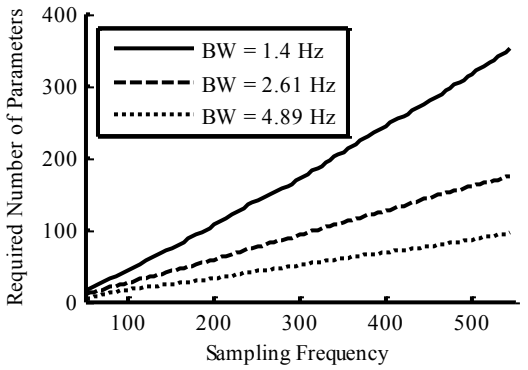



**Figure 2-9. Reqs for $\sigma$ < 1 of 2nd Order Plant**

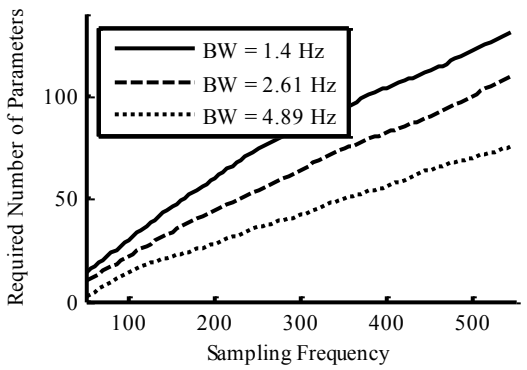



**Figure 2-10. Reqs for $|\lambda|$ < 1 of 2nd Order Plant**

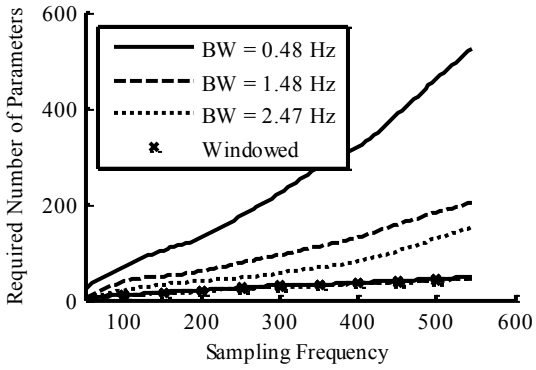



**Figure 2-11. Reqs for $\sigma$ < 1 of 3rd Order Plant**

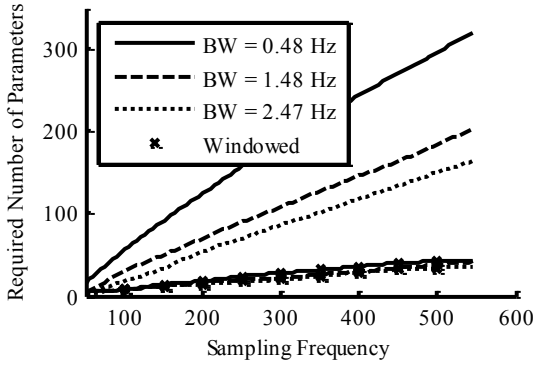
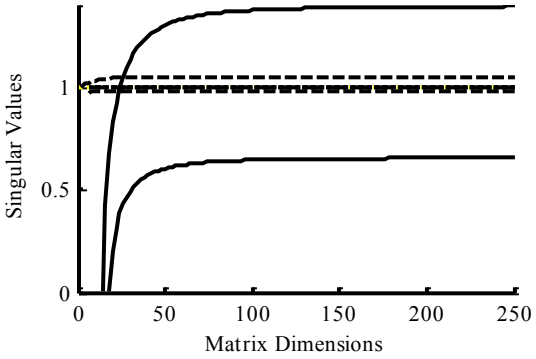
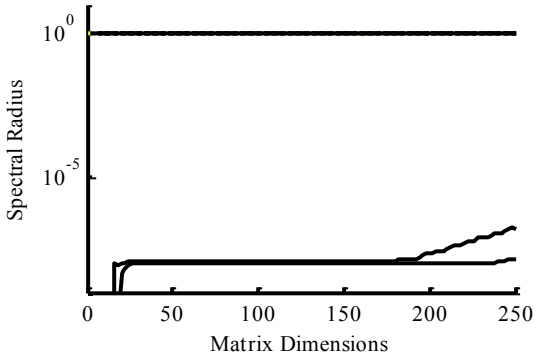


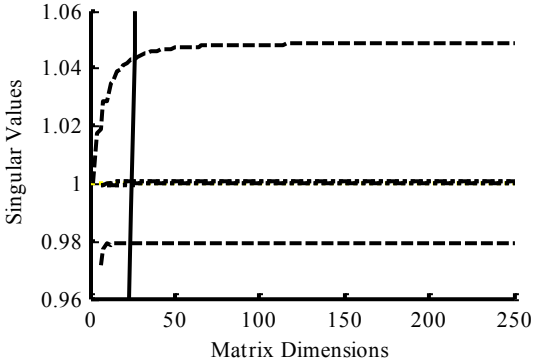

**Figure 2-12. Reqs for $|\lambda|$ < 1 of 3rd Order Plant**



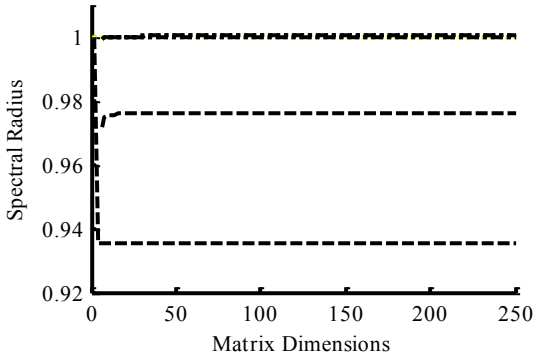

**a. Singular Value No Zoom**



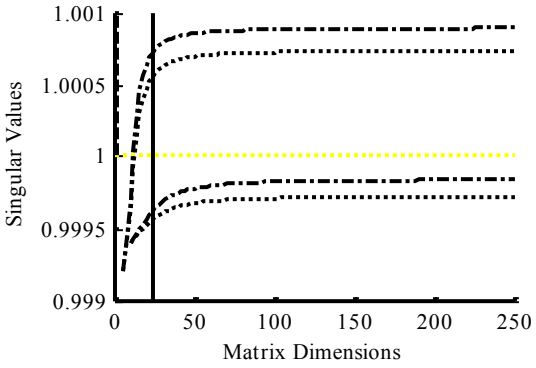

**d. Spectral Radius No Zoom**



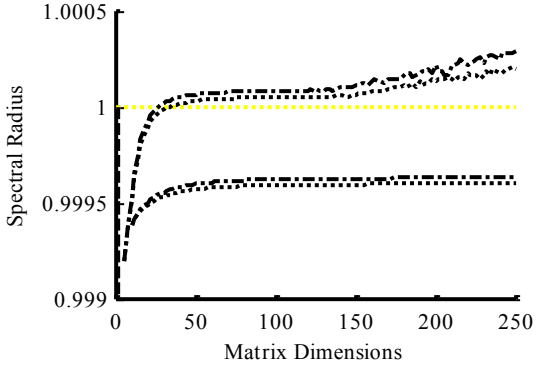

**b. Singular Value Zoom One**




**e. Spectral Radius Zoom One**



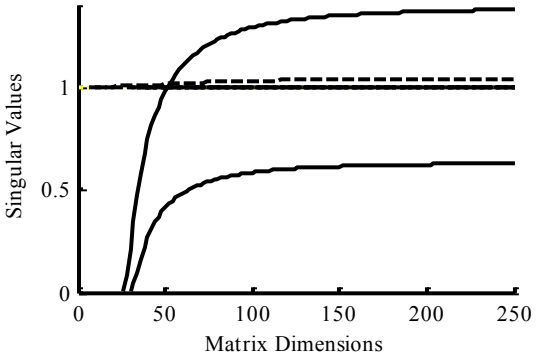

**c. Singular Value Zoom Two**



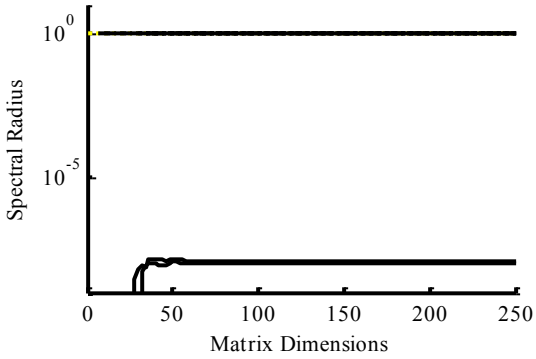

**f. Spectral Radius Zoom Two**

**Figure 2-13. Effects of Expansion on Noiseless Parameters Sampled at 50 Hz**

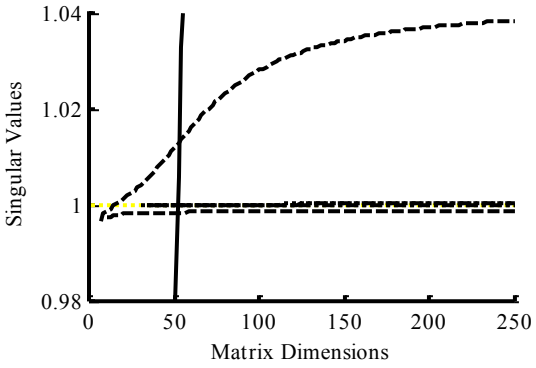
**—— Pseudoinverse  – – – Partial Isometry  – · – Euclidean Norm  · · · · Quadratic Cost**



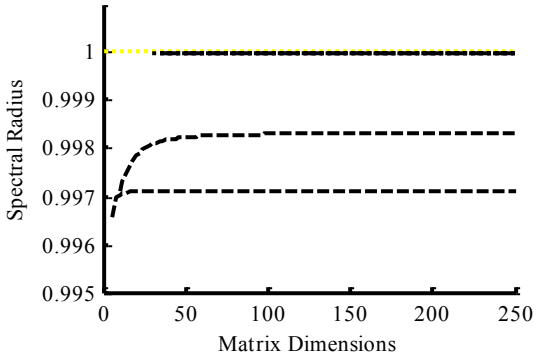

**a. Singular Value No Zoom**



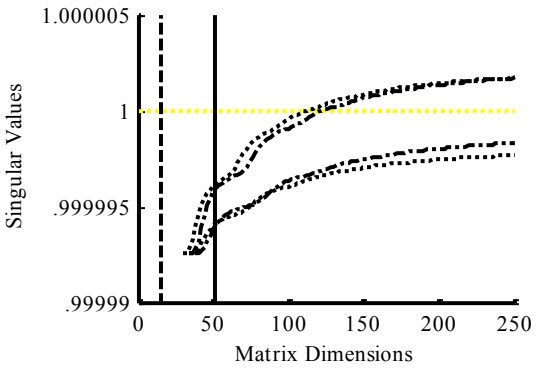

**d. Spectral Radius No Zoom**



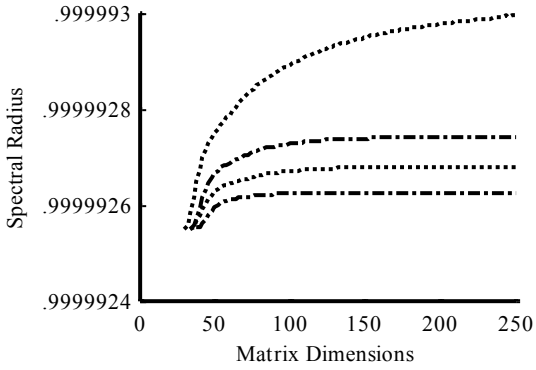

**b. Singular Value Zoom One**




**e. Spectral Radius Zoom One**



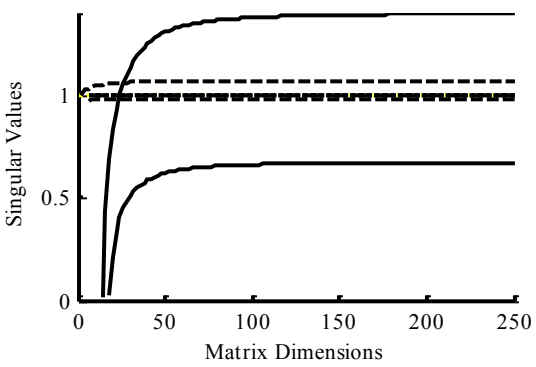

**c. Singular Value Zoom Two**



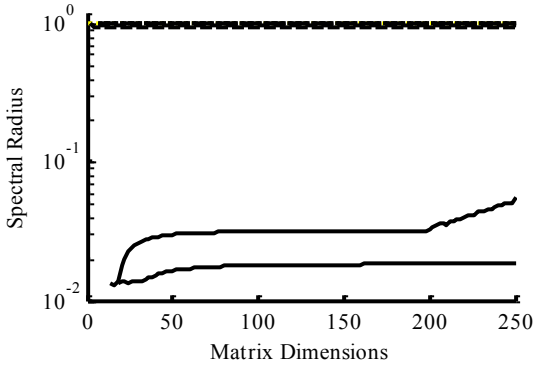

**f. Spectral Radius Zoom Two**

**Figure 2-14. Effects of Expansion on Noiseless Parameters Sampled at 100 Hz**

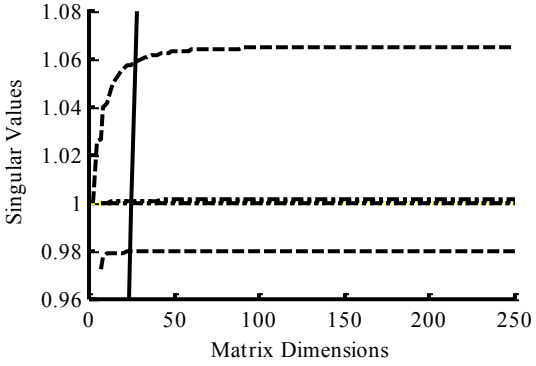
**—— Pseudoinverse  – – – Partial Isometry  – · – Euclidean Norm  · · · · Quadratic Cost**



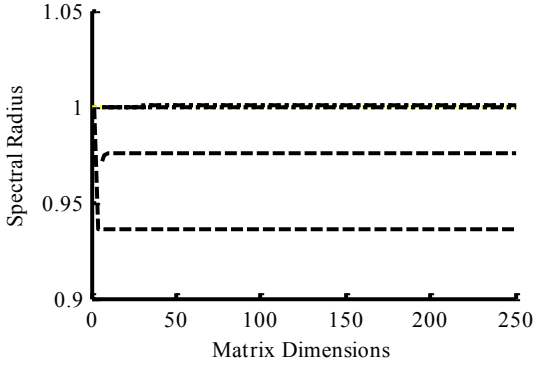

**a. Singular Value No Zoom**



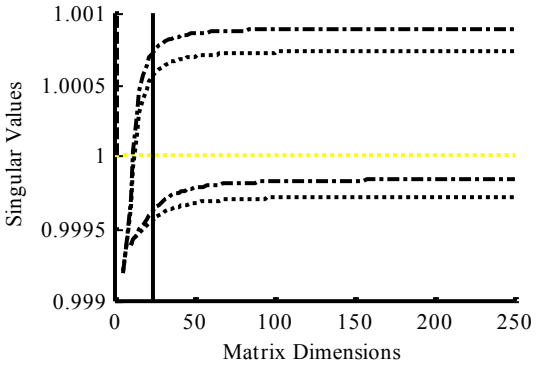

**d. Spectral Radius No Zoom**



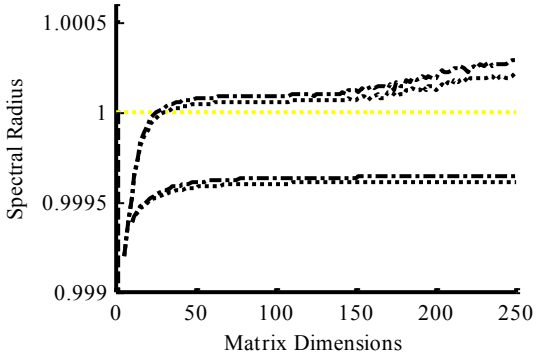

**b. Singular Value Zoom One**




**e. Spectral Radius Zoom One**



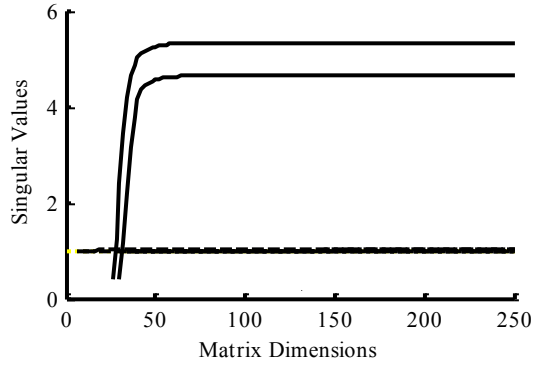

**c. Singular Value Zoom Two**



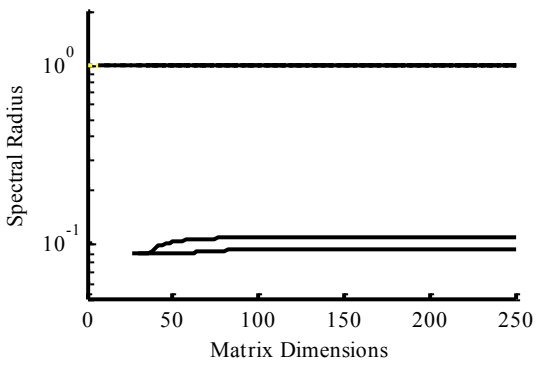

**f. Spectral Radius Zoom Two**

**Figure 2-15. Effects of Expansion on Noisy Parameters Sampled at 50 Hz**

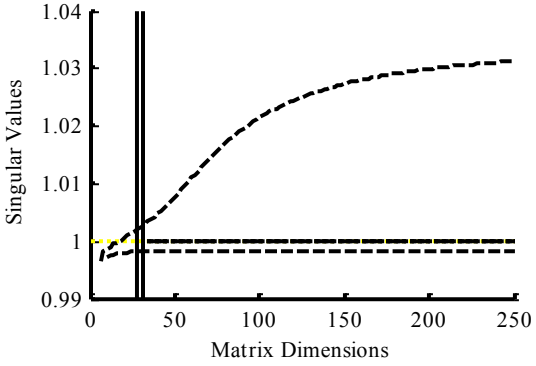
**—— Pseudoinverse  – – – Partial Isometry  – · – Euclidean Norm  · · · · Quadratic Cost**



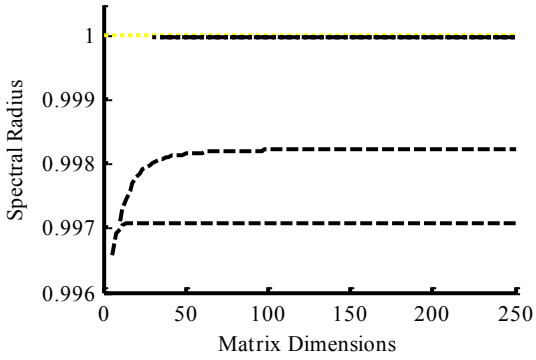

a. Singular Value No Zoom



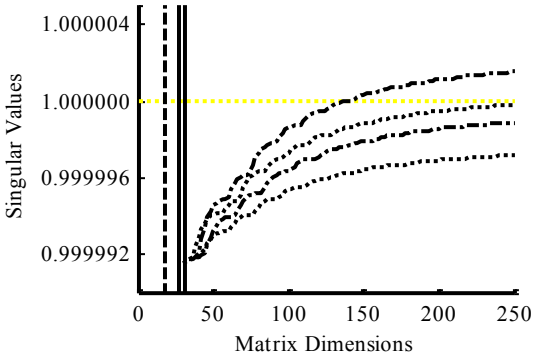

d. Spectral Radius No Zoom



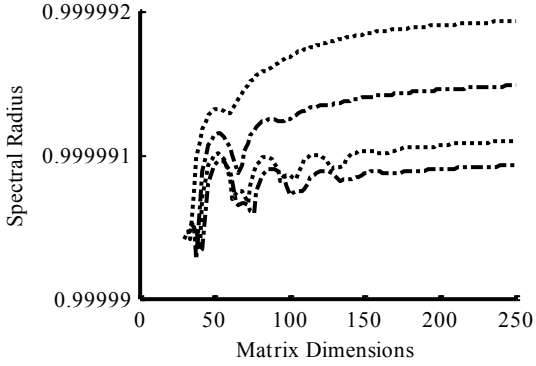

b. Singular Value Zoom One




e. Spectral Radius Zoom One



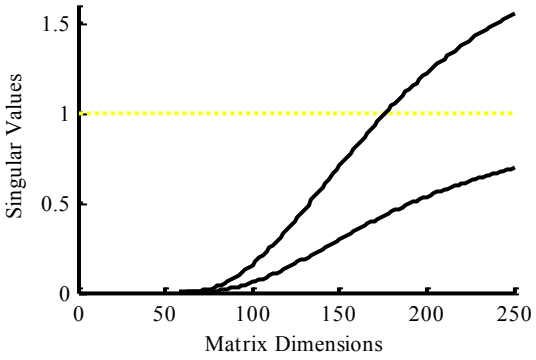

c. Singular Value Zoom Two



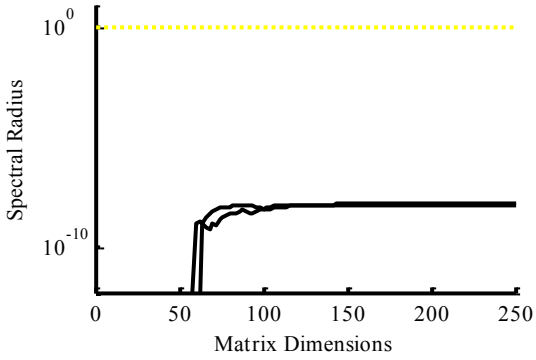

f. Spectral Radius Zoom Two

**Figure 2-16. Effects of Expansion on Noisy Parameters Sampled at 100 Hz**

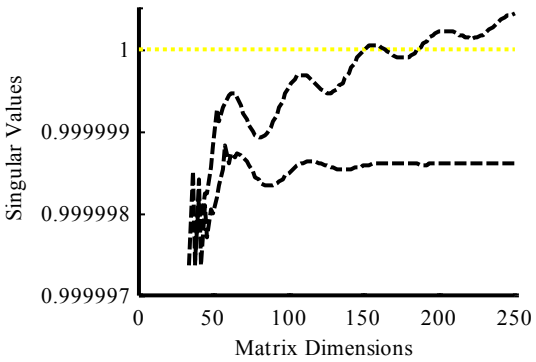
**—— Pseudoinverse  – – – Partial Isometry  – · – Euclidean Norm  · · · · Quadratic Cost**



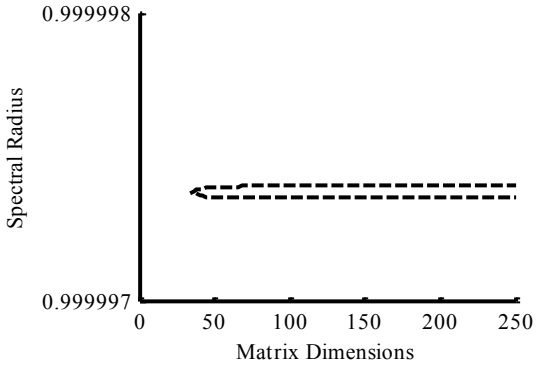

**a. Singular Value No Zoom**




**c. Spectral Radius No Zoom**




**b. Singular Value Zoom One**




**d. Spectral Radius Zoom One**

**Figure 2-17. Effects of Expansion on Noiseless Parameters Sampled at 1000 Hz**

**—— Pseudoinverse** **– – – Partial Isometry**

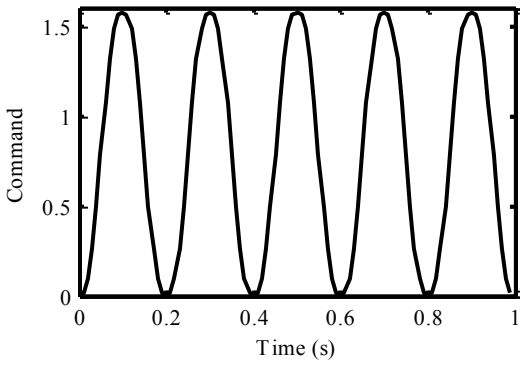



**Figure 2-18. Polynomial Command Signal**

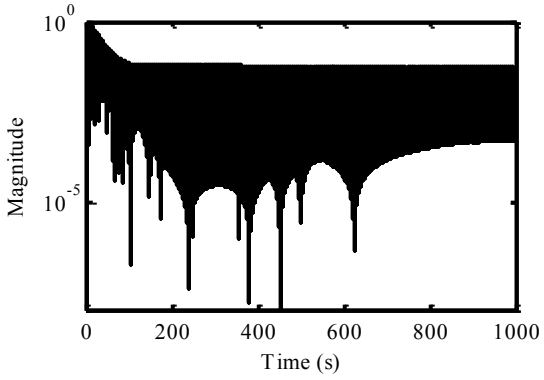



**Figure 2-19. Error with Noiseless Euclidean Norm**

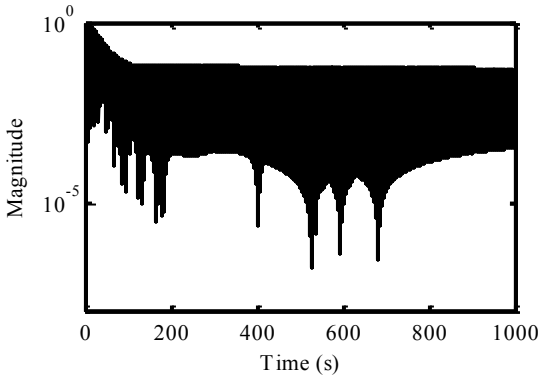



**Figure 2-20. Error with Noiseless Quadratic Cost**

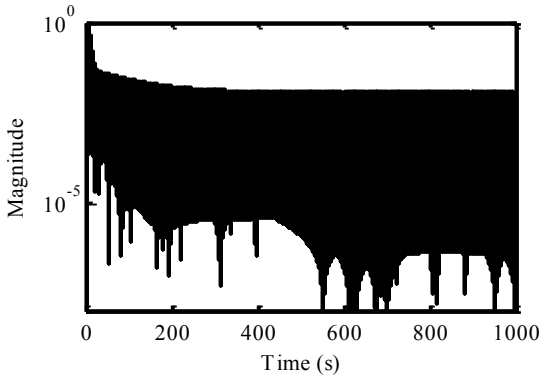



**Figure 2-21. Error with Noiseless Partial Isometry**

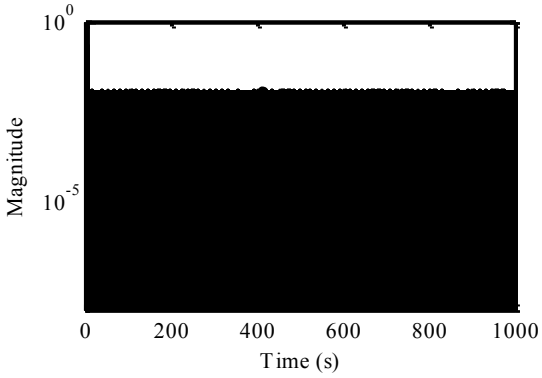



**Figure 2-22. Error with Noiseless Pseudoinverse**

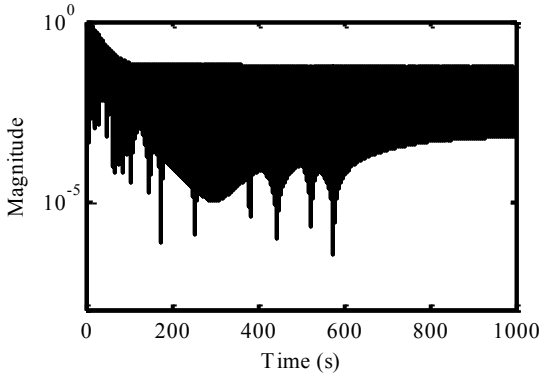



**Figure 2-23. Error with Noisy Euclidean Norm**

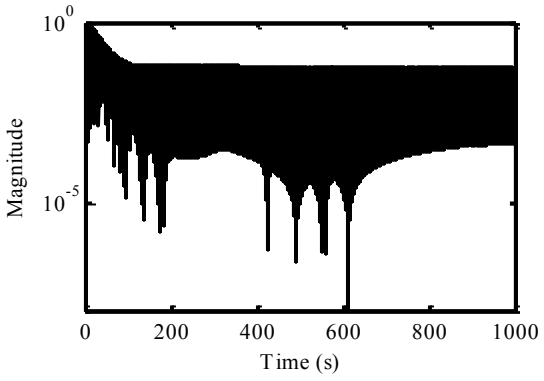



**Figure 2-24. Error with Noisy Quadratic Cost**

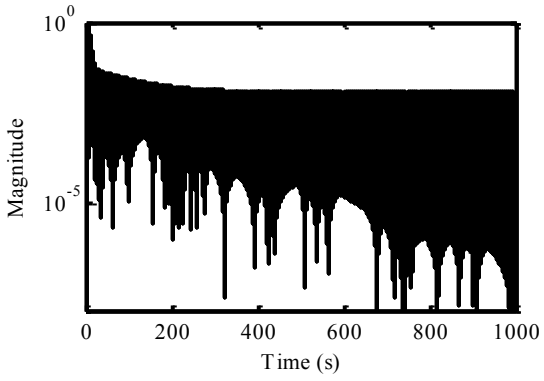



**Figure 2-25. Error with Noisy Partial Isometry**

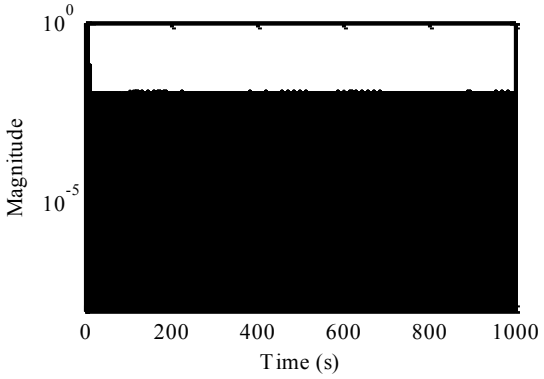



**Figure 2-26. Error with Noisy Pseudoinverse**

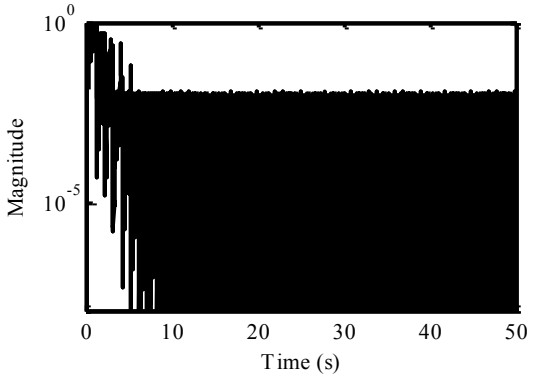



**Figure 2-27. Initial Cycles of Noisy Pseudoinverse**

# CHAPTER THREE: Feedback and Iterative Learning Control With Disturbance Estimators

In Chapter 2, a practical approach to designing and implementation basic ILC when faced with real world limitations such as an imperfect model and pulse response data was presented. In this chapter, the design of a feedback and iterative learning combination controller is presented initially assuming the same disturbance environment model and problem setup, Eqs. (2.1) – (2.3), used in the last chapter. Earlier works restricted the structure of the FBC gain matrices $R$ and $G$ to be of lower sub-triangular structure to ensure causality. That restriction is relaxed in this chapter to allow the FBC gain matrices to be of lower triangular structure and still maintain causality, which can be particularly useful, especially in applications involving a well-conditioned plant.

This chapter also provides additional improvement of such a feedback and learning combination controller through the incorporation of disturbance estimators. The disturbance is estimated both in time and in repetition. Although ILC can handle repeated unknown disturbances in principle, the incorporation of disturbance estimation turns out to be especially advantageous. Both the feedback and learning performance can be improved with such disturbance estimators. In general, disturbance handling of both repeating and non-repeating disturbance is desired, with various works having investigated one or both of these within the context of ILC. Previous research includes disturbance analysis leading to both recursive and explicit expressions for the measured error in terms of desired output and disturbances, in addition to general discussions relating to the effects of disturbances on error evolution. The general aspects of disturbances in learning control as well as approaches to filter the ILC control actions, in

an effort to reduce the impact of noise on the error, have also been explored. Disturbance effects on a closed-loop system and a non-recursive expression of the tracking error, valid for both repeating and non-repeating disturbances, can also be found. This chapter builds upon prior research by developing simple techniques to take any FBC that is non-causal by one time step, defined here to be control gain matrices of lower triangular structure, and combine it with an ILC in an optimal fashion to ensure error convergence through the use of disturbance estimation. Specifically, for the feedback controller, the disturbance estimator is a one-time step behind disturbance estimator. For the iterative learning controller, the disturbance estimator is a one-repetition behind disturbance estimator. This chapter will introduce these disturbance estimation concepts and provide a method to build these disturbance estimators into an existing feedback and iterative learning combination controller. Significant improvement in both tracking performance and learning behavior can be achieved by the incorporation of these disturbance estimators without altering the structures of the existing feedback and iterative learning controllers. As with Chapter 2, the research contributions of this chapter were also published previously and are available in the literature.[98]

The chapter is organized in the following manner. First, the learning control problem is formulated to facilitate finite time-domain (current cycle) analysis for the FBC portion and repetition-domain (previous cycle) analysis for the ILC portion of the combination controller. Then, finite-time inverse FBC in the form of a Finite Impulse Response (FIR) filter, where the inverse of the model is used as controller gains in current cycle feedback form, is investigated. The investigation reveals the surprising result that this current cycle feedback action will not result in the expected inverse solution, but only half of the inverse solution. To approach the exact inverse solution through FIR FBC, high

gains must be used. If an Infinite Impulse Response (IIR) filter is used in place of the FIR filter, then the resultant FBC is shown to approach the desired inverse solution without the need for high gains. Both of these solutions, however, are known to be non-causal by one time step and therefore are not suitable for conventional FBC implementation. So, a technique is developed to convert these controllers into causal form. A physically realizable combination controller, incorporating both current cycle and previous cycle feedback, is then presented. This new implementation is shown to have the same form as the original combination controller and also enables use of the same non-causal FBC gain matrices, converted to causal form, with any desired ILC. The concept of disturbance estimation is then introduced to enhance this new combination controller. The one-step behind disturbance estimator is considered first to supplement the IIR FBC portion of the combination controller. The one-repetition behind disturbance estimator is then incorporated to enhance the ILC portion of the combination controller. Finally, a controller with full disturbance estimation, both one-step behind and one-repetition behind disturbance estimators, is presented. Numerical simulations are conducted using a well-conditioned and ill-conditioned plant to illustrate the effectiveness of this combination controller with disturbance estimators and conclusions are given.

### 3.1: Finite-Time Non-Causal Inverse FBC

It is known that ILC designs are typically restricted to finite-time applications, so that assumption is naturally extended to FBC here, without loss of generality. A hybrid controller, incorporating both finite-time FBC and ILC, where the current control input is a linear combination of the current and previous repetition control input and current and previous repetition tracking error, was presented in earlier work [41],[65]

$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} \tag{3.1}$$

Here, the error vectors $\underline{e}_j = \underline{y}^* - \underline{y}_j$ and $\underline{e}_{j-1}$ represent the difference between the observed output $\underline{y}_j$ at iteration $j$, or $j-1$, and the desired output $\underline{y}^*$ or reference trajectory. Assume for now that the ILC portion of the hybrid controller is removed leaving only the FBC portion, then Eq. (3.1) would be modified to

$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j \tag{3.2}$$

Next, assume that the plant to be controlled using Eq. (3.2) is well-conditioned and a very simple inverse FBC, in the form of a Finite Impulse Response (FIR) filter where the inverse of the model is used as controller gains, is generated

$$\underline{u}_j = P^{-1}\underline{e}_j \tag{3.3}$$

Recall from Eq. (1.2) that the plant model $P$ is lower triangular, hence the inverse, assuming $P$ is well-conditioned, is also lower triangular. Therefore, in the absence of direct feedthrough, which was assumed in this work for reasons given above, this very basic FIR FBC design is non-causal by one time step. Now, neglecting the non-causality aspect, it is reasonable to propose that this implementation should yield the exact inverse solution. Common intuition would suggest that if a plant is invertible and a controller can be designed to perform such an inversion, in theory, it should output the reference signal exactly. However, application to the system model given in Eq. (2.2) shows that the exact inverse solution can never be obtained using such an error feedback controller, even with zero initial conditions and no disturbances present

$$\underline{y}_j = PP^{-1}\underline{e}_j \tag{3.4}$$

$$\underline{y}_j = \frac{1}{2}\underline{y}^* \tag{3.5}$$

Clearly, Eq. (3.5) is not the desired inverse solution; rather, it is only the "half-inverse" solution. If the inverse FIR FBC is modified to allow a scalar multiplication of the controller gains, then the exact inverse solution can be approached

$$\underline{u}_j = \alpha P^{-1} \underline{e}_j \tag{3.6}$$

$$\underline{y}_j = \frac{\alpha}{1+\alpha} \underline{y}^* \tag{3.7}$$

It can be shown that the exact inverse solution is achieved in the limit as $\alpha \to \infty$, resulting in the well known high-gain FBC, which is generally poorly behaved and undesirable. Next, consider an inverse FBC in the form of an Infinite Impulse Response (IIR) filter, again where the inverse of the model is used as controller gains

$$\underline{u}_j = \beta I \underline{u}_j + P^{-1} \underline{e}_j \tag{3.8}$$

$$\underline{y}_j = \frac{1}{2-\beta} \underline{y}^* \tag{3.9}$$

If implemented in this fashion, the exact inverse solution is achieved in the limit as $\beta \to 1$, however if $\beta = 1$, then the input would become undefined in Eq. (3.8) when one tries to solve the equation for $\underline{u}_j$, so such a setup should be avoided. An implementation where $\beta$ is close to one avoids the necessity for high-gain FBC that is required for Eq. (3.7) and can still produce an output that is much better than Eq. (3.5) or the "half-inverse" solution. It should be noted that both the FIR and IIR FBC designs are non-causal by one time step when Eq. (1.2) is considered and therefore not directly suitable for conventional FBC implementation.

### 3.2: Non-Causal FBC and ILC Combination Controller

Now if the ILC portion is reinstated and the controller presented in Eq. (3.1) is considered, then conditions exist to match the learning behavior of the finite-time FBC and ILC combination controller to an ILC only implementation

$$R + S = I \tag{3.10}$$

$$L = (GP + I - R)L^* - G \tag{3.11}$$

For the ILC portion, there are no restrictions with regards to causality, since the control matrices operate on previous cycle information. Following the same approach used for inverse FBC above, an inverse ILC design can be formulated and, using matching conditions Eqs. (3.10) - (3.11), can be used to examine the corresponding combination controller. Now, consider the very basic ILC law

$$\begin{aligned} \underline{u}_j &= \underline{u}_{j-1} + L^* \underline{e}_{j-1} \\ &= \underline{u}_{j-1} + \gamma P^{-1} \underline{e}_{j-1} \end{aligned} \tag{3.12}$$

This basic linear learning control law can be used without loss of generality because more general forms of ILC laws very often equate to laws of this form. The matching combination controller is a more specific version of Eq. (3.1), with $R = 0$, as presented here

$$\begin{aligned} \underline{u}_j &= G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} \\ &= G\underline{e}_j + \underline{u}_{j-1} + [(\gamma - 1)G + \gamma P^{-1}]\underline{e}_{j-1} \end{aligned} \tag{3.13}$$

An examination of Eq. (3.13) reveals the fact that the combination controller can be designed to be of the form of learning with current cycle error feedback only, by setting $L = 0$ and solving for $G$ using Eq. (3.11), or learning with previous cycle error only, by setting $G = 0$ in Eq. (3.13), as desired. These two different forms can be interchanged, since it can be shown that they are mathematically equivalent, or simultaneously incorporated. Interested readers can refer to Appendix A for the mathematical proof of this equivalence. For the case when learning with current cycle error feedback only, the combination controller will be

$$\underline{u}_j = \underline{u}_{j-1} + \frac{\gamma}{1-\gamma} P^{-1} \underline{e}_j \tag{3.14}$$

The other case, when learning with previous cycle error only, is the inverse ILC controller given in Eq. (3.12). A comparison of these two controllers highlights the fact that ILC will produce the exact inverse solution, assuming zero initial conditions and no disturbances, without the need for high gain. This can be seen by setting $\gamma = 1$ in Eq. (3.12) and implementing as follows

$$\underline{y}_j = P\left(\underline{u}_{j-1} + P^{-1}\underline{e}_{j-1}\right) = \underline{y}_{j-1} + \underline{y}^* - \underline{y}_{j-1} \tag{3.15}$$

$$\underline{y}_j = \underline{y}^* \tag{3.16}$$

It was determined earlier that if an FIR FBC implementation is used, then it is necessary to use high gains to get the desired inverse solution. Here, when the analogous current cycle error feedback learning controller in Eq. (3.14) is equated to the previous cycle error learning controller of Eq. (3.12) with $\gamma = 1$, high gains naturally result. As stated earlier, these types of high gain controllers are generally considered to be poorly behaved and undesirable, not to mention they are non-causal in the common framework which is considered for this work.

## 3.3: Non-Causal to Causal Combination Controller Without Disturbance Estimation

It is possible to allow the FBC gain matrices to be of non-causal form by one time step and convert to an equivalent causal form that is suitable for conventional FBC design. For the case where the FBC is causal to begin with, which requires the control matrices to have gains of zero on the main diagonal and all upper diagonals, the conversion will have no effect. Here, however, the gain matrices considered are all non-

causal by one time step and must be converted to causal form before they can be implemented. So, begin with the general combination controller written as follows

$$\underline{u}_j = (R_D + R_S)\underline{u}_j + (G_D + G_S)\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} \tag{3.17}$$

Here subscripts $D$ and $S$ represent the diagonal and sub-diagonal portions, respectively, of the corresponding lower-triangular control matrices. For systems without direct feedthrough, earlier works restricted the structure of the FBC gain matrices $R$ and $G$ to be of lower sub-triangular structure to ensure causality.[41],[65] That restriction is loosened here to allow the FBC gain matrices to be of lower triangular structure and still maintain causality. To enable this, the diagonal matrices of Eq. (3.17) must be factored out to ensure the current input is independent of future error terms limited to a posteriori operation only. This is done by substitution using two additional equations, one for the error vector, $\underline{e}_j = \underline{y}^* - \underline{y}_j$, and the standard, disturbance free, convolution sum solution, $\underline{y}_j = P\underline{u}_j = (P_D + P_S)\underline{u}_j$, to replace the output term. Using these two equations and going through some algebraic manipulations yields the desired causal combination controller. Focusing on the first iteration, where only the FBC portion is active, the causal feedback law is

$$\underline{u}_0 = \overline{R}\underline{u}_0 + \overline{G}\underline{e}_0 + G^*\underline{y}^* \tag{3.18}$$

$$\begin{aligned} \overline{R} &= (I + G_D P_D - R_D)^{-1}(R_S - G_D P_S) \\ \overline{G} &= (I + G_D P_D - R_D)^{-1} G_S \\ G^* &= (I + G_D P_D - R_D)^{-1} G_D \end{aligned} \tag{3.19}$$

Here, the originally lower-triangular FBC matrices $R$ and $G$ have been converted to causal, lower sub-triangular matrices $\overline{R}$ and $\overline{G}$ as necessary for implementation. Since vector $y^*$ is fully known at all times, $G^*$ does not need to be lower sub-triangular for

implementation. For the case where an IIR FBC design is utilized, the causal feedback law becomes

$$\begin{aligned}
\underline{u}_0 &= \beta I \underline{u}_0 + (G_D + G_S)\underline{e}_0 \\
&= \beta I \underline{u}_0 + \left[\left(G_D\left(\underline{y}^* - \underline{y}_0\right) + G_S\underline{e}_0\right)\right] \\
&= \beta I \underline{u}_0 + \left[-G_D\underline{y}_0 + G_S\underline{e}_0 + G_D\underline{y}^*\right] \\
&= \beta I \underline{u}_0 + \left[-G_D(P_D + P_S)\underline{u}_0 + G_S\underline{e}_0 + G_D\underline{y}^*\right] \\
&= (\beta I - G_D P_D)\underline{u}_0 + \left[-G_D P_S \underline{u}_0 + G_S\underline{e}_0 + G_D\underline{y}^*\right] \\
\underline{u}_0 &= \left(\frac{1}{\mu - \beta}\right)\left[-G_D P_S \underline{u}_0 + G_S\underline{e}_0 + G_D\underline{y}^*\right]
\end{aligned} \tag{3.20}$$

Here, $\mu$ represents the scalar gain of the matrix $I + G_D P_D = \mu I$ which results when the above equation is simplified. When this result is compared to the original IIR FBC design given in Eq. (3.8), it becomes clear that the Eq. (3.20) controller is well-suited for direct practical implementation, even with $\beta = 1$, because there are no potential issues with the input becoming undefined or causality concerns. Through use of this causal FBC law, it is possible to achieve the effect of high-gain feedback without the need for high gains.

This causal FBC is used for the first repetition; however the controller must be modified to incorporate learning for successive repetitions. Again using matching conditions Eqs. (3.10) - (3.11), the causal FBC and ILC combination controller, for $j \geq 1$, becomes

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + \overline{S}\underline{u}_{j-1} + \overline{L}\underline{e}_{j-1} + G^*\underline{y}^* \tag{3.21}$$

$$\begin{aligned}
\overline{R} &= (I + G_D P_D - R_D)^{-1}(R_S - G_D P_S) \\
\overline{G} &= (I + G_D P_D - R_D)^{-1} G_S \\
\overline{S} &= (I + G_D P_D - R_D)^{-1} S \\
\overline{L} &= (I + G_D P_D - R_D)^{-1} L \\
G^* &= (I + G_D P_D - R_D)^{-1} G_D
\end{aligned} \tag{3.22}$$

As before, $\overline{R}$ and $\overline{G}$ are lower sub-triangular as required. Since $\overline{S}$ and $\overline{L}$ operate on data from previous trials, they do not need to be lower suw-triangular. Also, since $G^*$ operates on $y^*$, which is always known, it is not required to be lower sub-triangular either. Readers interested in the detailed derivation of the above controller are encouraged to refer to Appendix B.1 for more information. Recalling that the matching conditions ensure the learning behavior of the combination controller follows that of the ILC only implementation given in Eq. (3.12), it is important to establish conditions to ensure convergence in the repetition-domain. It was shown in earlier work that propagation of the tracking error in the repetition-domain is [21],[41]

$$\underline{e}_j = (I - PL^*)^j \underline{e}_0 \tag{3.23}$$

If the inverse ILC law specified in Eq. (15) is considered, is it easy to show that the necessary condition to ensure convergence in the repetition-domain is

$$|1 - \gamma| < 1 \tag{3.24}$$

It is important to note that the above non-causal to causal controllers assumed the ideal case when there are no disturbances present, for initial analysis purposes. Such an assumption should not be taken in practice, so the above formulations are now modified to include disturbance estimation.

### 3.4: Disturbance Estimation

It is understood that ILC handles unknown deterministic disturbances, present every iteration, automatically. This can be easily shown in the repetition-domain, even if disturbances are unmeasurable and, generally, without the need for an explicit model. For FBC, however, the presence of unmeasurable disturbances is generally known to degrade performance. This is true regardless of disturbances being deterministic or stochastic, unlike ILC, which is only ill equipped for stochastic errors and disturbances. In practical applications, all different types of disturbances can potentially enter the process and, in such situations, the causal combination controller proposed in Eqs. (3.21) - (3.22) will have performance limitations. Thus the control actions generated, in the presence of a disturbance, will cause the error to digress from the convergence track predicted by Eq. (3.23) for learning and deteriorate the performance of the subsequent cycles. The resulting propagation of the tracking error in the repetition-domain will in fact be

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - (S + PG)^{-1}PG_D\underline{d} \tag{3.25}$$

Interested readers are encouraged to refer to Appendix C.2 for the detailed derivation of this tracking error propagation result. Clearly, convergence to zero error is not possible with this controller even when only a repeating disturbance is present, as has been assumed until this point. Now consider another problem where learning control is applicable, but the deterministic errors and disturbances go through a sort of evolution phase in the repetition-domain before settling. The evolution can be an adjustment period following changes to the system itself, such as hardware modifications and replacements, or to a dynamic operating environment. This phenomenon can persist in any iterative process involving a system which routinely starts up from a dormant or shut

down state and can be referred to as a cold start.[99] These types of practical problems, whether it be a repeating disturbance that iteratively persists or evolves before settling into a persistent state, are known to plague common learning control applications. For such a disturbance environment, use of Eqs. (2.1) – (2.3) is no longer valid since non-repeating disturbances must now be generally assumed. Therefore, the remainder of this chapter will utilize the general learning control formulation, Eqs. (1.1) – (1.8), introduced in Chapter 1.

One potential way to address both repeating and evolving disturbances, within the context of this work, is the use of disturbance estimation. For the case of a slowly fluctuating disturbance, a discrete time control process can be aided with knowledge of what the disturbance was during the last time step. If the change in the disturbance for successive time steps in not large, this knowledge can help improve the learning process in ILC applications. Here disturbance estimation is now utilized to re-establish the originally presented optimized error propagation given in Eq. (3.23) irrespective of the presence of a disturbance. Define the following

$$\begin{bmatrix}\Delta(1)\\ \Delta(2)\\ \Delta(3)\\ \vdots\\ \Delta(p)\end{bmatrix}=\begin{bmatrix}0 & 0 & 0 & \cdots & 0\\ 1 & 0 & 0 & \ddots & \vdots\\ 0 & 1 & \ddots & \ddots & 0\\ \vdots & \ddots & \ddots & 0 & 0\\ 0 & \cdots & 0 & 1 & 0\end{bmatrix}\begin{bmatrix}\Delta(1)\\ \Delta(2)\\ \Delta(3)\\ \vdots\\ \Delta(p)\end{bmatrix}+\begin{bmatrix}\varepsilon(1)\\ \varepsilon(2)\\ \varepsilon(3)\\ \vdots\\ \varepsilon(p)\end{bmatrix} \tag{3.26}$$

$$\underline{\Delta}_j = D\underline{\Delta}_j + \underline{\varepsilon} \tag{3.27}$$

Here epsilon $\underline{\varepsilon}$ is the one-step behind error of $\underline{\Delta}_j$, which will approach zero as the change in the disturbance for successive time steps approaches zero. This definition will now be used to aid the FBC portion of the combination controller, through the use of

one-step behind estimation. The ILC portion of the combination controller will then also be improved, through incorporation of one-repetition behind estimation.

### 3.5: One-Step Behind Disturbance Estimator

To incorporate a one-step behind disturbance estimator, the practical formulation of Eq. (1.3), where a disturbance is generally present, is considered. Start with the inverse FIR FBC design given in Eq. (3.6) and implement it with Eq. (1.3) to output

$$\underline{y}_j = \frac{\alpha}{1+\alpha}\underline{y}^* + \frac{1}{1+\alpha}\underline{\Delta}_j \tag{3.28}$$

The exact inverse solution is again achieved in the limit as $\alpha \to \infty$, so high-gain control also works in the presence of unknown disturbance. Next, consider the inverse IIR FBC design given in Eq. (3.8) and using this type of controller will produce

$$\underline{y}_j = \frac{1}{2-\beta}\underline{y}^* + \frac{1-\beta}{2-\beta}\underline{\Delta}_j \tag{3.29}$$

Here, the exact inverse solution is again achieved in the limit as $\beta \to 1$, even in the presence of unknown disturbances. As before, however, if $\beta = 1$ the input again becomes undefined, so such a setup should be avoided. An implementation where $\beta$ is close to one avoids the necessity for high-gain while still achieving the same high-gain effect.

In order to implement the above FBC designs, both will generally need to be converted to causal form as done before, but this time incorporating one-step behind disturbance estimation. For the FIR FBC start with the general version of Eq. (3.3) written as follows

$$\underline{u}_j = (G_D + G_S)\underline{e}_j \tag{3.30}$$

Here subscripts $D$ and $S$ are defined in the same fashion as in Eq. (3.17) but for the corresponding lower-triangular error FBC gain matrix. The causal version of this

feedback law with a built-in one-step behind disturbance estimator is derived starting with Eq. (3.30) as follows

$$\begin{aligned}
\underline{u}_j &= G_D\underline{y}^* - G_D\left[(P_D + P_S)\underline{u}_j + \left(D\underline{\Delta}_j + \underline{\varepsilon}\right)\right] + G_S\underline{e}_j \\
&= G_D\underline{y}^* - G_D\left[(P_D + P_S)\underline{u}_j + D\left(\underline{y}_j - P\underline{u}_j\right) + \underline{\varepsilon}\right] + G_S\underline{e}_j \\
&= G_D\underline{y}^* - G_D\left[(P_D + P_S)\underline{u}_j + D\left(\underline{y}_j - P\underline{u}_j\right)\right] + G_S\underline{e}_j - G_D\underline{\varepsilon} \\
&= G_D\underline{y}^* - G_D\left[(P_D + P_S)\underline{u}_j + D\left(\underline{y}_j - P\underline{u}_j\right) + \left(D\underline{y}^* - D\underline{y}^*\right)\right] + G_S\underline{e}_j - G_D\underline{\varepsilon} \\
&= G_D\underline{y}^* - G_D\left[P_D\underline{u}_j + (P_S - DP)\underline{u}_j + D\left(\underline{y}_j - \underline{y}^*\right) + D\underline{y}^*\right] + G_S\underline{e}_j - G_D\underline{\varepsilon} \\
&= G_D\underline{y}^* - G_D\left[P_D\underline{u}_j + (P_S - DP)\underline{u}_j + D\underline{y}^* - D\underline{e}_j\right] + G_S\underline{e}_j - G_D\underline{\varepsilon}
\end{aligned} \tag{3.31}$$

In the above derivation, Eq. (1.3) is used to define the output in terms of the input and disturbance. Then use of Eq. (3.27) enables one-step behind disturbance estimation with available a posteriori input-output information. Finally, collecting all the diagonal matrices and simplifying results in

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + G^*\underline{y}^* + H\underline{\varepsilon} \tag{3.32}$$

$$\begin{aligned}
\overline{R} &= (I + G_D P_D)^{-1}(G_D DP - G_D P_S) \\
\overline{G} &= (I + G_D P_D)^{-1}(G_S + G_D D) \\
G^* &= (I + G_D P_D)^{-1}(G_D - G_D D) \\
H &= -(I + G_D P_D)^{-1} G_D
\end{aligned} \tag{3.33}$$

Assuming that the disturbance does not change substantially from one time step to the next, we estimate the disturbance at the current time step to be equivalent to the disturbance at the previous time step. This approximation is accomplished by neglecting the epsilon term in Eq. (3.32) to arrive at the same general form of causal FBC as Eq. (3.18), but with the gain matrices newly designed for one-step behind disturbance

estimation. Depending on how the error FBC gain matrix $G$ is defined, the output can be shown to be the "half-inverse" or exact inverse solution as before, however one-step behind estimating of the disturbance will occur for both cases here.

To reformulate the IIR FBC design to include one-step behind estimation, start by rewriting the general control law, Eq. (3.2), as follows

$$\underline{u}_j = (R_D + R_S)\underline{u}_j + (G_D + G_S)\underline{e}_j \tag{3.34}$$

Again, subscripts $D$ and $S$ are defined as in Eq. (3.30) and, using the FIR FBC results, the causal version of this feedback law with an integrated one-step behind disturbance estimator is obtained

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + G^*\underline{y}^* + H\underline{\varepsilon} \tag{3.35}$$

$$\begin{aligned} \overline{R} &= (I + G_D P_D - R_D)^{-1}(R_S - G_D P_S + G_D D P) \\ \overline{G} &= (I + G_D P_D - R_D)^{-1}(G_S + G_D D) \\ G^* &= (I + G_D P_D - R_D)^{-1}(G_D - G_D D) \\ H &= -(I + G_D P_D - R_D)^{-1} G_D \end{aligned} \tag{3.36}$$

Here the epsilon term should also be neglected and the same general form of causal FBC reoccurs, but again with the gain matrices designed for disturbance estimation. Like Eq. (3.18), this IIR FBC design is also well-suited for direct practical implementation since there are no causality concerns or potential for an undefined input. However, this causal IIR FBC law is improved because not only is it able to achieve the high-gain effect, it also implicitly performs one-step behind disturbance estimation. This FBC can be combined with an ILC using the matching conditions Eqs. (3.10) - (3.11) or implemented independently. However, without the incorporation of learning, this implementation has no way to utilize previous repetitions information and cannot take full

advantage of the experience gained from interactions with the real world, which are fundamental goals of learning control applications.

### 3.6: One-Repetition Behind Disturbance Estimator

The one-step behind estimator will aid the FBC portion of the combination controller, but does nothing to enhance the ILC portion of the combination controller, which requires incorporation of one-repetition behind estimation. In order for learning to work in the presence of disturbance, the causal FBC and ILC combination controller is reformulated starting again with Eq. (3.17) above. For the first cycle, since there is no previous repetition information available for disturbance estimation, the same causal FBC design developed earlier in Eqs. (3.18) – (3.19) can be used. For the second iteration forward, one-repetition behind estimation of the disturbance is incorporated

$$\begin{aligned} \underline{u}_j &= R\underline{u}_j + G_D\underline{y}^* - G_D\left(P\underline{u}_j + \underline{\Delta}_j\right) + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} \\ &= R\underline{u}_j + G_D\underline{y}^* - G_D\left[P\underline{u}_j + \left(\underline{y}_{j-1} - P\underline{u}_{j-1}\right)\right] + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} \end{aligned} \tag{3.37}$$

Again, collecting the diagonal matrices and simplifying produces the desired combination controller

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + \overline{S}\underline{u}_{j-1} + \overline{L}\underline{e}_{j-1} \tag{3.38}$$

$$\begin{aligned} \overline{R} &= (I + G_D P_D - R_D)^{-1}(R_S - G_D P_S) \\ \overline{G} &= (I + G_D P_D - R_D)^{-1} G_S \\ \overline{S} &= (I + G_D P_D - R_D)^{-1}(S + G_D P) \\ \overline{L} &= (I + G_D P_D - R_D)^{-1}(L + G_D) \end{aligned} \tag{3.39}$$

The detailed development of the above controller is provided in Appendix B.2 and interested readers are referred there for the full derivation.

This controller is more general than the original causal FBC and ILC combination controller given in Eqs. (3.21) – (3.22) since it does not require use of a $G^*$ term. The one-repetition behind estimation allows this term to be integrated into the $\overline{L}$ gain matrix through implicit use of previous cycle information. With the use of this form of estimation, the resulting combination controller is able to effectively compensate for any purely repeating disturbance and the error will follow the convergence track predicted by Eq. (3.23) for learning as desired. The resulting propagation of the tracking error in the repetition-domain is predicted to be

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - (S + PG)^{-1}(S + PG_D)\delta_j \underline{d} \tag{3.40}$$

Readers interested in the detailed derivation of this tracking error propagation expression are encouraged to refer to Appendix C.3 for more information. Clearly, this controller is an improvement from the original causal FBC and ILC combination controller given in Eqs. (3.21) – (3.22) because it can converge to zero error even in the presence of a repeating disturbance. However, inclusion of the one-step behind estimator can enhance the combination controller even further.

### 3.7: Controller With Full Disturbance Estimation

To implement a combination controller with both one-step behind and one-repetition behind disturbance estimators, start again with Eq. (3.17), the general combination controller. For the first iteration, where only the FBC portion of the combination controller is active, the causal FBC law with an integrated one-step behind disturbance estimator, developed earlier, is used

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + G^*\underline{y}^* \tag{3.41}$$

$$\overline{R} = (I + G_D P_D - R_D)^{-1}(R_S - G_D P_S + G_D DP)$$
$$\overline{G} = (I + G_D P_D - R_D)^{-1}(G_S + G_D D) \tag{3.42}$$
$$G^* = (I + G_D P_D - R_D)^{-1}(G_D - G_D D)$$

Notice that this is the same controller given in Eqs. (3.35) – (3.36) but without the epsilon term. From the second iteration forward, a new controller is derived by combining the above one-step behind disturbance estimator law with a one-repetition behind disturbance estimator law as follows

$$\underline{u}_j = R\underline{u}_j + G_D\underline{y}^* - G_D\left[P\underline{u}_j + \left(D\underline{\Delta}_j + \underline{\varepsilon}\right)\right] + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1}$$
$$D\underline{\Delta}_j = D\left(\underline{y}_j - P\underline{u}_j\right) \quad ; \quad \underline{\varepsilon} = \left(\underline{\Delta}_j - D\underline{\Delta}_j\right) = \left(\underline{y}_{j-1} - P\underline{u}_{j-1}\right) - D\left(\underline{y}_{j-1} - P\underline{u}_{j-1}\right) \tag{3.43}$$

From Eq. (3.43), it can be seen that one-step behind estimation is used to estimate the current cycle disturbance and one-repetition behind estimation is used to estimate the one-step behind error epsilon $\underline{\varepsilon}$ which is neglected for the first iteration. For the remaining cycles, epsilon is estimated as shown in Eq. (3.43) and the resultant combination controller is

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + \overline{S}\underline{u}_{j-1} + \overline{L}\underline{e}_{j-1} \tag{3.44}$$

$$\overline{R} = (I + G_D P_D - R_D)^{-1}(R_S - G_D P_S + G_D DP)$$
$$\overline{G} = (I + G_D P_D - R_D)^{-1}(G_S + G_D D)$$
$$\overline{S} = (I + G_D P_D - R_D)^{-1}(S + G_D P - G_D DP) \tag{3.45}$$
$$\overline{L} = (I + G_D P_D - R_D)^{-1}(L + G_D - G_D D)$$

Interested readers are encouraged to refer to Appendix B.3 for the detailed derivation of this controller utilizing full disturbance estimation.

Here again note that this controller is more general than the original causal FBC and ILC combination controller without disturbance estimation since there is no $G^*$ term. The

use of full estimation enables this combination controller to also effectively compensate for any purely repeating disturbance and, additionally, to better handle an evolving repeating disturbance, introduced earlier. In fact, this combination controller can closely follow the convergence track predicted by Eq. (3.23) for learning even during the evolution or transient phase of an evolving repeating disturbance. Here, the tracking error propagation in the repetition-domain is anticipated to be

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - (S + PG)^{-1}\big(S + PG_D(I - D)\big)\delta_j\underline{d} \tag{3.46}$$

For comparison purposes, the use of the basic ILC law given in Eq. (3.12) will result in tracking error propagation, in the presence of an evolving disturbance, following

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - \delta_j\underline{d} \tag{3.47}$$

The detailed derivations for both of the tracking error propagation expressions presented in Eqs. (3.46) and (3.47) are provided in Appendix C.4 and C.1 respectively. A comparison of Eqs. (3.25), (3.40), (3.46), and (3.47) shows that if a combination controller using lower triangular FBC gain matrices is desired, disturbance estimation is necessary to handle any purely repeating disturbance. If disturbance considerations are expanded to include evolving repeating disturbances, full disturbance estimation shows potential to maximize disturbance rejection.

### 3.8: Numerical Simulation

Numerical simulations were conducted on MATLAB 7.7 to illustrate the effectiveness of the above feedback and iterative learning combination controllers. Two different types of system models are utilized to cover commonly encountered practical situations. A well-conditioned system model will effectively consider control problems in which a designer has the option to implement some of the inverse controllers discussed above. It is very important to remember that in order to be able to use an inverse controller in

practice, a relatively accurate system model is generally required. An ill-conditioned system model will effectively consider control problems where the inverse controllers are not realizable design options. With both models, lower triangular, hence non-causal by one time step, combination controllers are allowed. Simulations then demonstrate the effect of including disturbance estimators, one-repetition behind and full disturbance estimation, in the ideal case and the practical case where a repeating disturbance is generally present. Simulations with an evolving repeating disturbance are also conducted.

### 3.8.1: Implementation Using Well-Conditioned and Ill-Conditioned Plant

The first setup to be simulated is the most ideal case where the plant is well-conditioned and there is no disturbance affecting the control process. For this setup, a third order system model with the following transfer function is considered

$$Y(s) = G(s)U(s) = \frac{a{\omega_n}^2(s^2 + 2s + 2)}{(s + a)\left(\mathrm{s}^2 + 2\,\zeta\ \omega_n s + {\omega_n}^2\right)}U(s) \tag{3.48}$$

For the numerical simulations $a = 8.8$, $\omega_n = 37$, and $\zeta = 0.5$, and the continuous time model was discretized using a ZOH, as described in the learning formulation section, using a 100 Hz sample rate. Notice that the continuous time transfer function has a pole excess of one. Therefore, no zeros are introduced by the discretization process that can potentially be outside the unit circle.

An ill-conditioned plant is considered also using a third order system model, but with a transfer function with a pole excess of three as follows

$$Y(s) = G(s)U(s) = \left(\frac{a}{s + a}\right)\left(\frac{{\omega_n}^2}{\mathrm{s}^2 + 2\,\zeta\ \omega_n s + {\omega_n}^2}\right)U(s) \tag{3.49}$$

As before, for the numerical simulations $a = 8.8,\ \omega_n = 37,$ and $\zeta = 0.5,$ and the continuous time model was again discretized using a ZOH at a 100 Hz sample rate. When the pole excess is three or more and the sample time is fast enough, a zero is introduced outside the unit circle by the discretization process. This means that the inverse problem that is addressed by learning control, of finding the input needed to produce the desired output, is working with the inverse of a discrete transfer function which is unstable. Therefore, this plant is considered ill-conditioned in this context and inverse control is not possible. Furthermore, with an ill-conditioned plant, perfect tracking is not possible even in the ideal case where there is no disturbance present.

**3.8.2: Results**

Using the well-conditioned plant modeled by Eq. (3.48) and the ill-conditioned plant modeled by Eq. (3.49), numerical simulations were conducted to determine the effectiveness of the combination controllers presented in this chapter at following a polynomial command signal based on

$$y^*(t) = \pi\left(5\left(\frac{t}{t_p/2}\right)^3 - 7.5\left(\frac{t}{t_p/2}\right)^4 + 3\left(\frac{t}{t_p/2}\right)^5\right) \tag{3.50}$$

A plot of the desired trajectory is given in Figure 3-1 with a fundamental frequency of 2 Hz. The equation used to generate the first quarter of this command is Eq. (3.50) and the second quarter is merely a reflection of the first quarter. Then that signal is repeated to produce the entire command. This trajectory has the unique characteristic that it has continuous position, velocity, and acceleration starting from rest and along the segment boundaries described above, which implies that it will induce relatively smooth operation. Also, the fundamental frequency of this command is set by specifying $t_p$, its period. For

the simulations involving disturbances, Figure 3-2 shows the repeating disturbance, which purposely includes noise to introduce higher frequency components, and Figure 3-3 shows the evolving disturbance. The first series of simulations conducted started with the most ideal case, a well-conditioned plant and no disturbances. All three of the combination controllers simulated performed identically, with results given in Figure 3-4 showing their performance. Next, the repeating disturbance was introduced for the well-conditioned plant and Figures 3-5 – 3-7 show how the controller performed with no disturbance estimation, only one-repetition behind disturbance estimation, and full disturbance estimation, respectively. The second series of simulations were for the ill-conditioned plant and, as before, all three combination controllers performed identically in the no disturbance case, see Figure 3-8 for results. Similarly, Figures 3-9 – 3-11 show the performance of the three controllers in the presence of the repeating disturbance, as before. The final series of simulations involved the evolving repeating disturbance, with Figures 3-12 – 3-13 showing performance on a well-conditioned plant and Figures 3-14 – 3-15 presenting the results on the ill-conditioned plant.

### 3.9: Summary of Findings

Improved combination controllers, incorporating both FBC and ILC, have been developed for learning controller applications in linear systems. These controllers stem from previously derived combination controllers, but with an enhancement that allows for the use of FBC designs which are non-causal by one time step. The controllers are then further advanced through the utilization of disturbance estimators. Numerical simulations confirm that these combination controllers, with disturbance estimation, significantly improve both the tracking performance and the learning behavior.

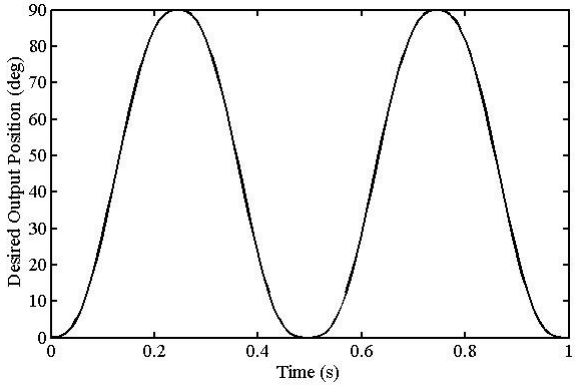



**Figure 3-1. Desired Trajectory**

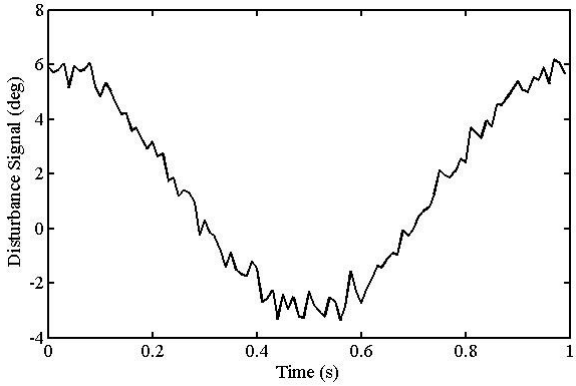



**Figure 3-2. Repeating Disturbance**

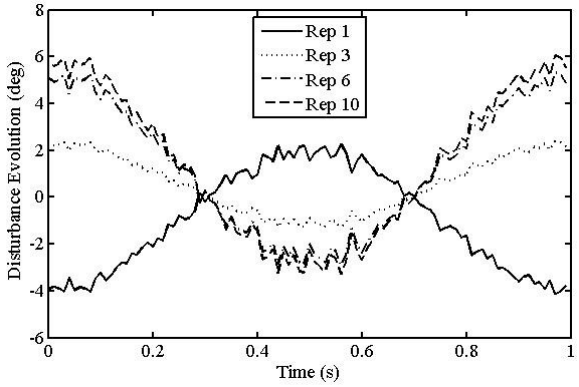



**Figure 3-3. Evolving Repeating Disturbance**

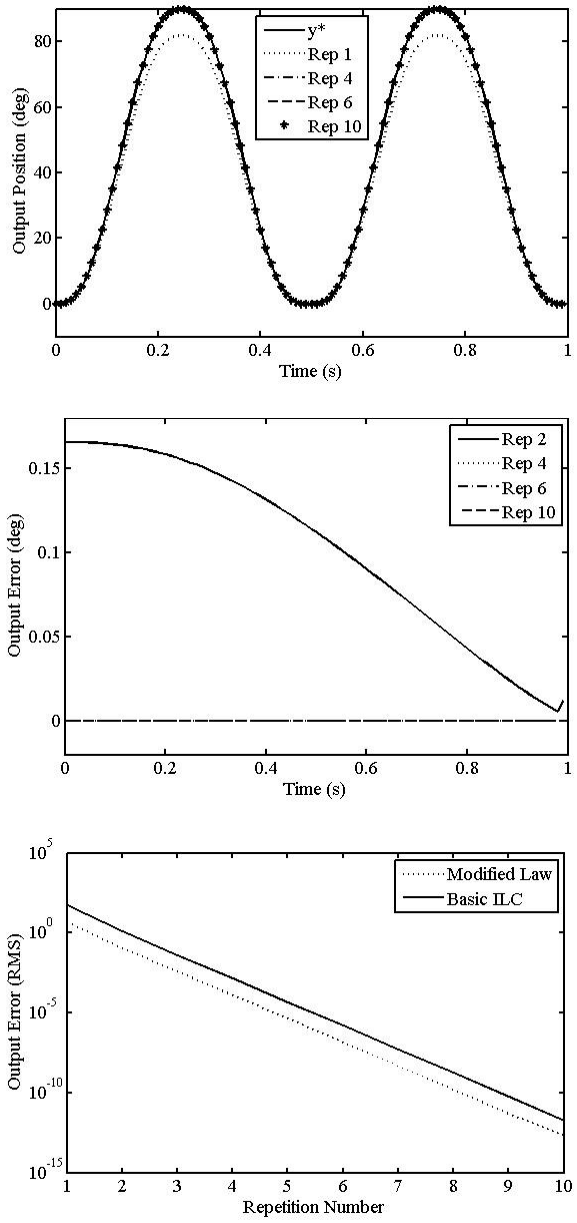



**Figure 3-4. Well-Conditioned Plant With No Disturbance Present**

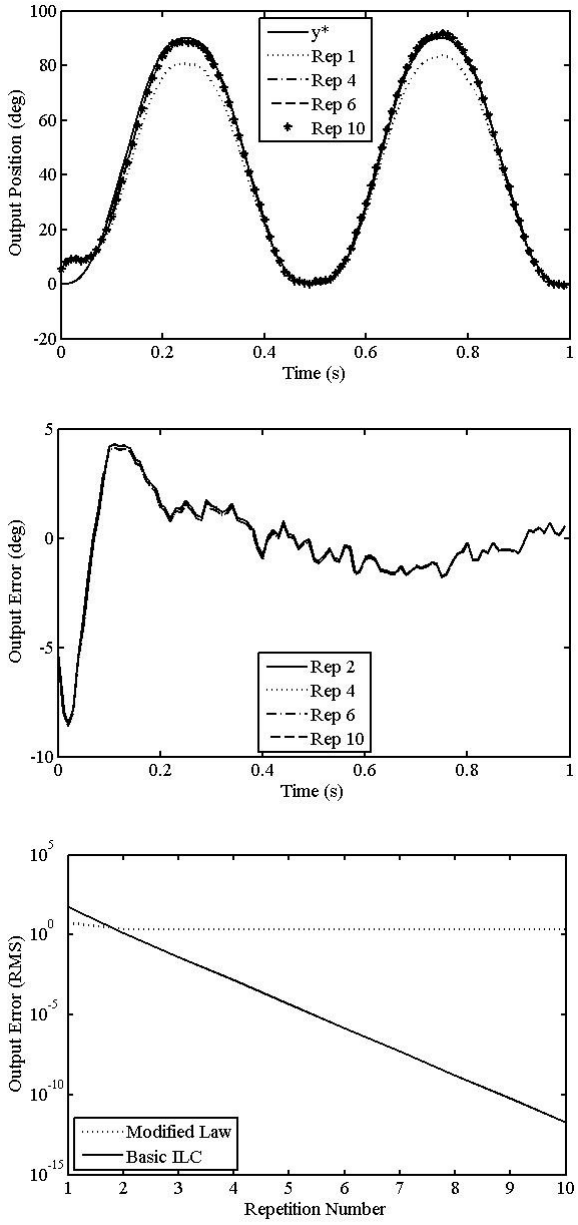



**Figure 3-5. Well-Conditioned Plant With Repeating Disturbance and No Estimation**

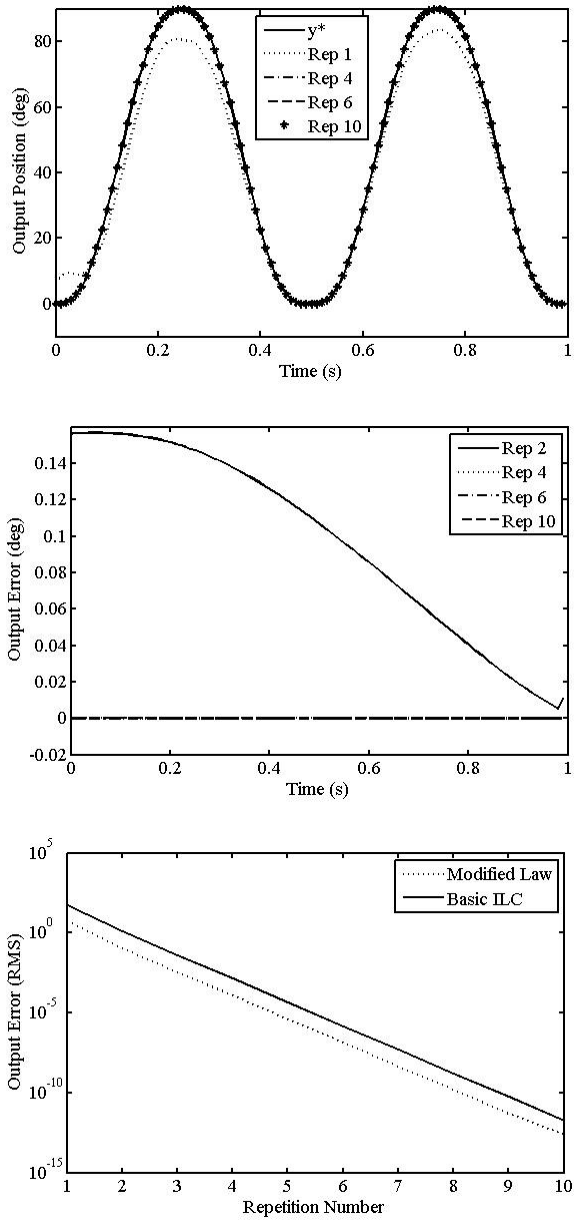



**Figure 3-6. Well-Conditioned Plant With Repeating Disturbance and One Rep Estimation**

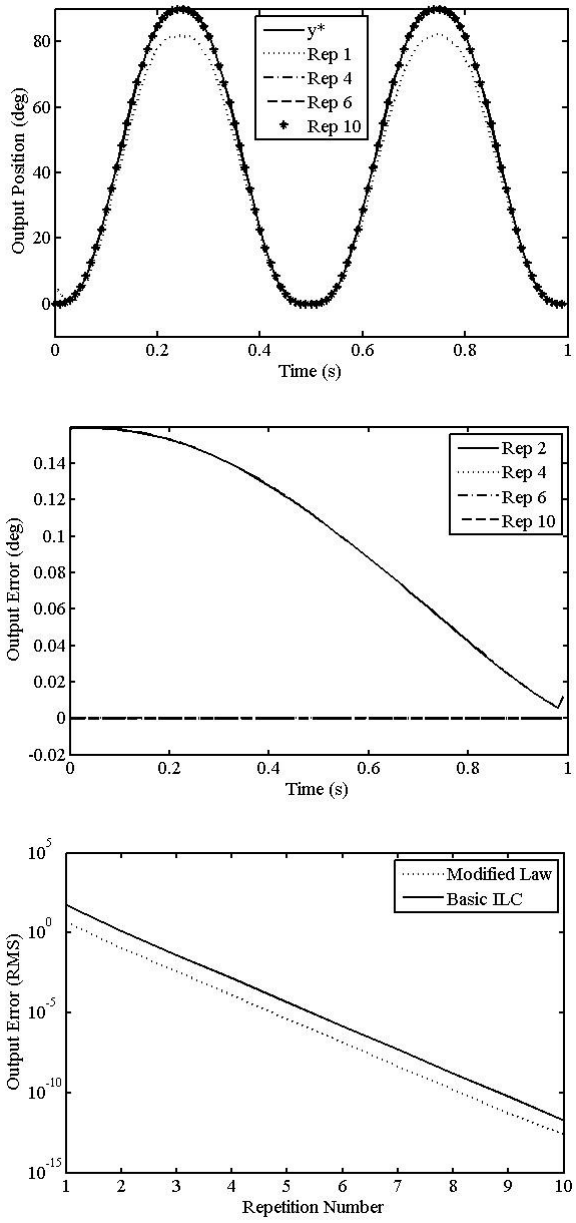
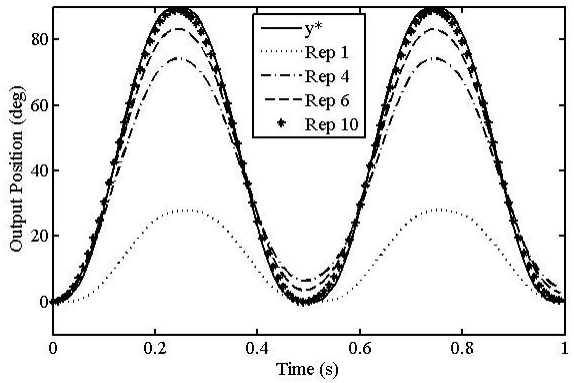
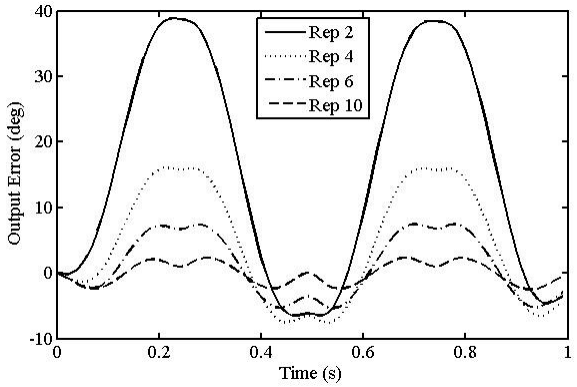



**Figure 3-7. Well-Conditioned Plant With Repeating Disturbance and Full Estimation**



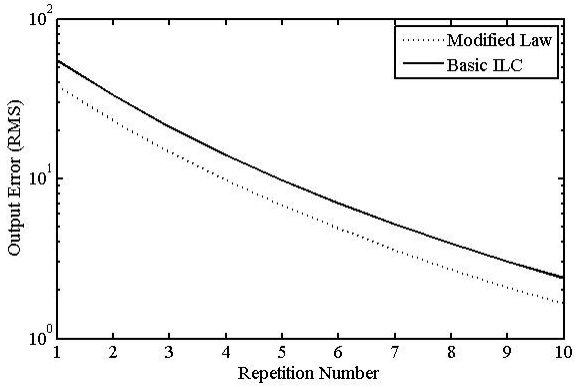
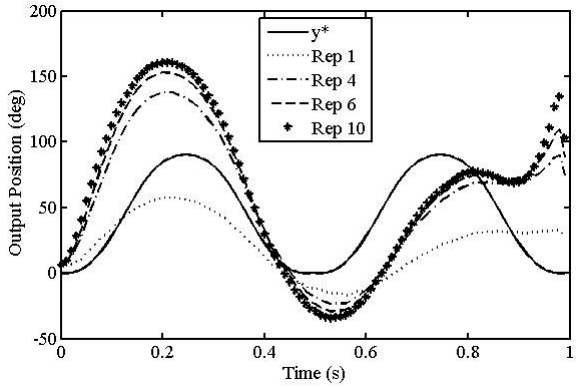
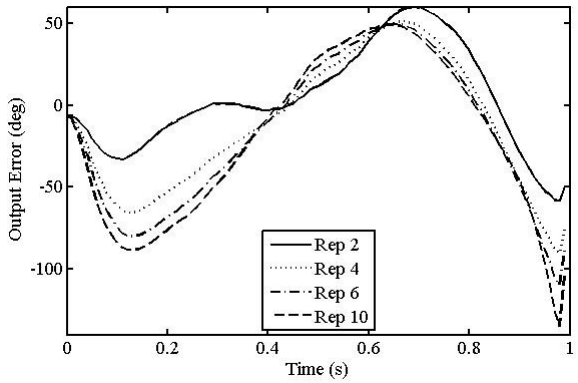




**Figure 3-8. Ill-Conditioned Plant With No Disturbance Present**



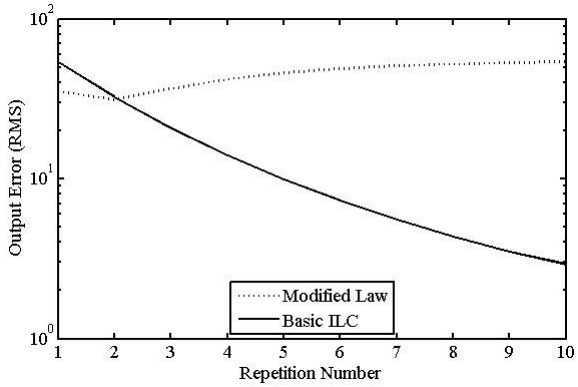




**Figure 3-9. Ill-Conditioned Plant With Repeating Disturbance and No Estimation**

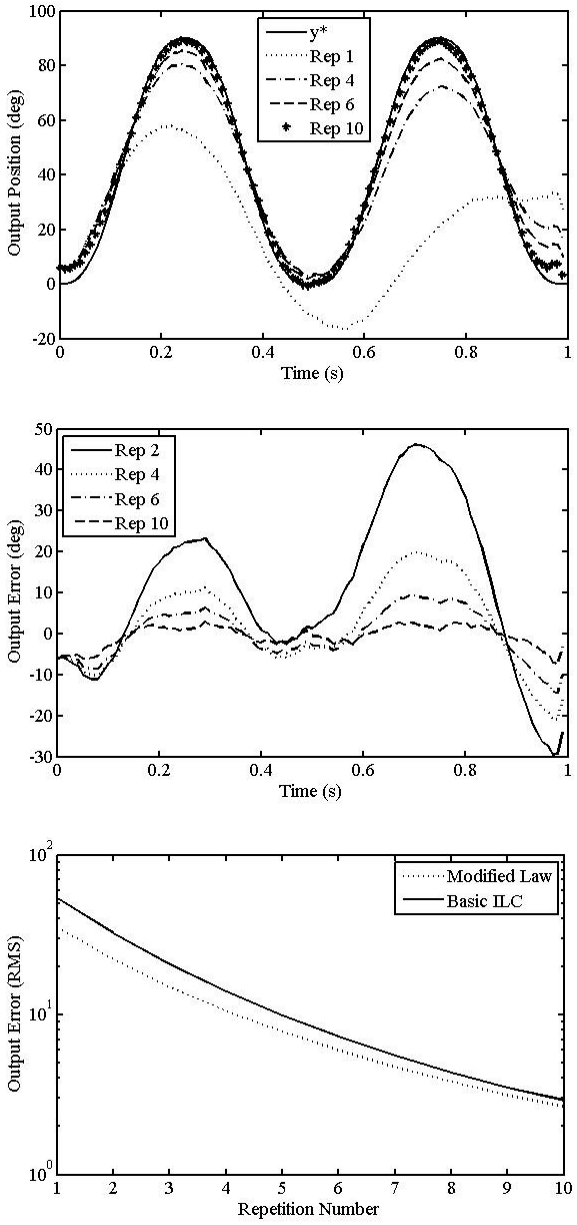



**Figure 3-10. Ill-Conditioned Plant With Repeating Disturbance and One Rep Estimation**

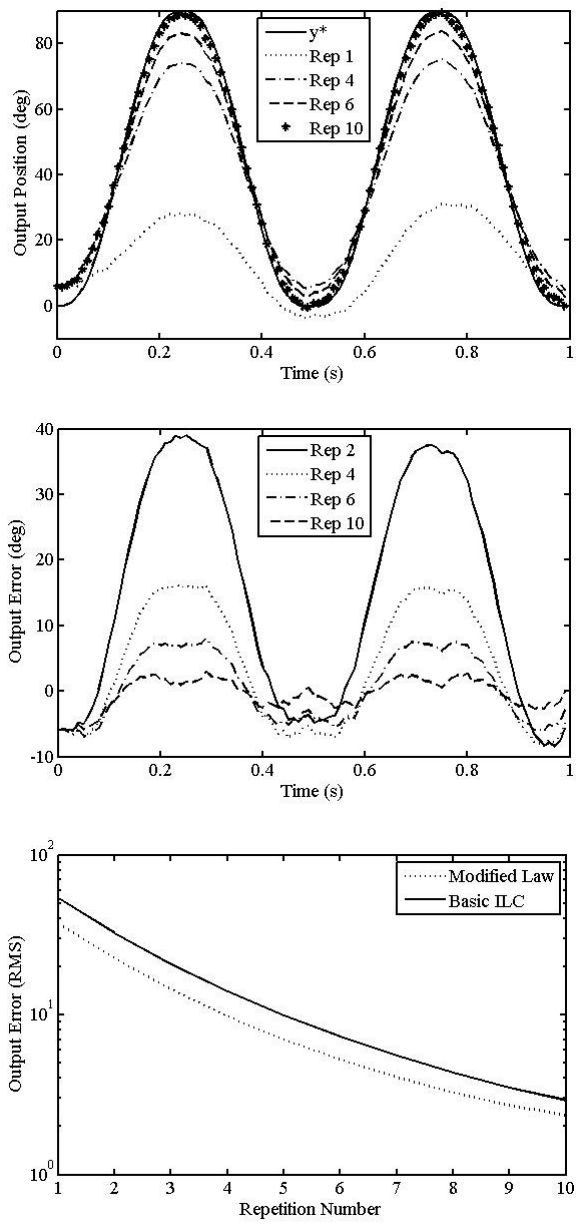



**Figure 3-11. Ill-Conditioned Plant With Repeating Disturbance and Full Estimation**

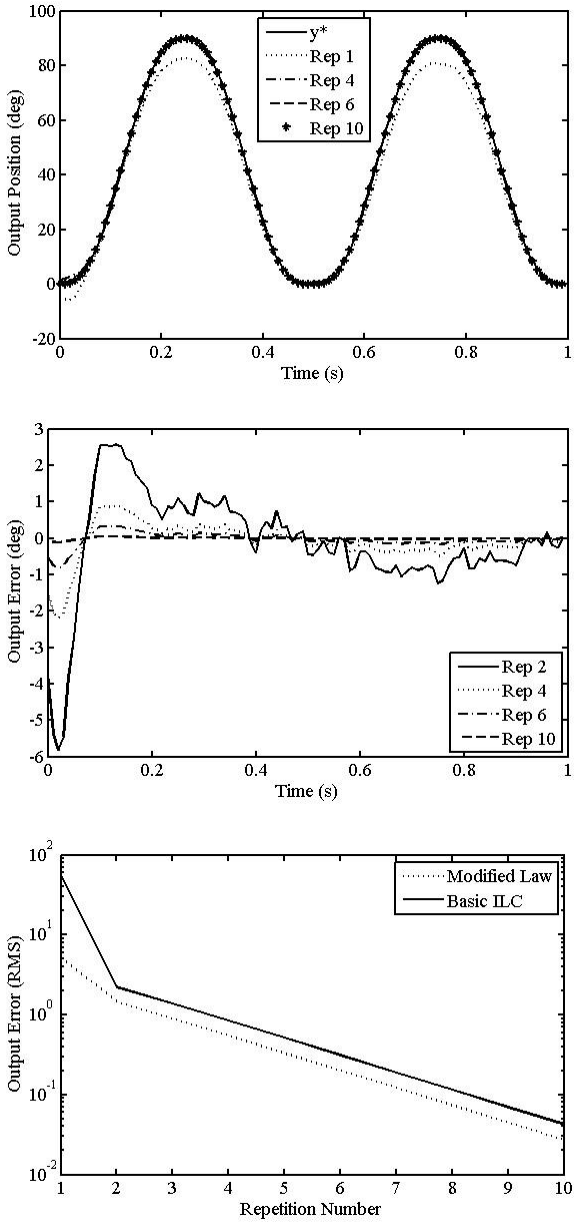



**Figure 3-12. Well-Conditioned Plant With Evolving Disturbance and One Rep Estimation**

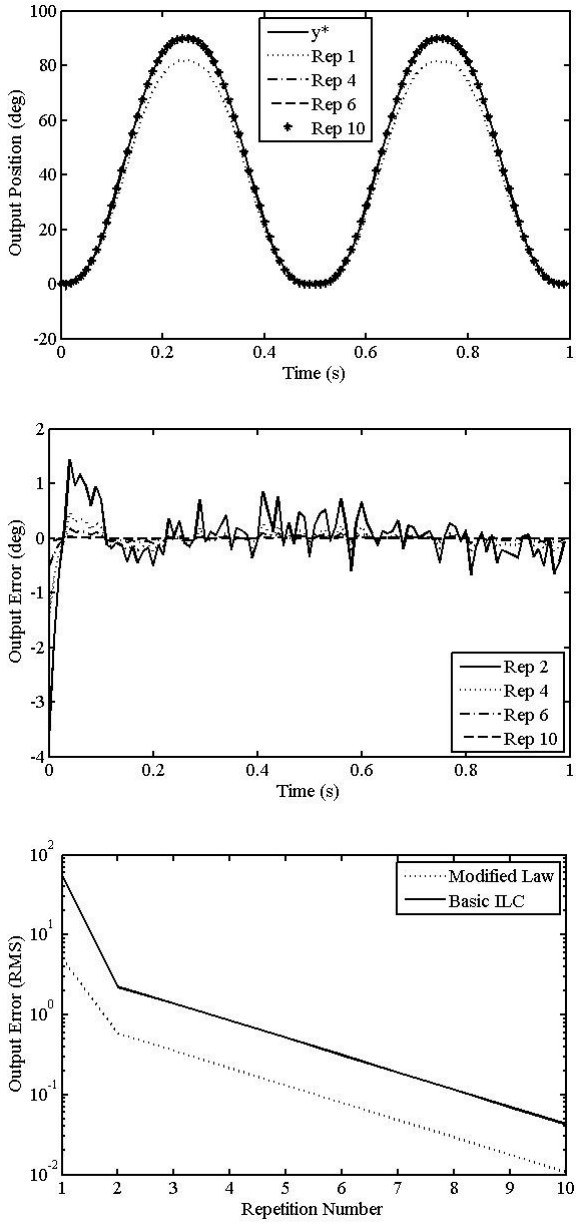



**Figure 3-13. Well-Conditioned Plant With Evolving Disturbance and Full Estimation**

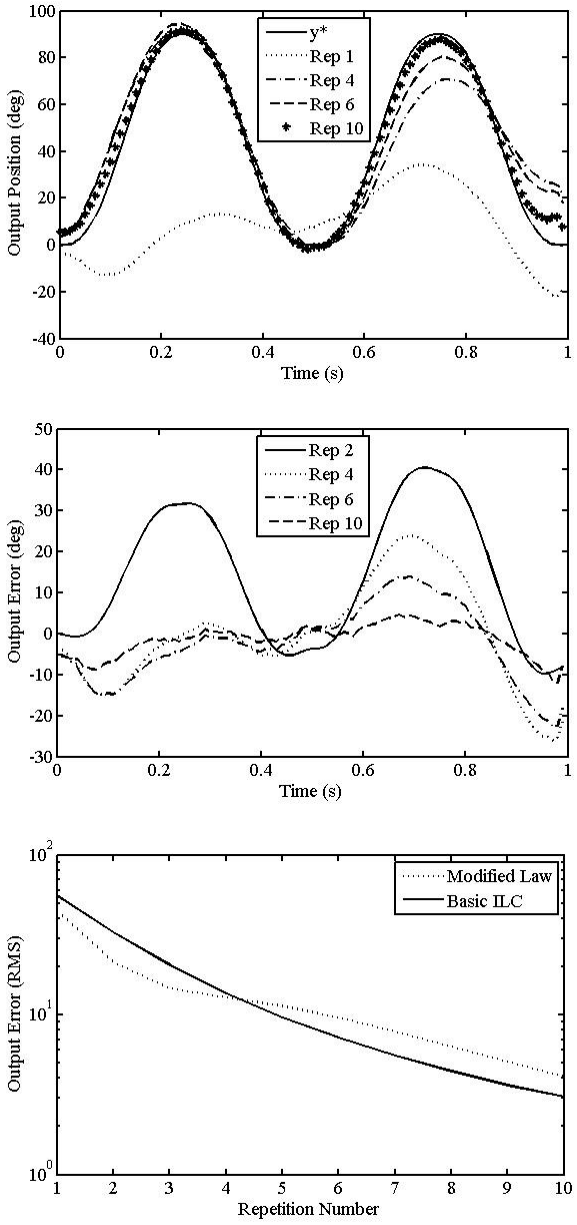



**Figure 3-14. Ill-Conditioned Plant With Evolving Disturbance and One Rep Estimation**

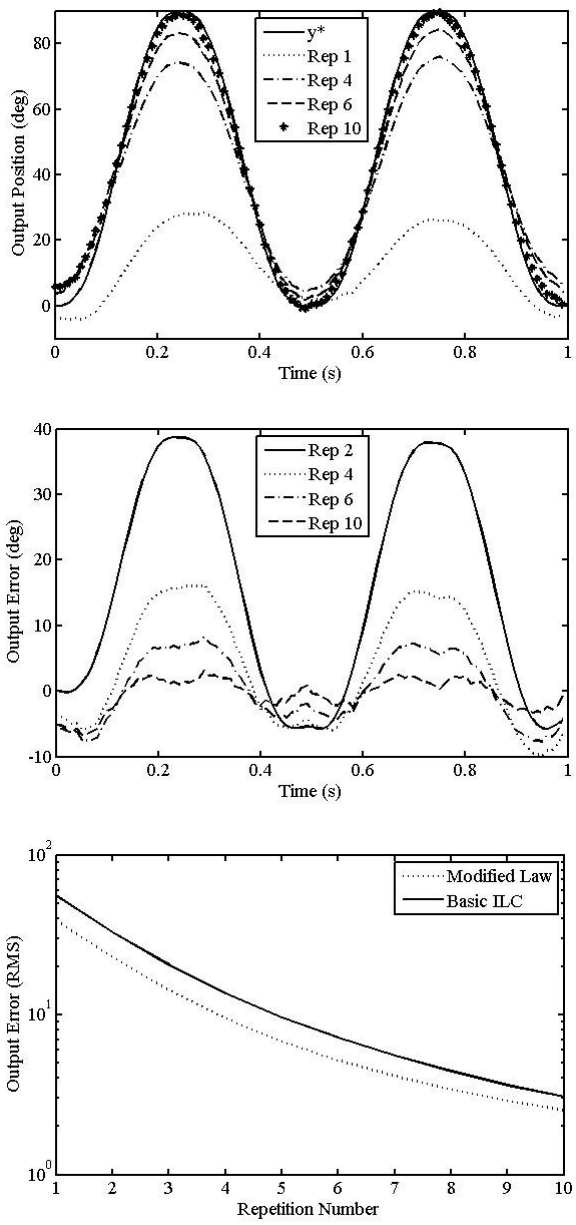



**Figure 3-15. Ill-Conditioned Plant With Evolving Disturbance and Full Estimation**

# CHAPTER FOUR: Optimized Finite-Time Feedback and Iterative Learning Controller Design

This chapter presents a novel approach that enables the design of a finite-time feedback and learning combination controller. The cost function-based approach allows for simultaneously design of both the FBC and ILC portions of the combination controller and naturally facilitates design decisions related to how the control burden is shared between FBC and ILC. If the designer decides the optimal combination controller should exhibit qualities primarily associated with FBC, then such a controller can be easily produced through the appropriate tuning. Similarly, if the optimal controller should have primarily ILC attributes, then it can be designed in an analogous fashion. In many ILC problems, the ideal division of control burden is difficult or impossible to determine initially. So, a combination controller that iteratively adapts in some optimal fashion through interactions with its operating environment may perform best. This chapter presents a method that can achieve these objectives by adjusting a few gains in the basic design, or a few weighting matrices in a more complex and tailored design.

The remainder of this chapter is organized into four main sections. In the first section, the learning control problem is formulated to facilitate finite time-domain (current cycle) analysis for the FBC portion and repetition-domain (previous cycle) analysis for the ILC portion of the combination controller. The second section presents an optimized finite-time FBC and ILC design in six sub-sections. The first sub-section provides background information on the causal $Q$ matrix based finite-time FBC concept. The second sub-section proposes the general $Q$ matrix for finite-time FBC conversion along with some

basic guidelines for implementation. The third and fourth sub-sections present the derivation of the optimal static $Q$ based finite-time FBC and ILC design and the corresponding error propagation equations. The last two sub-sections present the derivation of the optimal dynamic $Q$ based finite-time FBC and ILC design and the corresponding error propagation equations. The third section discusses some general design considerations for both the static and dynamic optimal $Q$ based combination controllers related to training matrices, finite-time sensitivity transfer matrices, and the implementation of conventional FBC using $Q$ matrix based designs. The final section presents the results of numerical simulations using a well-conditioned plant and two different ill-conditioned plants to illustrate the effectiveness of the proposed designs in various practical operating environments involving repeating and random, non-repeating disturbances.

### 4.1: Optimized Finite-Time FBC and ILC Design

The design of finite-time FBC was investigated in a previous study. However, the design of a feedback and learning combination controller was not established.[69] The study presents feedback controllers based on a causal inverse matrix $Q$ of the form

$$\underline{u}_j = Q\underline{y}^* \tag{4.1}$$

This $Q$ matrix is essentially an intermediate step used in the derivation of a linear solution to the initially non-linear problem of finding the gain matrices $R$ and $G$ to implement the feedback controller of the form

$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j \tag{4.2}$$

The steps involved in designing $Q$ and then finding the corresponding gain matrices $R$ and $G$ to implement the controller of Eq. (4.2) require extensive use of the stack operator

and the Kronecker product.[69] The direct relationship between the inverse matrix $Q$ and the associated feedback gain matrices $R$ and $G$ is

$$\begin{aligned} Q &\stackrel{\text{def}}{=} (I - R + GP)^{-1}G \\ &\stackrel{\text{def}}{=} KG \end{aligned} \tag{4.3}$$

Here, the matrix $K$ is introduced purely for convenience because it will be seen that the term $(I - R + GP)^{-1}$ appears repeatedly in numerous subsequent derivations. It is important to clarify that the use of Eq. (4.3) to define the inverse matrix $Q$ directly from any set of causal feedback gain matrices $R$ and $G$ is always acceptable and valid. However, it is even more imperative to establish that these definitions are unidirectional and the use of Eq. (4.3) to find $R$ and $G$ directly from $Q$ in learning control applications is generally invalid. This is due to the fact that in order to enable the general use of Eq. (4.3) to calculate $R$ and $G$ directly from $Q$, the $K$ matrix must be accordingly incorporated into the relationship between the overall plant input and the feedback and learning combination controller. This can be clearly seen through the following derivation, starting with the feedback controller given in Eq. (4.2) combined with the learning control signal $\underline{v}_j$, and use of Eq. (1.3) along with the standard definition for tracking error

$$\begin{aligned} \underline{u}_j &= R\underline{u}_j + G\underline{e}_j + \underline{v}_j \\ \underline{u}_j &= R\underline{u}_j + G\left(\underline{y}^* - P\underline{u}_j - \underline{\Delta}_j\right) + \underline{v}_j \\ &= (I - R + GP)^{-1}\left[G\left(\underline{y}^* - \underline{\Delta}_j\right) + \underline{v}_j\right] \\ &= K\left[G\left(\underline{y}^* - \underline{\Delta}_j\right) + \underline{v}_j\right] \end{aligned} \tag{4.4}$$

It is apparent that the general use of Eq. (4.3) to calculate $R$ and $G$ directly from $Q$ would undo the desired linearization effect because $K$ would now be introduced as an unknown

variable as shown in Eq. (4.4), which is counterproductive. In other words, if the overall plant input is generated using Eq. (4.4), it would be a function of three variables $K$, $G$, and $\underline{v}_j$, all of which are to be designed. If these variables are unknown, then designing them simultaneously using a cost function is again a non-linear problem because of the inherent coupling between $K$ and $G$ that naturally exists with this type of feedback implementation. Therefore, a more suitable conversion will be established in this chapter. Recall, as long as the $Q$ matrix is calculated independently, determining the feedback gain matrices $R$ and $G$ from it is a linear problem. For feedback and learning combination controller implementations, the design of the $Q$ matrix is done simultaneously with the design of the learning control signal. So the overall plant input should be defined as

$$\begin{aligned} \underline{u}_j &= Q\underline{y}^* + \underline{v}_j \\ &= \left(\underline{y}^{*T} \otimes I\right) Q^S + \underline{v}_j \\ &= A_1 \underline{q} + \underline{v}_j \end{aligned} \tag{4.5}$$

Notice that the stack operator, denoted with the superscript $^S$, and the Kronecker product, denoted with the symbol $\otimes$, are both used to permute the original equation. This is done so that the unknown $Q$ matrix can be transformed into an unknown stacked $\underline{q}$ vector, which can then be optimized using a cost function. The design objective is to optimize both $Q$, or $\underline{q}$, and the learning signal $\underline{v}_j$ according to some specified criteria. Once $\underline{q}$, or the elements of $Q$, are determined in the stacked vector form given in Eq. (4.5), they must be transformed back into the $Q$ matrix by reversing the steps used to define the original stacked vectors. It is up to the designer to determine if time varying or time-invariant elements are desired. The use of time-invariant elements will result in a

lower-dimensional problem since the resulting $Q$ matrix would be Toeplitz. Therefore, only a smaller number of unique elements need to be determined, as opposed to the time varying $Q$ matrix solution where significantly more unique gains must be calculated. It is generally recommended that for either design, full freedom be given by using all available gains in the solution. If full freedom is not given, achieving the goal of finding a truly optimal combination of $Q$ and $\underline{v}_j$ may not be possible.

### 4.1.1: General Q Matrix to Finite-Time FBC Conversion for Learning Control Applications

For the moment, assume that such an optimization was done and the corresponding optimal combination of $Q$ and $\underline{v}_j$ have been determined. Furthermore, assume that $Q$ was designed with exact knowledge of the disturbance using the following condition

$$PQ\underline{Y}^* = PQ\left(\underline{y}^* - \underline{\Delta}_j\right) \approx \underline{y}^* - \underline{\Delta}_j \tag{4.6}$$

In order to implement the optimal combination, the feedback gain matrices $R$ and $G$ need to be calculated. To do this, start with the appropriately modified versions of Eqs. (4.1), (4.2), and (4.5) and equate the appropriate terms

$$\underline{u}_j = Q\underline{Y}^* + \underline{v}_j = R\underline{u}_j + G\underline{e}_j + \underline{v}_j \tag{4.7}$$

Next, the appropriate definitions should be inserted into Eq. (4.7) for the overall current cycle plant input $\underline{u}_j$ and current cycle error $\underline{e}_j$ as shown

$$\begin{aligned} Q\underline{Y}^* + \underline{v}_j &= R\left(Q\underline{Y}^* + \underline{v}_j\right) + G\left(\underline{y}^* - \underline{y}_j\right) + \underline{v}_j \\ Q\underline{Y}^* &= R\left(Q\underline{Y}^* + \underline{v}_j\right) + G\left[\underline{y}^* - \left(P\underline{u}_j + \underline{\Delta}_j\right)\right] \\ &= RQ\underline{Y}^* + R\underline{v}_j + G\underline{y}^* - GP\left(Q\underline{Y}^* + \underline{v}_j\right) - G\underline{\Delta}_j \\ &= RQ\underline{Y}^* + R\underline{v}_j + G\underline{y}^* - GPQ\underline{Y}^* - GP\underline{v}_j - G\underline{\Delta}_j \end{aligned} \tag{4.8}$$

Now, since it was assumed that the current cycle disturbance $\underline{\Delta}_j$ is known exactly, the definition for $\underline{Y}^*$, given as part of Eq. (4.6), can be applied

$$
\begin{aligned}
Q\left(\underline{y}^* - \underline{\Delta}_j\right) &= RQ\left(\underline{y}^* - \underline{\Delta}_j\right) + R\underline{v}_j + G\underline{y}^* - GPQ\left(\underline{y}^* - \underline{\Delta}_j\right) - GP\underline{v}_j - G\underline{\Delta}_j \\
Q\underline{y}^* &= RQ\underline{y}^* + R\underline{v}_j + G\underline{y}^* - GPQ\underline{y}^* - GP\underline{v}_j + Q\underline{\Delta}_j - RQ\underline{\Delta}_j + GPQ\underline{\Delta}_j - G\underline{\Delta}_j \\
&= R\left(Q\underline{y}^* + \underline{v}_j\right) + G\left[(I - PQ)\underline{y}^* - P\underline{v}_j\right] + (I - R + GP)KG\underline{\Delta}_j - G\underline{\Delta}_j \\
&= IR\left(Q\underline{y}^* + \underline{v}_j\right) + IG\left[(I - PQ)\underline{y}^* - P\underline{v}_j\right] + G\underline{\Delta}_j - G\underline{\Delta}_j \\
&= IR\left(Q\underline{y}^* + \underline{v}_j\right) + IG\left[(I - PQ)\underline{y}^* - P\underline{v}_j\right]
\end{aligned}
\tag{4.9}
$$

It should be clarified that the use of the unidirectional definition given Eq. (4.3) in the derivation of Eq. (4.9) is valid since $Q$ is being defined as only a function of $K$ and $G$, or more fundamentally $R$ and $G$, and not the other way around. It is also worthwhile to note that the current cycle disturbance cancels out in Eq. (4.9) if, and only if, the condition given in Eq. (4.6) is satisfied. Recall, however, that Eq. (4.6) assumes the current cycle disturbance is known exactly, which is not a practical assumption for most applications. Typically, $\underline{\Delta}_j$ is treated as an unknown that must be estimated or compensated for by the controller design. The $Q$ matrix based finite-time FBC and ILC design is well-equipped to handle such unknowns, as discussed later in this sub-section. It is sufficient, for now, to simply accept that the given optimal combination of $Q$ and $\underline{v}_j$ was appropriately designed. Given such a controller, it is possible to replace $\underline{Y}^*$ with $\underline{y}^*$ in Eq. (4.7) and go through a similar derivation to arrive at the final result of Eq. (4.9), which does not involve the unknown current cycle disturbance. This is the most general approach, so Eq. (4.7) is redefined accordingly

$$\underline{u}_j = Q\underline{y}^* + \underline{v}_j = R\underline{u}_j + G\underline{e}_j + \underline{v}_j \tag{4.10}$$

Now, the same final result given in Eq. (4.9) can be obtained by going through similar steps, but starting with Eq. (4.10) instead of Eq. (4.8) and neglecting $\underline{\Delta}_j$ entirely from the derivation. This is acceptable because even though the necessary condition in Eq. (4.6) for designing $Q$ involved theoretical vector $\underline{Y}^*$, a more practical condition can be used in its place to design $Q$ in an analogous fashion. This alternative constraint could use an argument similar to $\underline{Y}^*$, but containing only known information about the disturbance environment. In order to obtain the desired feedback gain matrices from the final result of Eq. (4.9), it is necessary to permute the equation using the stack operator and the Kronecker product as shown

$$\begin{gathered}\left[Q\underline{y}^*\right]^S = \left[\left(Q\underline{y}^* + \underline{v}_j\right)^T \otimes I\right] R^S + \left[\left[(I - PQ)\underline{y}^* - P\underline{v}_j\right]^T \otimes I\right] G^S = A_R R^S + A_G G^S \\ [A_R \quad A_G]\begin{bmatrix} R^S \\ G^S \end{bmatrix} = \left[Q\underline{y}^*\right]^S\end{gathered} \tag{4.11}$$

This process is analogous to the finite-time FBC only conversion, with the major difference being that the learning signal must now be incorporated into this derivation. Once the feedback gains are determined in the stacked vector form given in Eq. (4.11), they must be transformed into gain matrices in a similar fashion as done for the $Q$ matrix. Here, it is again up to the designer to determine if time varying gains are desired or traditional time-invariant feedback gains are desired. As with the $Q$ matrix, it is generally recommended that full freedom be given, meaning all available gains should be used in either implementation. Once this is decided, both the $Q$ matrix and associated $R$ and $G$ feedback gain matrices should be generated accordingly. In either case, Eq. (4.11) will allow one to obtain the desired feedback gain matrices from given $Q$ and $\underline{v}_j$, even for

more than one desired signal. If a family of signals must be followed or disturbances spanning a particular frequency band must be rejected, the $Q$ matrix based design facilitates this objective by simply replacing the singular desired signal vector $\underline{y}^*$ with a training matrix $Y^*$, as appropriate. It should be noted that the training matrix $Y^*$ is not the same as the theoretical vector $\underline{Y}^*$ specified in Eq. (4.6), yet they both serve the same essential purpose. It is through the appropriate use of the training matrix $Y^*$, in a similar fashion to $\underline{Y}^*$, that the current cycle disturbance can be neglected. This is done by imposing the following condition

$$PQY^* \approx Y^* \tag{4.12}$$

The better the training matrix is able to estimate the disturbance environment, the better the $Q$ matrix can compensate for the unknown disturbances and the more accurate the overall conversion process becomes. Exactly how this can be done will be discussed in detail in a subsequent section. In learning control applications, Eq. (4.11) is recommended for the conversion of $R$ and $G$ from $Q$ and $\underline{v}_j$, along with the appropriately designed training matrix. It is reiterated that for this conversion, Eq. (4.11) supplants Eq. (4.3) because the latter is generally a unidirectional definition when a learning signal is present. Therefore, the use of Eq. (4.3) to find $R$ and $G$ directly from $Q$ is generally invalid if the desired linearization effects of using the $Q$ matrix are to be maintained. Next, the primary objective of how to obtain the necessary optimal combination of $Q$ matrix and learning signal $\underline{v}_j$ will be discussed.

### 4.1.2: Optimal Static Q Based Finite-Time FBC and ILC Design

The finite-time FBC and ILC combination controller implementation considered in Eq. (4.5) assumes that the $Q$ matrix is static in the repetition-domain. If the FBC is static, that means that only the ILC would exhibit repetition dynamics

$$\delta_j \underline{u} = \delta_j \underline{v} \tag{4.13}$$

Based on Eq. (4.13), the following cost function is proposed to optimize the finite-time feedback and learning combination controller

$$J_j = \left[\underline{e}_j\right]^T Q_e \underline{e}_j + \left[\delta_j \underline{u}\right]^T Q_u \delta_j \underline{u} + [Y^* - PQY^*]^{ST} Q_{q1} [Y^* - PQY^*]^S + \left[IQ\underline{y}^*\right]^T Q_{q2} IQ\underline{y}^* \tag{4.14}$$

This cost function penalizes the current cycle error, iterative updates to the overall plant input, the matching criteria of the static inverse matrix $Q$ based on Eq. (4.12), and the feedback control effort. Here, notice that the aforementioned training matrix $Y^*$ is used for designing $Q$ versus a single desired trajectory. In order to optimize the cost function presented in Eq. (4.14), it must first be modified into the correct format that explicitly shows the implicit relationships between the control variables to be optimized. In this case, the control variables to be optimized are the $Q$ matrix and iterative updates to the learning signal $\delta_j \underline{v}$, and so once again the stack operator and the Kronecker product are utilized

$$\begin{aligned} J_j = \left[\underline{e}_j\right]^T Q_e \underline{e}_j + \left[\delta_j \underline{u}\right]^T Q_u \delta_j \underline{u} + \left[Y^{*S} - \left(Y^{*T} \otimes P\right)\underline{q}\right]^T Q_{q1} \left[Y^{*S} - \left(Y^{*T} \otimes P\right)\underline{q}\right] \\ + \left[\left(\underline{y}^{*T} \otimes I\right)\underline{q}\right]^T Q_{q2} \left(\underline{y}^{*T} \otimes I\right)\underline{q} \end{aligned} \tag{4.15}$$

Notice that one of the two Kronecker products is the same as in Eq. (4.5), the other can be similarly represented with a static matrix, and $\underline{e}_j$ can be replaced using $\delta_j \underline{e} = \underline{e}_j - \underline{e}_{j-1}$ to get

$$J_j = \left[\underline{e}_{j-1} + \delta_j \underline{e}\right]^T Q_e \left[\underline{e}_{j-1} + \delta_j \underline{e}\right] + \left[\delta_j \underline{u}\right]^T Q_u \delta_j \underline{u} + \left[Y^{*S} - A_2 \underline{q}\right]^T Q_{q1} \left[Y^{*S} - A_2 \underline{q}\right] + \left[A_1 \underline{q}\right]^T Q_{q2} A_1 \underline{q} \tag{4.16}$$

Now, Eqs. (1.7) and (4.13) are used to explicitly show the underlying relations to the control variables of interest

$$J_j = \left[\underline{e}_{j-1} - P\delta_j \underline{v} - \delta_j \underline{d}\right]^T Q_e \left[\underline{e}_{j-1} - P\delta_j \underline{v} - \delta_j \underline{d}\right] + \left[\delta_j \underline{v}\right]^T Q_u \delta_j \underline{v} + \left[Y^{*S} - A_2 \underline{q}\right]^T Q_{q1} \left[Y^{*S} - A_2 \underline{q}\right] + \left[A_1 \underline{q}\right]^T Q_{q2} A_1 \underline{q} \tag{4.17}$$

The cost function given in Eq. (4.17) can now be optimized by taking the partial derivative with respect to $\underline{q}$ and $\delta_j \underline{v}$ and setting to zero as follows

$$\partial J_j \Big/ \partial \underline{q} = 2\left(A_2{}^T Q_{q1} A_2 + A_1{}^T Q_{q2} A_1\right)\underline{q} - 2A_2{}^T Q_{q1} Y^{*S} = 0 \tag{4.18}$$

$$\partial J_j \Big/ \partial \delta_j \underline{v} = 2(P^T Q_e P + Q_u)\delta_j \underline{v} - 2P^T Q_e \underline{e}_{j-1} + 2P^T Q_e \delta_j \underline{d} = 0 \tag{4.19}$$

Finding the optimal combination of finite-time FBC and ILC using a static $Q$ matrix in this manner leads to a naturally uncoupled solution that enables the independent design of the feedback and learning controllers. This suggests that the problem of finding an optimal combination of iteration invariant feedback and learning control can be viewed as a mutually exclusive problem that can first be solved separately and then implemented together in an optimal fashion. Here the independent designs are optimal for the cost function given in Eq. (4.17) if

$$\underline{q} = \left(A_2{}^T Q_{q1} A_2 + A_1{}^T Q_{q2} A_1\right)^{-1} A_2{}^T Q_{q1} Y^{*S} \tag{4.20}$$

$$\delta_j \underline{v} = (P^T Q_e P + Q_u)^{-1} P^T Q_e \left(\underline{e}_{j-1} - \delta_j \underline{d}\right) \tag{4.21}$$

It should be noted that Eq. (4.20) can generally be ill-conditioned and it is often necessary to condition the matrix summation to be inverted before solving for the $Q$ matrix. Popular conditioning techniques like truncation of bad singular values or Tikhonov regularization can be used to achieve more desirable results. Also, since $\delta_j \underline{d}$ is generally unknown and difficult, if not impossible, to accurately model and predict, it can be neglected for the learning signal update

$$\begin{aligned} \delta_j \underline{v} &= (P^T Q_e P + Q_u)^{-1} P^T Q_e \underline{e}_{j-1} \\ &= L^* \underline{e}_{j-1} \end{aligned} \tag{4.22}$$

Here, $L^*$ is introduced purely for mathematical convenience to represent $(P^T Q_e P + Q_u)^{-1} P^T Q_e$ in numerous subsequent derivations. If an accurate model or estimate of $\delta_j \underline{d}$ is available, Eq. (4.21) can be used with the appropriate replacement of the unknown $\delta_j \underline{d}$ variable as desired. However, Eq. (4.22) is generally the more applicable rule to be used for updating the learning signal.

### 4.1.3: Error Propagation of Optimal Static Q Based Finite-Time FBC and ILC Design

In order to predict the error performance of the static $Q$ matrix based feedback and learning combination controller, start with Eq. (4.10) and apply the difference operator $\delta_j$ as shown

$$\delta_j \underline{u} = R \delta_j \underline{u} + G \delta_j \underline{e} + \delta_j \underline{v} \tag{4.23}$$

Now, using Eq. (1.7), it is possible to rewrite Eq. (4.23) by substituting in for the unknown iterative changes to the error, which are to be determined

$$\delta_j \underline{u} = R \delta_j \underline{u} - G\left(P \delta_j \underline{u} + \delta_j \underline{d}\right) + \delta_j \underline{v} \tag{4.24}$$

After some standard arithmetic to collect like terms and simplify the equation, it is apparent that Eq. (4.24) can be rewritten as

$$\delta_j \underline{u} = (I - R + GP)^{-1}\delta_j \underline{v} - (I - R + GP)^{-1} G \delta_j \underline{d} \tag{4.25}$$

Notice that the definitions established in Eq. (4.3) for $K$ and $Q$ naturally appear in Eq. (4.25), so applying the appropriate substitution produces the following equation

$$\delta_j \underline{u} = K\delta_j \underline{v} - Q\delta_j \underline{d} \tag{4.26}$$

Next, by using Eqs. (1.6) and (1.8), or Eq. (1.7) directly, along with Eq. (4.26), the error propagation for this optimization is found in terms of $\underline{e}_{j-1}$, $\delta_j \underline{v}$, and $\delta_j \underline{d}$ as seen

$$\underline{e}_j = \underline{e}_{j-1} - P\left(K\delta_j \underline{v} - Q\delta_j \underline{d}\right) - \delta_j \underline{d} \tag{4.27}$$

To get it into its final form, one more substitution must be made with Eq. (4.22) and this leads to the result

$$\underline{e}_j = \underline{e}_{j-1} - P\left(KL^* \underline{e}_{j-1} - Q\delta_j \underline{d}\right) - \delta_j \underline{d} \tag{4.28}$$

Some additional standard arithmetic to collect like terms and simplify the equation further gives the final expression for error performance of the static $Q$ matrix based feedback and learning combination controller

$$\underline{e}_j = (I - PKL^*)\underline{e}_{j-1} - (I - PQ)\delta_j \underline{d} \tag{4.29}$$

Here, it should be evident that the error performance will be limited by the magnitude of the randomness from iteration to iteration, even if the plant is fully controllable and observable. However, through the use of a properly designed static $Q$ matrix based finite-time feedback, substantial improvement can be achieved, particularly if the plant is well-conditioned. Consequently, if no feedback was incorporated, which can be evaluated by setting $Q$ as well as the corresponding feedback gain matrices to zero and $K$ to the appropriately dimensioned identity matrix, it can be seen from Eq. (4.29) that there is no way to mitigate the performance limitations caused by random disturbances.

### 4.1.4: Optimal Dynamic Q Based Finite-Time FBC and ILC Design

Now consider a finite-time FBC and ILC combination controller implementation in which the feedback could also be updated in response to interactions with its operating environment in the same manner as the learning control. This can be done with a slight modification to the original combination controller proposed in Eq. (4.5) as shown

$$\begin{aligned} \underline{u}_j &= Q_j \underline{y}^* + \underline{v}_j \\ &= \left(\underline{y}^{*T} \otimes I\right) Q_j{}^S + \underline{v}_j \\ &= A_1 \underline{q}_j + \underline{v}_j \end{aligned} \tag{4.30}$$

Here the $Q_j$ matrix is now dynamic in the repetition-domain. Therefore, the corresponding updates to the plant input would now involve both FBC and ILC repetition dynamics

$$\delta_j \underline{u} = A_1 \delta_j \underline{q} + \delta_j \underline{v} \tag{4.31}$$

Based on Eq. (4.31), the following cost function is proposed to optimize the finite-time feedback and learning combination controller

$$\begin{aligned} J_j = \left[\underline{e}_j\right]^T Q_e \underline{e}_j + \left[\delta_j \underline{u}\right]^T Q_u \delta_j \underline{u} + \left[Y_j^* - PQ_j Y_j^*\right]^{ST} Q_{q1j} \left[Y_j^* - PQ_j Y_j^*\right]^S + \left[\delta_j \underline{q}\right]^T Q_q \delta_j \underline{q} \\ + \left[IQ_j \underline{y}^*\right]^T Q_{q2} IQ_j \underline{y}^* \end{aligned} \tag{4.32}$$

This cost function is similar to Eq. (4.14) in that it also penalizes the current cycle error, iterative updates to the overall plant input, the matching criteria of the inverse matrix $Q_j$ based on the dynamic version of Eq. (4.12), and the feedback control effort. However, unlike that cost function, Eq. (4.32) also penalizes iterative updates to the gains of the dynamic inverse matrix, which should be kept minimal in magnitude unless the overall design dictates otherwise. Notice that Eq. (4.32) allows the training matrix $Y_j^*$ to be

updated in an iterative fashion, which gives the designer full control over how updates to the $Q_j$ matrix are made. Following similar steps, as was done for the static $Q$ matrix design, this cost function is modified into a more suitable form for optimization

$$\begin{aligned} J_j &= \left[\underline{e}_j\right]^T Q_e \underline{e}_j + \left[\delta_j \underline{u}\right]^T Q_u \delta_j \underline{u} + \left[Y_j^{*S} - \left(Y_j^{*T} \otimes P\right)\underline{q}_j\right]^T Q_{q1} \left[Y_j^{*S} - \left(Y_j^{*T} \otimes P\right)\underline{q}_j\right] \\ &\quad + \left[\delta_j \underline{q}\right]^T Q_q \delta_j \underline{q} + \left[\left(\underline{y}^{*T} \otimes I\right)\underline{q}_j\right]^T Q_{q2} \left(\underline{y}^{*T} \otimes I\right)\underline{q}_j \\ &= \left[\underline{e}_{j-1} + \delta_j \underline{e}\right]^T Q_e \left[\underline{e}_{j-1} + \delta_j \underline{e}\right] + \left[\delta_j \underline{u}\right]^T Q_u \delta_j \underline{u} \\ &\quad + \left[Y_j^{*S} - A_{2j}\underline{q}_j\right]^T Q_{q1j} \left[Y_j^{*S} - A_{2j}\underline{q}_j\right] + \left[\delta_j \underline{q}\right]^T Q_q \delta_j \underline{q} \\ &\quad + \left[A_1 \underline{q}_j\right]^T Q_{q2} A_1 \underline{q}_j \end{aligned} \tag{4.33}$$

Here the control variables to be optimized will be $\delta_j \underline{q}$ and $\delta_j \underline{v}$, so the proper substitutions are made using Eqs. (1.7), (4.31), and the difference operator $\delta_j$ where appropriate

$$\begin{aligned} J_j &= \left[\underline{e}_{j-1} - P\left(A_1 \delta_j \underline{q} + \delta_j \underline{v}\right) - \delta_j \underline{d}\right]^T Q_e \left[\underline{e}_{j-1} - P\left(A_1 \delta_j \underline{q} + \delta_j \underline{v}\right) - \delta_j \underline{d}\right] \\ &\quad + \left[A_1 \delta_j \underline{q} + \delta_j \underline{v}\right]^T Q_u \left[A_1 \delta_j \underline{q} + \delta_j \underline{v}\right] \\ &\quad + \left[Y_j^{*S} - A_{2j}\left(\delta_j \underline{q} + \underline{q}_{j-1}\right)\right]^T Q_{q1j} \left[Y_j^{*S} - A_{2j}\left(\delta_j \underline{q} + \underline{q}_{j-1}\right)\right] \\ &\quad + \left[\delta_j \underline{q}\right]^T Q_q \delta_j \underline{q} + \left[A_1\left(\delta_j \underline{q} + \underline{q}_{j-1}\right)\right]^T Q_{q2} A_1 \left(\delta_j \underline{q} + \underline{q}_{j-1}\right) \end{aligned} \tag{4.34}$$

As done before, the cost function given in Eq. (4.34) can now be optimized by taking the partial derivative with respect to $\delta_j \underline{q}$ and $\delta_j \underline{v}$ and setting to zero as follows

$$\begin{aligned} {\partial J_j}\big/{\partial \delta_j \underline{q}} &= 2\left(A_1{}^T Q_{eu} A_1 + Q_{q12j} + Q_q\right)\delta_j \underline{q} + 2[A_1]^T Q_{eu} \delta_j \underline{v} - 2[PA_1]^T Q_e \underline{e}_{j-1} \\ &\quad + 2[PA_1]^T Q_e \delta_j \underline{d} + 2Q_{q12j}\underline{q}_{j-1} - 2A_{2j}{}^T Q_{q1j} Y_j^{*S} = 0 \end{aligned} \tag{4.35}$$

$$\partial J_j \Big/ \partial \delta_j \underline{v} = 2Q_{eu}\left[\delta_j \underline{v}\right] + 2Q_{eu}A_1\delta_j \underline{q} - 2P^T Q_e \underline{e}_{j-1} + 2P^T Q_e \delta_j \underline{d} = 0 \tag{4.36}$$

Here, $Q_{q12j} = {A_{2j}}^T Q_{q1} A_{2j} + {A_1}^T Q_{q2} A_1$ and $Q_{eu} = P^T Q_e P + Q_u$ and these two optimization equations can be conveniently solved simultaneously

$$\begin{bmatrix} \left({A_1}^T Q_{eu} A_1 + Q_{q12j} + Q_q\right) & {A_1}^T Q_{eu} \\ Q_{eu}A_1 & Q_{eu} \end{bmatrix} \begin{bmatrix} \delta_j \underline{q} \\ \delta_j \underline{v} \end{bmatrix}$$

$$= \begin{bmatrix} -Q_{q12j} & [PA_1]^T Q_e \\ 0 & P^T Q_e \end{bmatrix} \begin{bmatrix} \underline{q}_{j-1} \\ \underline{e}_{j-1} \end{bmatrix} + \begin{bmatrix} -[PA_1]^T Q_e & {A_{2j}}^T Q_{q1j} \\ -P^T Q_e & 0 \end{bmatrix} \begin{bmatrix} \delta_j \underline{d} \\ Y_j^{*S} \end{bmatrix} \tag{4.37}$$

$$A_j \begin{bmatrix} \delta_j \underline{q} \\ \delta_j \underline{v} \end{bmatrix} = B_j \begin{bmatrix} \underline{q}_{j-1} \\ \underline{e}_{j-1} \end{bmatrix} + C_j \begin{bmatrix} \delta_j \underline{d} \\ Y_j^{*S} \end{bmatrix}$$

It should be noted that unlike Eq. (4.20), Eq. (4.37) is generally well-conditioned, except when $Q_q$ is very small, and conditioning techniques are often not necessary. However, in cases where it is necessary to condition the $A_j$ matrix before finding the desired optimal updates, the same conditioning techniques used for Eq. (4.20) are again recommended here.

The optimal updates $\delta_j \underline{q}$ and $\delta_j \underline{v}$ can be obtained by solving Eq. (4.37) as given above or by determining the update laws directly through algebraic manipulation. If Eq. (4.35) is rearranged and the appropriate substitution is made using Eq. (4.36), it can be shown that

$$\begin{aligned} &{A_1}^T\left[P^T Q_e\left(\underline{e}_{j-1} - \delta_j \underline{d}\right)\right] + \left(Q_{q12j} + Q_q\right)\delta_j \underline{q} \\ &\qquad = -Q_{q12j}\underline{q}_{j-1} + [PA_1]^T Q_e\left(\underline{e}_{j-1} - \delta_j \underline{d}\right) + {A_2}^T Q_{q1j} Y_j^{*S} \end{aligned} \tag{4.38}$$

Cancelling like terms, applying the desired conditioning if necessary in Eq. (4.38), and then taking the appropriate inverses will produce the optimal dynamic $Q_j$ matrix based combination controller update laws

$$\delta_j \underline{q} = \left({A_{2j}}^T Q_{q1} A_{2j} + {A_1}^T Q_{q2} A_1 + Q_q\right)^{-1} {A_{2j}}^T Q_{q1j} {Y_j^*}^S - \left({A_{2j}}^T Q_{q1} A_{2j} + {A_1}^T Q_{q2} A_1 + Q_q\right)^{-1} \left({A_{2j}}^T Q_{q1} A_{2j} + {A_1}^T Q_{q2} A_1\right) \underline{q}_{j-1} \tag{4.39}$$

$$\delta_j \underline{v} = (P^T Q_e P + Q_u)^{-1} P^T Q_e \left(\underline{e}_{j-1} - \delta_j \underline{d}\right) - A_1 \delta_j \underline{q} \tag{4.40}$$

Again, applying the same definition for $L^*$ used in Eq. (4.22) and following the same reasoning for neglecting $\delta_j \underline{d}$ established there, the general learning signal update becomes

$$\delta_j \underline{v} = L^* \underline{e}_{j-1} - A_1 \delta_j \underline{q} \tag{4.41}$$

By comparing Eqs. (4.39) and (4.41) with the analogous optimal static $Q$ matrix update laws given in Eqs. (4.20) and (4.22), it should be clear that the primary design agent for both controllers is $Y^*$, the training matrix. The two main differences between the static $Q$ matrix and dynamic $Q_j$ matrix based designs are that for the dynamic controller, the optimization equations are generally well-conditioned and the training matrix is allowed to vary in the repetition-domain. A well-conditioned solution is always preferred and does not require additional explanation. While the advantages of having a variable training matrix are less obvious, the main improvement is that the design can now be truly optimal both in-repetition and in-run, or in time over the course of each individual run. The static controller, on the other hand, is truly optimal only in repetition, and can be considered suboptimal in-run when compared to the dynamic design. This is due to the general nature of both the system and operating environment which can be considered dynamic when the iteration number $j$ becomes large. There are a myriad of practical reasons for this, such as fluctuations in the daily operational temperature and even the natural heating, cooling, and eventual deterioration of system hardware. While the dynamic design can optimally adapt to such changes with both its feedback and learning

control actions, the static design can only compensate for these changes in repetition through learning control actions alone.

### 4.1.5: Error Propagation of Optimal Dynamic Q Based Finite-Time FBC and ILC Design

In order to predict the error performance of the dynamic $Q_j$ matrix based feedback and learning combination controller, start with the general, dynamic equivalent of Eq. (4.10) as shown

$$\underline{u}_j = Q_j \underline{y}^* + \underline{v}_j = R_j \underline{u}_j + G_j \underline{e}_j + \underline{v}_j \tag{4.42}$$

Next, make the appropriate substitutions using Eq. (1.3) and the definition $\underline{e}_j = \underline{y}^* - \underline{y}_j$ for error

$$\underline{u}_j = R_j \underline{u}_j + G_j \left[ \underline{y}^* - \left( P \underline{u}_j + \underline{\Delta}_j \right) \right] + \underline{v}_j \tag{4.43}$$

Expanding out the substitutions and collecting like terms will result in the following expression

$$\left( I - R_j + G_j P \right) \underline{u}_j = G_j \left( \underline{y}^* - \underline{\Delta}_j \right) + \underline{v}_j \tag{4.44}$$

Now, solving for the overall current cycle plant input and applying the general definition given in Eq. (4.3) produces the result

$$\begin{aligned} \underline{u}_j &= \left( I - R_j + G_j P \right)^{-1} \underline{v}_j + \left( I - R_j + G_j P \right)^{-1} G_j \left( \underline{y}^* - \underline{\Delta}_j \right) \\ &= K_j \underline{v}_j + Q_j \left( \underline{y}^* - \underline{\Delta}_j \right) \end{aligned} \tag{4.45}$$

Here, $K_j$ and $Q_j$ are the dynamic equivalents to the definitions given in Eq. (4.3) and, as observed in Eq. (4.25), these terms also naturally appear in Eq. (4.45) as seen above. The next steps are to permute $Q_j \underline{y}^*$ as done before and apply the difference operator $\delta_j$ appropriately as shown

$$\left(\delta_j \underline{u} + \underline{u}_{j-1}\right) = K_j\left(\delta_j \underline{v} + \underline{v}_{j-1}\right) + A_1 \underline{q}_j - Q_j \underline{\Delta}_j \tag{4.46}$$

Some basic steps involving algebraic manipulation to expand and rearrange terms, along with further applications of the difference operator where appropriate, results in the following

$$\begin{aligned}
\delta_j \underline{u} &= -\underline{u}_{j-1} + K_j \delta_j \underline{v} + K_j \underline{v}_{j-1} + A_1 \underline{q}_j - Q_j \underline{\Delta}_j \\
&= -\underline{u}_{j-1} + K_j \delta_j \underline{v} + \left(\delta_j K + K_{j-1}\right)\underline{v}_{j-1} + A_1 \underline{q}_j - Q_j\left(\delta_j \underline{d} + \underline{\Delta}_{j-1}\right) \\
&= K_j \delta_j \underline{v} + \delta_j K \underline{v}_{j-1} - \left(\underline{u}_{j-1} - K_{j-1}\underline{v}_{j-1}\right) + A_1 \underline{q}_j - Q_j \delta_j \underline{d} - Q_j \underline{\Delta}_{j-1}
\end{aligned} \tag{4.47}$$

Using the corresponding iteration of Eq. (4.45) allows for a necessary substitution in the derivation and subsequent algebraic steps, similar to those above, will lead to another desired expression

$$\begin{aligned}
\delta_j \underline{u} &= K_j \delta_j \underline{v} + \delta_j K \underline{v}_{j-1} - \left[Q_{j-1}\left(\underline{y}^* - \underline{\Delta}_{j-1}\right)\right] + A_1 \underline{q}_j - Q_j \delta_j \underline{d} - Q_j \underline{\Delta}_{j-1} \\
&= K_j \delta_j \underline{v} + \delta_j K \underline{v}_{j-1} - Q_{j-1}\underline{y}^* + Q_{j-1}\underline{\Delta}_{j-1} + A_1 \underline{q}_j - Q_j \delta_j \underline{d} - Q_j \underline{\Delta}_{j-1} \\
&= K_j \delta_j \underline{v} + \delta_j K \underline{v}_{j-1} + A_1 \underline{q}_j - A_1 \underline{q}_{j-1} - Q_j \delta_j \underline{d} - \left(Q_j - Q_{j-1}\right)\underline{\Delta}_{j-1} \\
&= K_j \delta_j \underline{v} + \delta_j K \underline{v}_{j-1} + A_1 \delta_j \underline{q} - Q_j \delta_j \underline{d} - \delta_j Q \underline{\Delta}_{j-1}
\end{aligned} \tag{4.48}$$

Notice that the final line of Eq. (4.48) is the dynamic $Q_j$ matrix version of Eq. (4.26) and can similarly be used, along with Eq. (1.7), to determine the error propagation for this optimization

$$\begin{aligned}
\underline{e}_j &= \underline{e}_{j-1} - P\left(K_j \delta_j \underline{v} + \delta_j K \underline{v}_{j-1} + A_1 \delta_j \underline{q} - Q_j \delta_j \underline{d} - \delta_j Q \underline{\Delta}_{j-1}\right) - \delta_j \underline{d} \\
&= \underline{e}_{j-1} - P K_j \delta_j \underline{v} - P \delta_j K \underline{v}_{j-1} - P A_1 \delta_j \underline{q} + P Q_j \delta_j \underline{d} + P \delta_j Q \underline{\Delta}_{j-1} - \delta_j \underline{d}
\end{aligned} \tag{4.49}$$

In order to arrive at the ultimate expression for error performance of the dynamic $Q_j$ matrix based design, one more substitution must be made using Eq. (4.41) before the remaining algebraic steps can be performed

$$\underline{e}_j = \underline{e}_{j-1} - PK_j\left(L^*\underline{e}_{j-1} - A_1\delta_j\underline{q}\right) - PA_1\delta_j\underline{q} - \left(I - PQ_j\right)\delta_j\underline{d} \\ - P\left(\delta_j K\underline{v}_{j-1} - \delta_j Q\underline{\Delta}_{j-1}\right) \tag{4.50}$$

Finally, a few more simple steps to collect like terms and rearrange others produces the desired final expression for error propagation of the optimal dynamic $Q_j$ matrix based feedback and learning combination controller

$$\underline{e}_j = \left(I - PK_jL^*\right)\underline{e}_{j-1} - \left(I - PQ_j\right)\delta_j\underline{d} - P\left(I - K_j\right)A_1\delta_j\underline{q} - P\left(\delta_j K\underline{v}_{j-1} - \delta_j Q\underline{\Delta}_{j-1}\right) \tag{4.51}$$

Upon inspection of Eq. (4.51), it should be apparent why it was suggested in the previous section, when the cost function was first proposed, that iterative updates to the gains of the dynamic $Q_j$ matrix should be kept minimal in magnitude unless the overall design dictates otherwise. This can be precisely controlled using the weighting matrix $Q_q$ in Eq. (4.32) as desired. In general, if the magnitude of those iterative updates is allowed to be large, the terms $\delta_j\underline{q}$, $\delta_j K$, and $\delta_j Q$ can become large and potentially make the current cycle error worse. If the magnitude of the iterative updates is effectively minimized or sufficient iterations are allowed to pass such that the dynamic $Q_j$ matrix settles to an optimal in-repetition and in-run state, then those terms become negligible and the error propagation can be approximated with

$$\underline{e}_j \approx \left(I - PK_jL^*\right)\underline{e}_{j-1} - \left(I - PQ_j\right)\delta_j\underline{d} \tag{4.52}$$

A simple comparison to the error propagation of the static $Q$ matrix based feedback and learning combination controller given in Eq. (4.29) shows a strikingly similar expression. The advantage of the dynamic matrix design should become clear when considering Eq. (4.52), since $K_j$ and $Q_j$ can now be optimized each iteration based on recent interactions with its operating environment. This means that the dynamic $Q_j$ matrix based feedback

and learning combination controller has the potential to be optimal both in-repetition and in-run, unlike the static $Q$ matrix based design.

## 4.2: Optimal Q Based Finite-Time FBC and ILC Design Considerations

The previous section established two different design approaches for optimal $Q$ matrix based finite-time FBC and ILC combination controllers. These approaches are developed sufficiently for direct implementation. However, in order to most efficiently harness the tremendous potential of these approaches, some important design considerations should be discussed in detail. Significant general information about how to design the training matrix to meet various design objectives and how to use finite-time sensitivity transfer matrices as a general evaluation tool, analogous to conventional frequency response analysis, for the finite-time FBC only case was presented in a previous study.[69] Here, the focus is to leverage the understanding gained from that work and apply it directly to take full advantage of the inherent flexibility of the $Q$ matrix based design for learning control applications.

### 4.2.1: The Training Matrix for Learning Control Applications

The primary design agent for both the static and dynamic feedback and learning combination controllers is $Y^*$, as stated in the prior section. For simplicity, assume for now that the current cycle disturbance, expressed in Eq. (1.4), is bounded and known for every repetition. In this context, bounded implies that the overall magnitude of the disturbance is sufficiently small relative to the desired signal such that effective control is possible. If this assumption was valid, then a training matrix could, in theory, utilize that information to generate a finite-time FBC to suppress the disturbances each repetition and output the desired signal with nearly negligible error. Such a training matrix would take the form

$$Y^* = \left[\underline{y}^*, \underline{\Delta}_0, \underline{\Delta}_1, \cdots, \underline{\Delta}_{j-1}, \underline{\Delta}_j\right] \tag{4.53}$$

This theoretical training matrix could be used for either the static $Q$ or dynamic $Q_j$ matrix based design. However, if such a training matrix existed, then there would be no real advantage to using the dynamic design. Unfortunately, the current cycle disturbance and all future cycle disturbances, though can be assumed to be bounded, are generally unknown in practice.

It is still possible, however, to allow for a sufficient number of calibration trials in an attempt to acquire useful information about the disturbance environment to aid the design of the training matrix. For the static $Q$ matrix design, that calibration period could result in a training matrix of the form

$$Y^* = \left[\underline{y}^*, \underline{\hat{d}}, \sin\left(2\pi \underline{f}\,\underline{t}\right)\right] \tag{4.54}$$

This practically feasible training matrix is composed of the desired reference signal, the estimate $\underline{\hat{d}}$ of any repeating disturbance term $\underline{d}$ that is identified during calibration, and a sinusoidal disturbance rejection matrix. The sinusoidal disturbance rejection matrix is composed of appropriately selected frequencies to reject disturbances with content in the band surrounding those frequencies

$$\underline{f} = [f_1, f_2, f_3, \dots] \tag{4.55}$$

$$\underline{t} = kT \qquad \text{for } k = [0,1,2, \dots, p-1]^T \tag{4.56}$$

The number of sinusoidal disturbance rejection frequencies that should be used is design-specific and any variety may be considered, but the goal is to reject random disturbances over a desired band of frequencies. The calibration period will typically be integral to the overall performance in static $Q$ matrix designs. This is because the finite-time FBC is unable to adjust for current or future inaccuracies in the training matrix

without routine calibration, which is not always feasible. The dynamic $Q_j$ matrix based design is far less susceptible to calibration errors since it can be designed to iteratively adjust to maintain optimal performance in-repetition and in-run, with no need for routine calibration. The brute force way to do this is to use a training matrix of the form

$$Y^* = \left[\underline{y}^*, \underline{\hat{d}}, \delta_1\underline{d}, \delta_2\underline{d}, \cdots, \delta_{j-2}\underline{d}, \delta_{j-1}\underline{d}\right] \tag{4.57}$$

While this design is feasible and practically implementable, the computational demand will again be a significant limiting factor as the number of iterations grows. A more computationally efficient training matrix which can be used to achieve the expected optimal performance is

$$Y^* = \left[\underline{y}^*, \underline{\hat{d}}, \hat{\delta}_j\underline{d}, \sin\left(2\pi\underline{\hat{f}}\underline{t}\right)\right] \tag{4.58}$$

Here the estimate $\underline{\hat{d}}$ of the repeating disturbance $\underline{d}$, estimate $\hat{\delta}_j\underline{d}$ of the iterative change $\delta_j\underline{d}$ in the current cycle disturbance, and estimate $\underline{\hat{f}}$ to cover the random disturbance spectrum, can be obtained using any estimation technique, such as basic least squares or Kalman filters for example. The estimation process can utilize all of the previous repetition information in a numerically and computationally efficient manner to produce optimal results. The more accurately these estimates represent the operating environment, the better the overall controller will perform. However, even with initial inaccuracies in the training matrix, it should be reiterated that the dynamic $Q_j$ based design can still improve the overall performance.

### 4.2.2: Finite-Time Sensitivity Transfer Matrices for Q Matrix Based FBC and ILC Combinations

Before discussing the implementation of more conventional, steady state response based FBC designs through the $Q$ matrix approach, it is important to establish a method

for comparison. The use of finite-time sensitivity transfer matrices to support such a comparison, in addition to general evaluation, is now established for use in learning control applications. The primary focus here is sensitivity of the overall system to signals of varying frequency, in terms of impact to tracking error. This can be assessed using the error propagation equations derived for each design presented in Eqs. (4.29) and (4.52) previously. For both designs, the current cycle tracking error is a function of the previous cycle tracking error $\underline{e}_{j-1}$ and iterative change $\delta_j \underline{d}$ in the current cycle disturbance. If these two terms are thought of as inputs and the tracking error as the output, then it is possible to drive the tracking error output using a variable signal input that is frequency dependent and normalized to be a unit vector. For the purpose of error analysis, the norm of this frequency dependent tracking error output is of most interest. Following this approach, two general finite-time sensitivity transfer matrices are proposed

$$e_j(\omega) = \left\| \left(I - PK_jL^*\right)\left[\eta \sin\left(\omega \underline{t}\right)\right] \right\| = \left\| \left(I - PK_jL^*\right)\underline{s} \right\| \tag{4.59}$$

$$e_j(\omega) = \left\| \left(I - PQ_j\right)\left[\eta \sin\left(\omega \underline{t}\right)\right] \right\| = \left\| \left(I - PQ_j\right)\underline{s} \right\| \tag{4.60}$$

Here, Eq. (4.59) is the previous cycle error finite-time sensitivity transfer matrix and Eq. (4.60) is the disturbance difference finite-time sensitivity transfer matrix. The three common variables in Eqs. (4.59) and (4.60), the input transfer frequency $\omega$, input time vector $\underline{t}$, and normalizing scale factor $\eta$, must all be appropriately defined by the designer when driving these finite-time sensitivity transfer matrices. The following is some guidance on the proper way to define these variables, in an attempt to reliably and efficiently bridge the gap between the finite-time design approaches and more conventional control design approaches. The input time vector $\underline{t}$ is simply defined using

Eq. (4.56) above. The input transfer frequency $\omega$ should span the full operational spectrum, from zero to Nyquist frequency $f_N$, for the controller implementation and with sufficient granularity to ensure adequate frequency resolution. This can be accomplished by using a predefined set of input transfer frequencies $\omega_i = 2\pi f_i$, which are a function of the fundamental frequency resolution

$$f_0 = \frac{1}{2Tp} = \frac{f_s}{2p} = \frac{f_N}{p} \tag{4.61}$$

It should be evident that as the sampling frequency $f_s$ is increased, the total number of time steps $p$ would need to be proportionally increased to maintain $f_0$ and, consequently, the desired overall frequency resolution. The set of input transfer frequencies can now be defined as

$$\omega_i = 2\pi f_i = 2\pi[0, f_0, 2f_0, \dots, (p-1)f_0] \tag{4.62}$$

Notice here that the largest transfer frequency in the input set $\omega_i$ approaches $f_N$ as $p$ approaches infinity, but for finite-time applications it will always be less than the Nyquist frequency. Finally, the normalizing scale factor must be calculated for each frequency $\omega$ in the set of input transfer frequencies $\omega_i$, such that the norm of the driving vector is unity

$$\eta = \frac{1}{\left\|\sin(\omega \underline{t})\right\|} \tag{4.63}$$

When using Eq. (4.63), care must be taken to avoid a divide by zero. Some interesting conclusions can be made by recognizing that Eqs. (4.59) and (4.60) are subordinate matrix norms, so by definition

$$\left\|\left(I - PK_jL^*\right)\right\| = \sup\left\{\left\|\left(I - PK_jL^*\right)\underline{s}\right\| \;\middle|\; \underline{s} \in \mathbb{R}^p \;\text{ and }\; \left\|\underline{s}\right\| = 1\right\} \tag{4.64}$$

$$\left\|\left(I - PQ_j\right)\right\| = \sup\left\{\left\|\left(I - PQ_j\right)\underline{s}\right\| \;\middle|\; \underline{s} \in \mathbb{R}^p \;\text{ and }\; \left\|\underline{s}\right\| = 1\right\} \tag{4.65}$$

If the Euclidean norm is used for this error analysis, these definitions can be utilized to arrive at a very important result

$$\left\|\left(I - PK_jL^*\right)\underline{s}\right\|_2 \leq \left\|\left(I - PK_jL^*\right)\right\|_2 \left\|\underline{s}\right\|_2 = \max_{1\leq i\leq p} \sigma_i \tag{4.66}$$

$$\left\|\left(I - PQ_j\right)\underline{s}\right\|_2 \leq \left\|\left(I - PQ_j\right)\right\|_2 \left\|\underline{s}\right\|_2 = \max_{1\leq i\leq p} \sigma_i \tag{4.67}$$

The significance of Eqs. (4.66) and (4.67) is that they prove that the error gains shown by the previous cycle error and disturbance difference finite-time sensitivity transfer matrices will always be bounded by the largest singular value $\sigma$ of the respective transfer matrix. Furthermore, it turns out that for monotonically increasing finite-time sensitivity transfer matrices, the entire plot is bounded by a corresponding plot of monotonically increasing singular values. This observation suggests that the overall performance of the controller is generally bounded by the singular values of the transfer matrix. Therefore, singular value decomposition can be used as another very useful analysis tool for finite-time applications. It is, however, recommended that the sensitivity transfer matrix be utilized as the primary evaluation tool for finite-time design, just as frequency response is utilized for steady state design.

### 4.2.3: Implementation of Conventional FBC Using Q Matrix Based Designs

Consider the case now where the implementation of more conventional FBC using frequency response based techniques is desired. Even though such steady state frequency response based design approaches are not best suited for finite-time applications, like learning control, it is insightful to demonstrate that $Q$ matrix based implementations are flexible enough to accommodate this. Automatic FBC designs used in various industrial applications include P, PI, PD, and PID controllers, with PID being

the most common. Here, implementation is restricted to the PID because the same approach can easily be applied to other conventional FBC designs

$$u(t) = K_P\left[e(t) + \frac{1}{T_I}\int_0^t e(\tau)d\tau + T_D\frac{d}{dt}e(t)\right] \Leftrightarrow E(s)\left[K_P + K_I\frac{1}{s} + K_D s\right] \tag{4.68}$$

Before use, the controller parameters $K_P$, the proportional gain, $T_I$, the reset time, and $T_D$, the rate time, must be tuned. There are numerous industrially established approaches to tuning a PID. The most noteworthy approach may likely be a closed-loop method first proposed by Ziegler and Nichols in 1942, also referred to as the continuous cycling or ultimate gain tuning method.[100-101] This frequency response tuning approach was once considered ideal and has been used extensively to tune loops for over half a century. For current standards, it may no longer be considered ideal, but a PID designed using this method is a good basis for comparison with the novel finite-time FBC designs presented in this work. In order to replicate such a controller using the $Q$ matrix design, the corresponding $R$ and $G$ gain matrices must be found. To do this, the z-transform of Eq. (4.68) is derived using the bilinear transformation to approximate the integral action and the backward difference to approximate the derivative action

$$\begin{aligned} U(z) &= E(z)\left[K_P + K_I\frac{T}{2}\frac{z+1}{z-1} + K_D\frac{z-1}{Tz}\right] \\ &= E(z)\frac{\left(K_P + \frac{K_I T}{2} + \frac{K_D}{T}\right)z^2 + \left(\frac{K_I T}{2} - K_P - \frac{2K_D}{T}\right)z + \frac{K_D}{T}}{z(z-1)} \\ &= E(z)\frac{K_3 + K_2 z^{-1} + K_1 z^{-2}}{1 - z^{-1}} \end{aligned} \tag{4.69}$$

It should be clear that $K_I$ and $K_D$ are defined through the tuned parameters. The time-domain version of Eq. (4.69) can be obtained by taking the inverse z-transform

$$u(k) = u(k-1) + K_3 e(k) + K_2 e(k-1) + K_1 e(k-2) \tag{4.70}$$

In order to properly convert Eq. (4.70) to the desired $R$ and $G$ gain matrices, the appropriate time indexing for the input and error vectors must be maintained. Recall that practical digital controllers typically require a one-step delay from input to output and, since the error is a function of the output, that one-step delay will exist in the error vector. With this fact in mind, the FBC gain matrices for the PID controller expressed in Eq. (4.70) are defined as follows

$$R = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \ddots & \vdots \\ 0 & 1 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \qquad G = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & \cdots & 0 \\ K_3 & 0 & 0 & 0 & 0 & \ddots & \vdots \\ K_2 & K_3 & 0 & 0 & 0 & \ddots & \vdots \\ K_1 & K_2 & K_3 & 0 & 0 & \ddots & \vdots \\ 0 & K_1 & K_2 & K_3 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & K_1 & K_2 & K_3 & 0 \end{bmatrix} \tag{4.71}$$

Now, it is possible to generate the corresponding $Q$ matrix based finite-time FBC in two steps. First, $Q$ must be calculated using Eq. (4.3) and then, $R$ and $G$ must be recalculated using Eq. (4.11) prior to implementation. Since Eq. (4.3) defines a $Q$ matrix independent of a signal vector $\underline{y}^*$ or training matrix $Y^*$, either can be used in Eq. (4.11) without impacting the result. However, some care must be taken when recalculating $R$ and $G$ because sufficient freedom is necessary to ensure a reliable solution. For example, if the total number of unique gains used in the finite-time FBC matrices is limited to the same number of gains as the original PID, then the recalculated $R$ and $G$ will match the original matrices of Eq. (4.71) above for the first iteration, where the learning signal is zero. However, future finite-time FBC gain matrices may not be able to accurately reproduce the control actions predicted by the $Q$ matrix with this limited freedom because the learning signal will no longer be zero. So, it is generally

recommended that full freedom be given by using all available gains for either time varying or time-invariant control matrices. However, as stated earlier, use of time-invariant FBC will result in a lower-dimensional problem because the resulting gain matrices would be Toeplitz.

### 4.3: Numerical Simulation

Numerical simulations were conducted on MATLAB 7.7 to illustrate the effectiveness of the optimal static and dynamic $Q$ based finite-time feedback and learning controller. Three different types of system models, one well-conditioned, one moderately damped ill-conditioned, and one extremely lightly damped ill-conditioned spine, were utilized in an attempt to cover the full spectrum of practical problems a control designer may encounter. The well-conditioned system model effectively considers the scenario in which inversion of the plant does not result in unstable control actions and the desired trajectory is sufficiently long to allow most of the transient response to decay to zero. The step response of the well-conditioned plant is given in Figure 4-1, which shows that the transient response decays to zero after approximately 0.4 seconds. The first ill-conditioned system model, which will simply be referred to as the ill-conditioned model from here forth, effectively considers control problems where inversion of the plant results in unstable control actions and the reference trajectory is comparable to the settling time of the system. The step response of the ill-conditioned plant is given in Figure 4-2, which shows that the transient response decays to zero after approximately 0.8 seconds. The second ill-conditioned system model, which will simply be referred to as the spine model from here forth, was specifically chosen to demonstrate that $Q$ matrix based designs can reliably improve performance even for extremely lightly damped, multi-modal plants where conventional controls fail. The step response of the spine

model is given in Figure 4-3, which shows that the system has not completely settled even after 20 seconds. For all three models, causal and time-invariant finite-time FBC designs were utilized solely due to ease of implementation. Simulations are conducted using all three systems in the ideal case with no disturbances present, and then in more practical situations that involve a repeating disturbance, non-repeating disturbances, and both repeating and non-repeating disturbances present simultaneously. The random disturbance is simply bandlimited white noise, appropriately scaled such that effective control is possible. There are also simulations attempting to use conventional feedback designs for comparison purposes.

### 4.3.1: Implementation Using Well-Conditioned and Ill-Conditioned Plant

The first setup to be simulated is the most ideal case where the plant is well-conditioned and there is no disturbance affecting the control process. For this setup, a third order system model with the following transfer function is considered

$$Y(s) = G(s)U(s) = \left(\frac{s^2 + 26s + 170}{100}\right)\left(\frac{a}{s+a}\right)\left(\frac{{\omega_n}^2}{\mathrm{s}^2 + 2\,\zeta\ \omega_n s + {\omega_n}^2}\right)U(s) \qquad (4.72)$$

For the numerical simulations $a = 8.8$, $\omega_n = 37$, $\zeta = 0.5$, and the continuous time model was discretized using a ZOH, as first described in the learning formulation section of Chapter 1, using a 100 Hz sample rate. Notice that the continuous time transfer function has a pole excess of one. Therefore, no zeros are introduced by the discretization process that can potentially be outside the unit circle.

The ill-conditioned plant considered is also represented using a third order system model, but with a transfer function with a pole excess of three as follows

$$Y(s) = G(s)U(s) = \left(\frac{a}{s+a}\right)\left(\frac{{\omega_n}^2}{s^2 + 2\,\zeta\,\omega_n s + {\omega_n}^2}\right)U(s) \tag{4.73}$$

As before, for the numerical simulations $a = 8.8,\ \omega_n = 37,\ \zeta = 0.5,$ and the continuous time model was again discretized using a ZOH at a 100 Hz sample rate. When the pole excess is three or more and the sample time is fast enough, a zero is introduced outside the unit circle by the discretization process. This means that the inverse problem that is addressed by learning control, of finding the input needed to produce the desired output, is working with the inverse of a discrete transfer function which is unstable. Therefore, this plant is considered ill-conditioned in this context and inverse control is not possible. Furthermore, with an ill-conditioned plant, perfect tracking is not possible even in the ideal case where there is no disturbance present.

The training matrix used for the static $Q$ based finite-time FBC and ILC controller is defined using Eq. (4.54) with $\underline{f} = [2{,}4]$ to cover the expected random disturbance spectrum. For the dynamic $Q$ matrix based finite-time FBC and ILC controller, Eq. (4.58) is used to define the training matrix with $\underline{\hat{d}}$ and $\hat{\delta}_j\underline{d}$ estimated using a simple running average of the previous 10 iterations and $\underline{\hat{f}} = [2{,}4]$, similar to the static design. Clearly, no specialized estimation techniques are employed here and it can be assumed that more intricate methods to generate the training matrix would likely improve performance to some extent.

**4.3.2: Implementation Using OKID Specified Spine Model**

The spine model represents an experimental system consisting of a set of parallel steel rods held together at the centers by a thin steel wire. It was obtained using the Observer/Kalman Filter Identification (OKID) method, where a discrete-time state-space

model of the system was constructed using data from experiments.[38] As stated previously, the system is extremely lightly damped and once it is perturbed even slightly, it takes more than 30 seconds for the vibrations to completely subside. In this chapter, the very aggressive control objective of following a 0.5 second trajectory is pursued for the ideal case with no disturbances present and then for more practical situations with various realistic disturbances present. Though perfect tracking is not possible even in the ideal case, as with the ill-conditioned plant, the true advantages of optimal static and dynamic $Q$ based finite-time feedback and learning controllers are most clearly seen with numerical simulations using this spine model.

As with the well-conditioned and ill-conditioned plants, the spine model training matrix used for the static $Q$ based finite-time FBC and ILC controller is also defined following Eq. (4.54), but with additional frequencies, $\underline{f} = [2,4,6,8,10]$, to cover the expected random disturbance spectrum. For the dynamic $Q$ matrix based finite-time FBC and ILC controller, Eq. (4.58) is used to define the training matrix with $\underline{\hat{d}}$ and $\hat{\delta}_j\underline{d}$ estimated using a simple running average of the previous 10 iterations and $\underline{\hat{f}} = [2,4,6,8,10]$, again similar to the static design. As with the well-conditioned and ill-conditioned plants, no specialized estimation techniques are employed for the spine model either. Thus, there is potential for improved performance if more intricate estimation techniques are used.

### 4.3.3: Results

For the well-conditioned plant modeled by Eq. (4.72) and the ill-conditioned plant modeled by Eq. (4.73), numerical simulations were conducted to determine the effectiveness of the combination controllers at following a polynomial command signal based on

$$y^*(t) = \pi\left(5\left(\frac{t}{t_p/2}\right)^3 - 7.5\left(\frac{t}{t_p/2}\right)^4 + 3\left(\frac{t}{t_p/2}\right)^5\right) \tag{4.74}$$

A plot of the desired trajectory followed by those two plants is given in Figure 4-4 with a fundamental frequency of 2 Hz. The equation used to generate the first half of this command is Eq. (4.74) and the second half is merely a reflection of the first half. This trajectory is unique in that it has continuous position, velocity, and acceleration starting from rest and along the segment boundaries described above, which implies that it will induce relatively smooth operation. Also, the fundamental frequency of this command is set by specifying $t_p$, its period. For the OKID specified spine model, Figure 4-5 shows the desired 0.5 second trajectory. For the simulations involving repeating disturbances, Figure 4-6 shows the first repeating disturbance to be used with the well-conditioned and ill-conditioned plants and Figure 4-7 shows the second repeating disturbance to be used with the spine model. The first spectrum for bandlimited white noise random disturbances is presented in Figure 4-8 for the well-conditioned and ill-conditioned plants, while Figure 4-9 presents the second spectrum of random disturbances to be used with the spine model. Finally, for the most practically applicable disturbance environment, Figure 4-10 gives one example of the combined disturbance, presented in Eq. (1.4), which must be handled by the well-conditioned and ill-conditioned plants and Figure 4-11 gives the second example of the combined disturbance which must be handled by the spine model.

The first series of simulations conducted was for the well-conditioned plant. The results of these simulations for the ideal case with no disturbances are shown in Figures 4-12a and 4-13a for the static and dynamic $Q$ matrix based finite-time FBC and ILC

controllers, respectively. Figure 4-14a presents the static $Q$ matrix based conventional PI FBC and ILC controller, also for the ideal case. Here the PI controller can be specified using Eq. (4.71) with $K_3 = 1.15$, $K_2 = -0.7$, and $K_1 = 0$ to seed the $Q$ matrix using Eq. (4.3), before recalculating $R$ and $G$ using Eq. (4.11) with full freedom. Figures 4-12b, 4-13b, and 4-14b show how the controllers performed in the presence of a repeating disturbance, in the same order as the no disturbance case. The performance of these same controllers in the presence of random disturbances is depicted in Figures 4-12c, 4-13c, and 4-14c, again in the same order. Similarly, Figures 4-12d, 4-13d, and 4-14d provide the simulations in the presence of combined disturbances. Finally, the corresponding previous cycle error finite-time sensitivity transfer matrix is presented in Figures 4-12e, 4-13e, and 4-14e, followed by the disturbance difference finite-time sensitivity transfer matrix in Figures 4-12f, 4-13f, and 4-14f.

The second series of simulations were for the ill-conditioned plant. As before, the performances of the static and dynamic $Q$ matrix based finite-time FBC and ILC controllers and the static $Q$ matrix based conventional PID FBC and ILC controller are presented in Figures 4-15, 4-16, and 4-17, respectively. Here, the PID controller is tuned using the Ziegler and Nichols ultimate gain method and the gain matrices can be specified using Eq. (4.71) with $K_3 \approx 9.7$, $K_2 \approx -15.5$, and $K_1 \approx 6.2$ to initially seed the $Q$ matrix. This is again done using Eq. (4.3), before recalculating $R$ and $G$ using Eq. (4.11) to obtain full freedom. The four distinct disturbance environments used for the well-conditioned plant are again simulated in the same order and the previous cycle error and disturbance difference finite-time sensitivity transfer matrices are similarly given for the combined disturbance environment.

The final series of simulations involved the spine model and the results of those simulations, for the same types of controllers and disturbance environments, are given in Figures 4-18, 4-19, and 4-20, as before. The PID controller was again tuned using the Ziegler and Nichols ultimate gain method, resulting in $K_3 \approx 0.13$, $K_2 \approx -0.21$, and $K_1 \approx 0.09$ to initialize the $Q$ matrix. This PID controller is unable to effectively reject random disturbances, as illustrated by Figure 4-20f, which indicates that there is no attenuation of disturbances at any frequency. Even if the gain values are significantly increased, performance is not notably improved.

From inspection of Figures 4-13, 4-16, and 4-19, it can be seen that the approximation of Eq. (4.51) using Eq. (4.52) is valid if the magnitudes of the iterative updates present in Eq. (4.51) are effectively minimized or sufficient iterations are allowed to pass, such that the dynamic $Q_j$ matrix settles to an optimal in-repetition and in-run state. A general comparison of all these numerical simulations confirms that the most optimal design, which guarantees optimal performance in all situations, is the dynamic $Q$ matrix based finite-time FBC and ILC controller. It can be argued that the static $Q$ matrix based finite-time FBC and ILC controller provides optimal performance in the majority of simulations. In fact, it is reasonable to propose that the static $Q$ matrix based finite-time FBC and ILC design is the best performing controller for the well-conditioned plant when comparing Figures 4-12a and 4-12b to Figures 4-13a, 4-13b and 4-14a, 4-14b at first glance. However, it is important to remember that the ultimate indicators of performance are the final error, as the repetition number $j$ approaches infinity, and the finite-time sensitivity transfer matrices. If the performance of the controllers are compared based on those ultimate indicators, it can be seen that the final error, for all three controllers, in the no disturbance and repeating disturbance environment is zero. For the random

disturbance and combination disturbance environment, the static and dynamic $Q$ matrix based finite-time FBC and ILC controllers give equivalent performance and obtain the same final error, which is about an order of magnitude better when compared to the PI FBC implementation. This discrepancy in performance can be explained by comparing the finite-time sensitivity transfer matrices shown in Figures 4-12e, 4-13e, 4-14e with those in 4-12f, 4-13f, 4-14f, and looking at the attenuation over the random disturbance spectrum. Keep in mind that the waterbed effect in feedback control will limit the frequencies over which the controller is able to effectively reject disturbances. So, the goal is to have the feedback controller attenuate as much as possible over the frequencies of interest. Since the static and dynamic $Q$ matrix based finite-time FBC and ILC controllers attenuate disturbances over the random disturbance spectrum to a greater level than the PI FBC, they perform better in such a disturbance environment. However, the waterbed effect results in much greater amplification over the other frequencies, so care must be taken to ensure that disturbances in those frequencies do not enter the control loop. For the ill-conditioned plant and spine model, the dynamic $Q$ matrix based finite-time FBC and ILC controller is notably superior when compared to the other controllers and is truly optimal both in-repetition and in-run. For some disturbance environments, the static $Q$ matrix based finite-time FBC and ILC controller is clearly not more optimal than even the basic ILC, which can be realized by setting the $Q$ matrix to zero. Figures 4-15a, 4-15b, 4-15d, and 4-18a illustrate this claim. However, the static $Q$ matrix based combination controller is still better than the PID FBC implementation.

## 4.4: Summary of Findings

Optimal cost function based combination controllers, incorporating both finite-time FBC and ILC, have been developed for learning controller applications in linear systems. These controllers are dependent on the $Q$ matrix to implement the finite-time feedback controller. Two separate designs, based on a static or dynamic $Q$ matrix, have been developed. The $Q$ matrix is found simultaneously with the learning controller by specifying a few gains in the most basic case. Specific design considerations are evaluated to take full advantage of the inherent flexibility of the $Q$ matrix based design for learning control applications. These considerations cover the fundamental aspects of this design approach including appropriate generation of the primary design agent, the training matrix, as well as the use of finite-time sensitivity transfer matrices as an evaluation tool. It has been demonstrated that finite-time sensitivity transfer matrices can be used to reliably bridge the gap between finite-time control design approaches and more conventional, steady-state, control design approaches. Numerical simulations confirm that these combination controllers significantly improve both tracking performance and learning behavior when compared to conventional feedback control only methods. It has been further demonstrated that the most optimal design, which guarantees optimal performance in all situations, is the dynamic $Q$ matrix based finite-time FBC and ILC controller.

## General Setup for Numerical Simulations

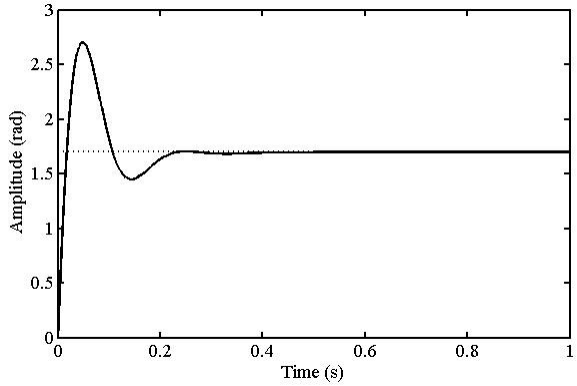



Figure 4-1. Well-Conditioned Plant Step Response

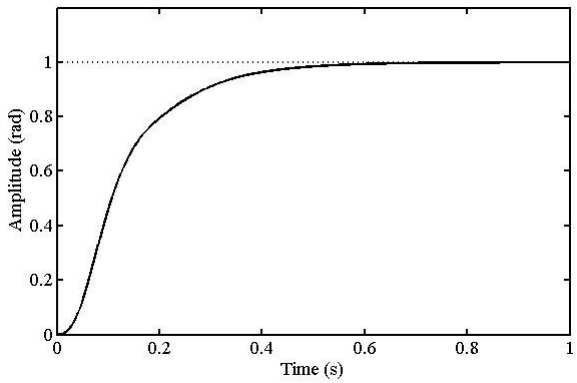
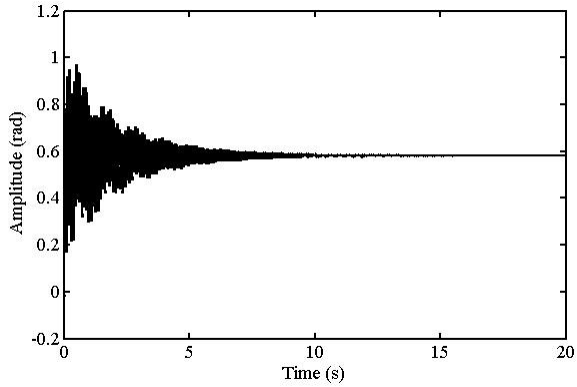



Figure 4-2. Ill-Conditioned Plant Step Response



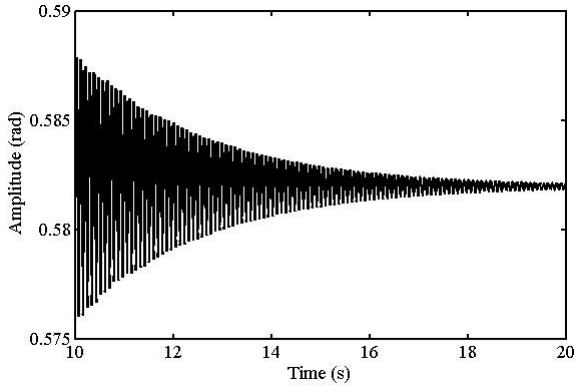




Figure 4-3. Spine Model Step Response

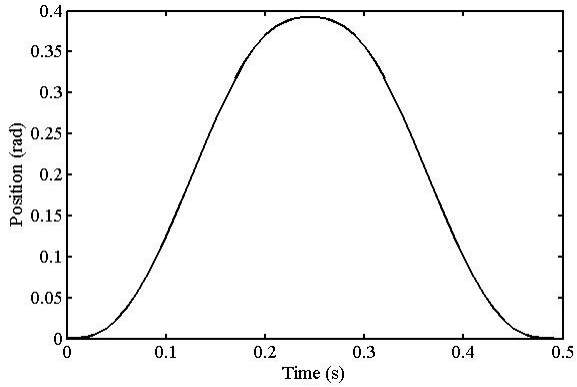



Figure 4-4. Desired Trajectory One

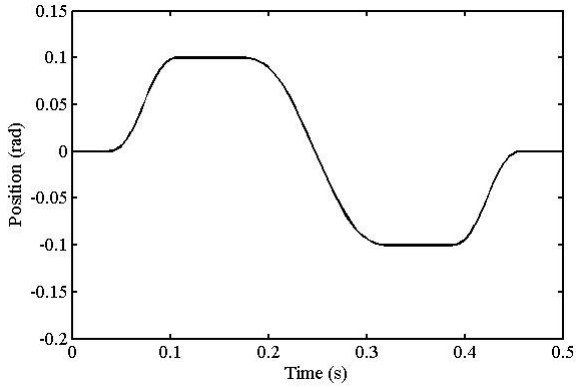



Figure 4-5. Desired Trajectory Two

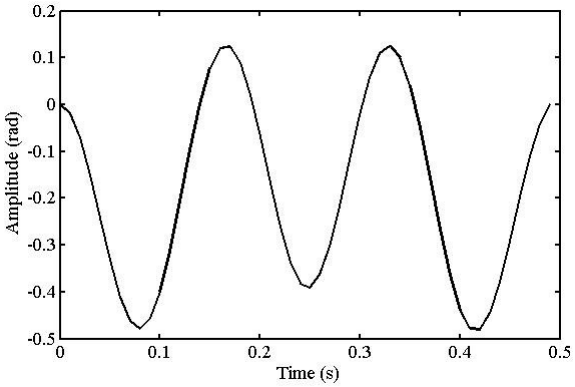



**Figure 4-6. Repeating Disturbance One**

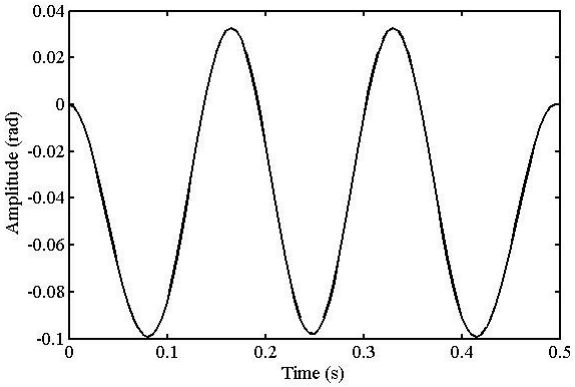



**Figure 4-7. Repeating Disturbance Two**

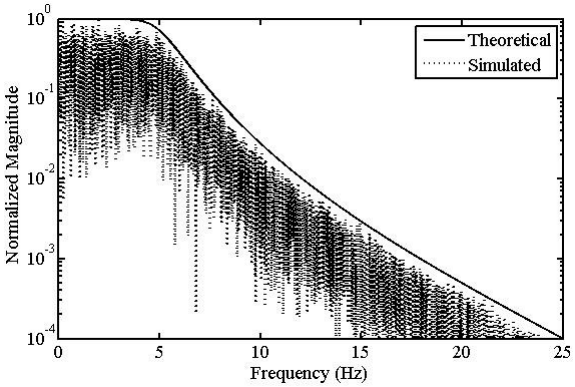



**Figure 4-8. Random Disturbance Spectrum One**

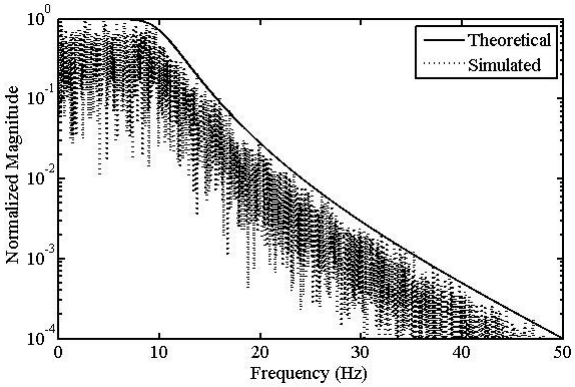



**Figure 4-9. Random Disturbance Spectrum Two**

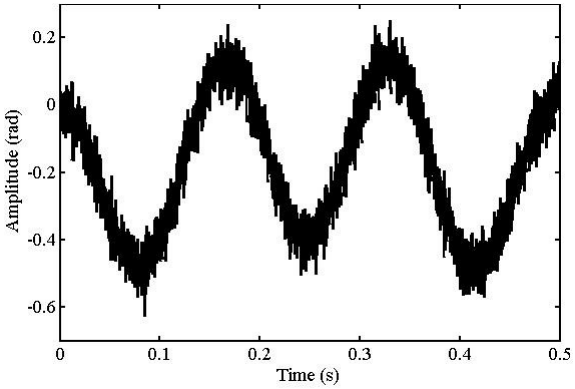



**Figure 4-10. Combined Disturbance Example One**

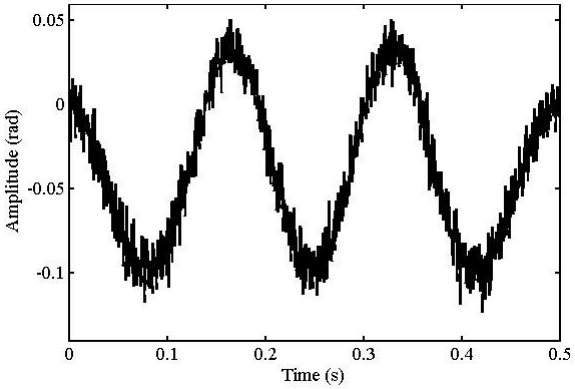
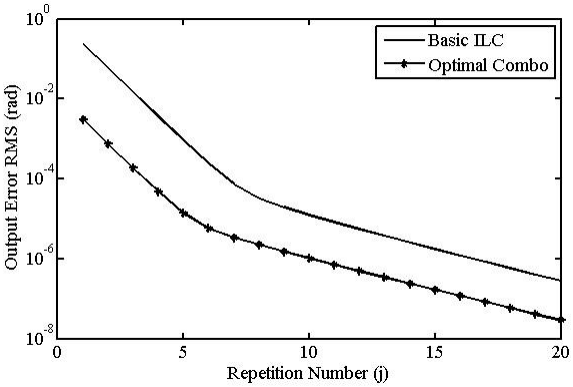
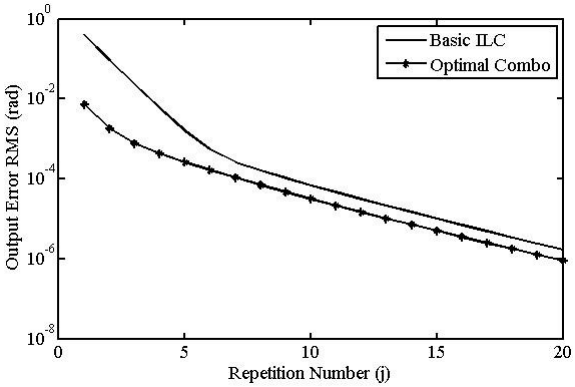



**Figure 4-11. Combined Disturbance Example Two**

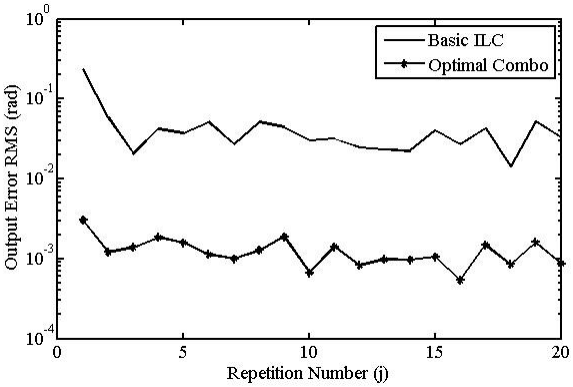
## Well-Conditioned Plant Model



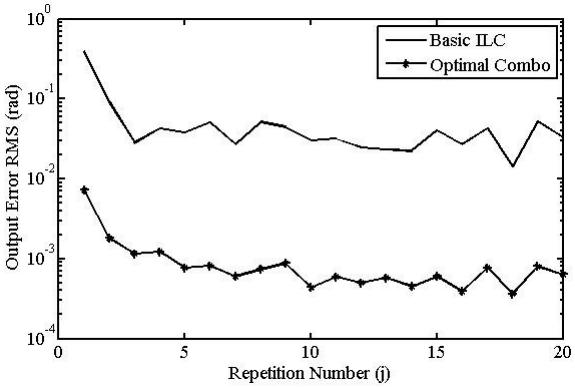

a. Ideal Case with No Disturbance



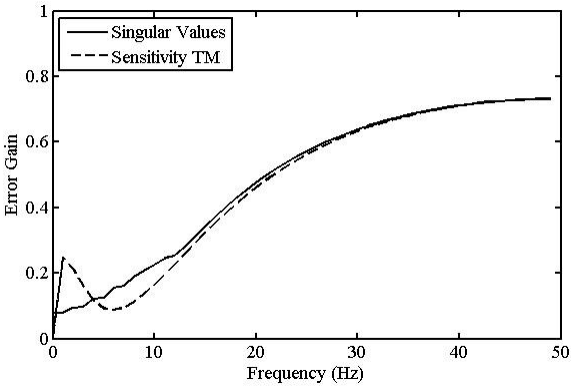

b. Repeating Disturbance



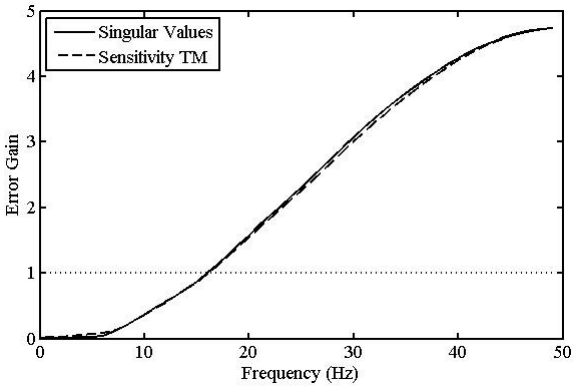

c. Random Disturbance




d. Combined Disturbance

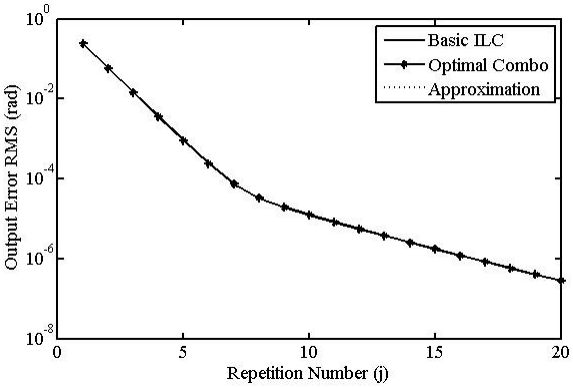



e. Previous Cycle Error Sensitivity TM

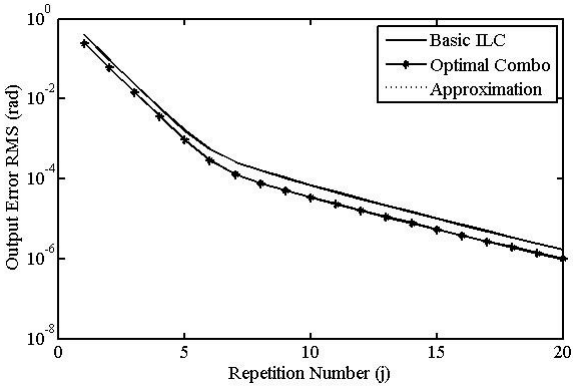



f. Disturbance Difference Sensitivity TM

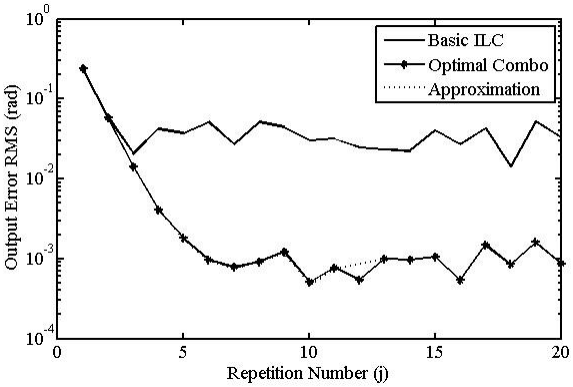
**Figure 4-12. Static Q Based Finite-Time FBC and ILC for Well-Conditioned Model**



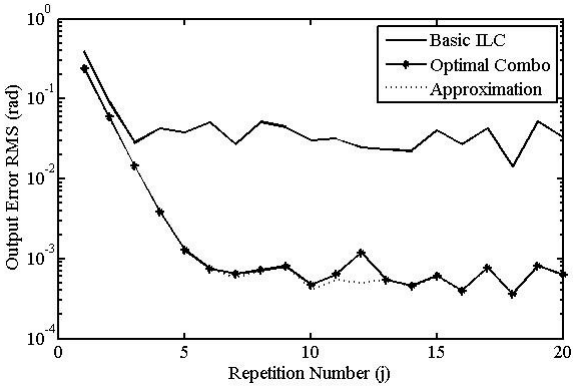

a. Ideal Case with No Disturbance



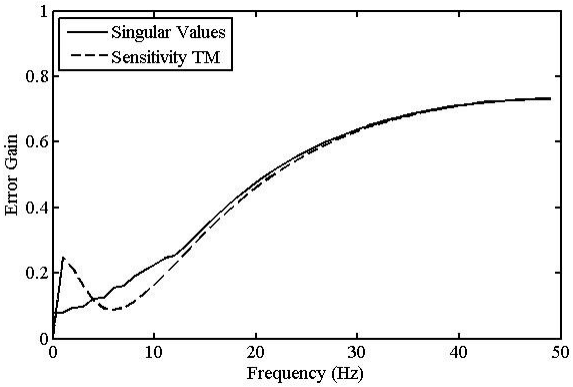

b. Repeating Disturbance



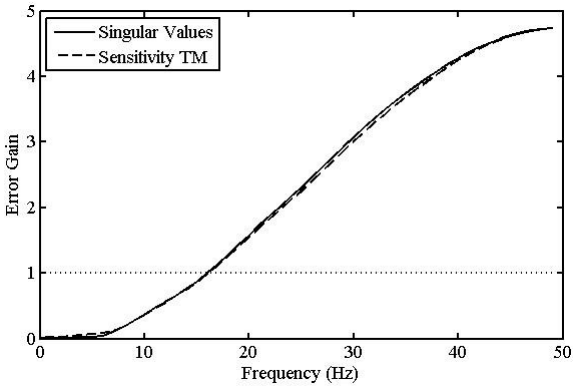

c. Random Disturbance




d. Combined Disturbance

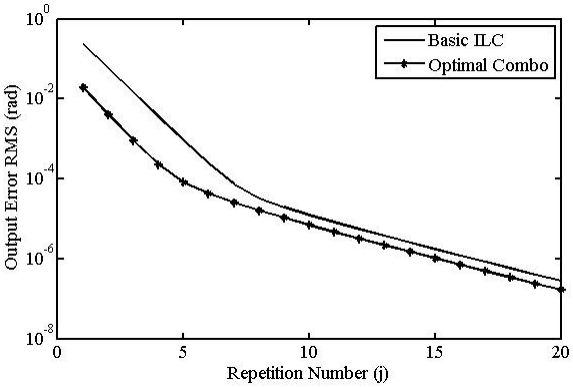



e. Previous Cycle Error Sensitivity TM

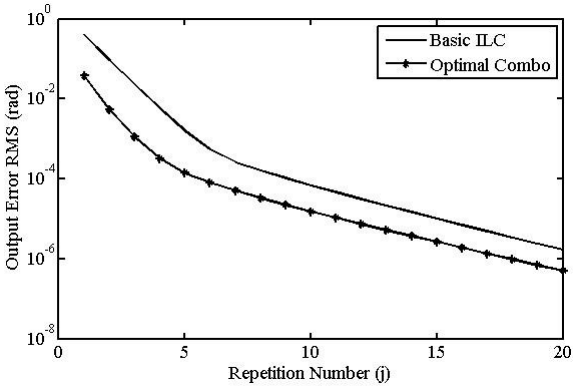



f. Disturbance Difference Sensitivity TM

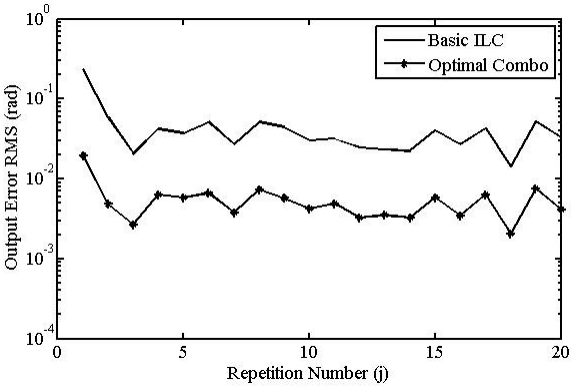
**Figure 4-13. Dynamic Q Based Finite-Time FBC and ILC for Well-Conditioned Model**



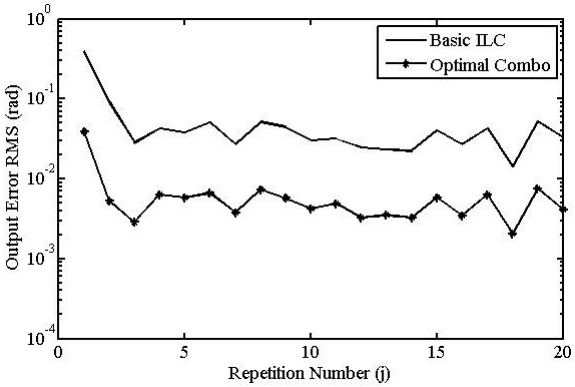

a. Ideal Case with No Disturbance



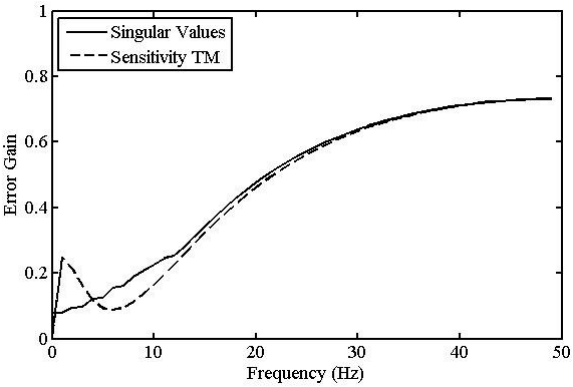

b. Repeating Disturbance



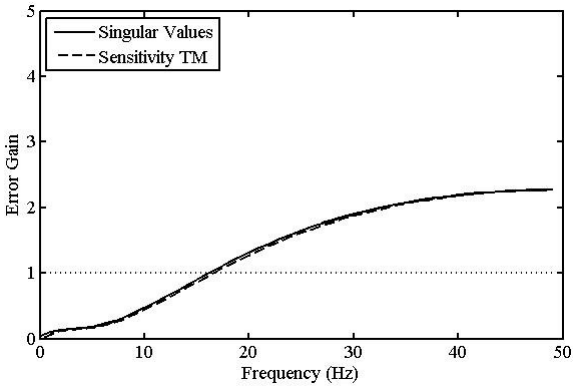

c. Random Disturbance




d. Combined Disturbance



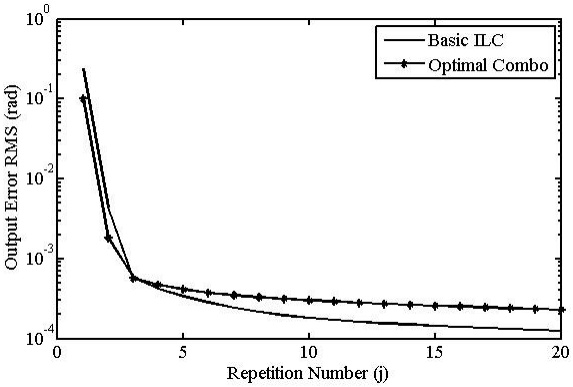

e. Previous Cycle Error Sensitivity TM



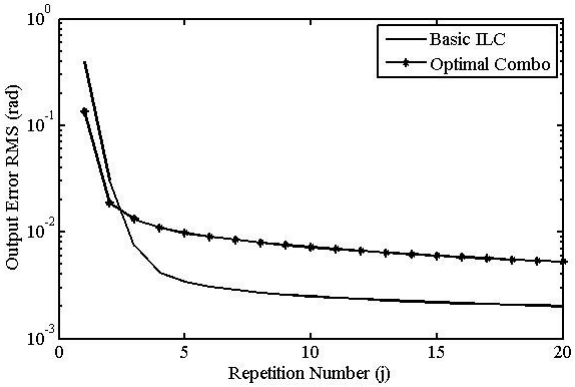

f. Disturbance Difference Sensitivity TM

**Figure 4-14. Static Q Based Conventional PI FBC and ILC for Well-Conditioned Model**

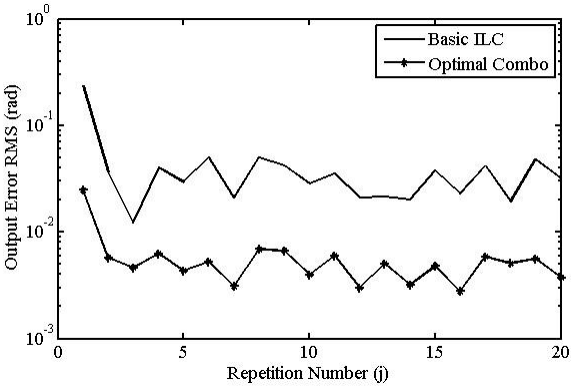
## Ill-Conditioned Plant Model



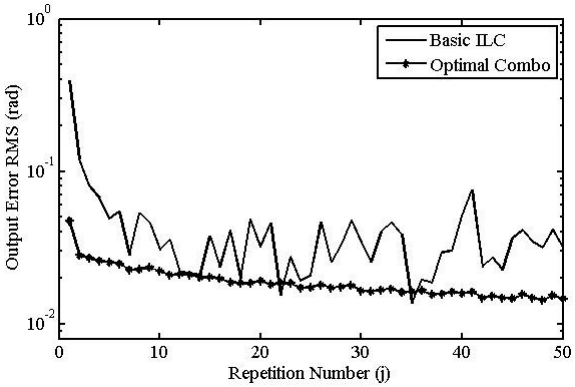

a. Ideal Case with No Disturbance



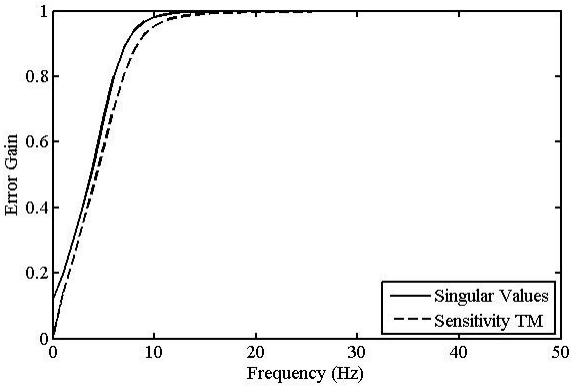

b. Repeating Disturbance



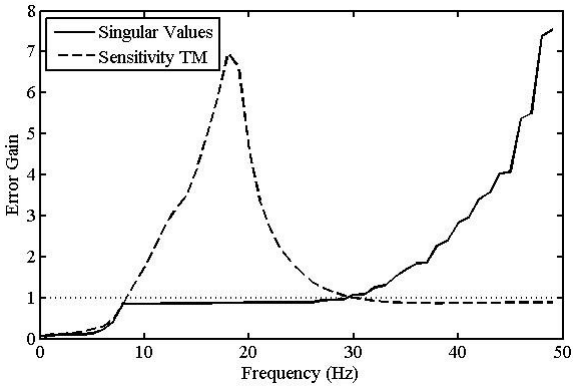

c. Random Disturbance




d. Combined Disturbance

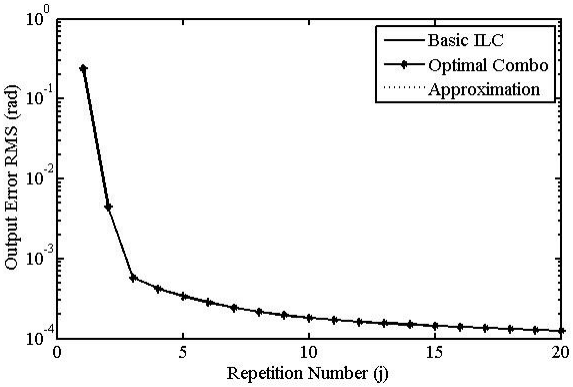



e. Previous Cycle Error Sensitivity TM

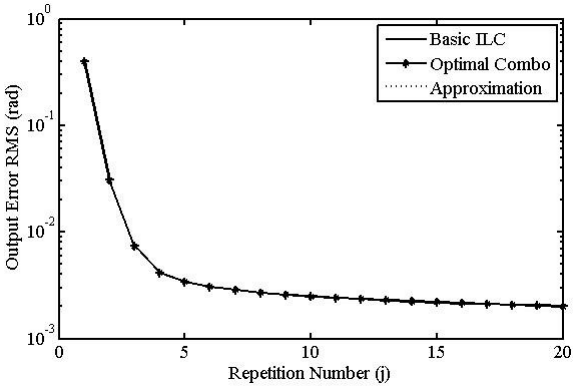



f. Disturbance Difference Sensitivity TM

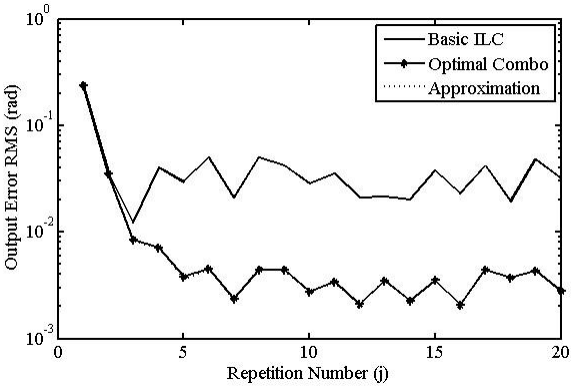
**Figure 4-15. Static Q Based Finite-Time FBC and ILC for Ill-Conditioned Model**



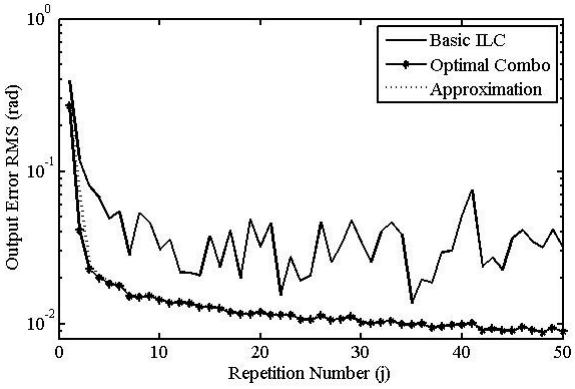

a. Ideal Case with No Disturbance



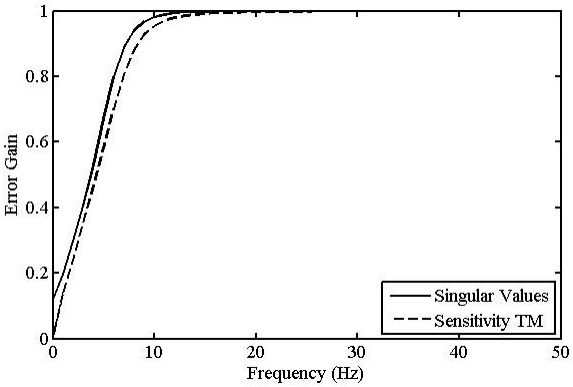

b. Repeating Disturbance



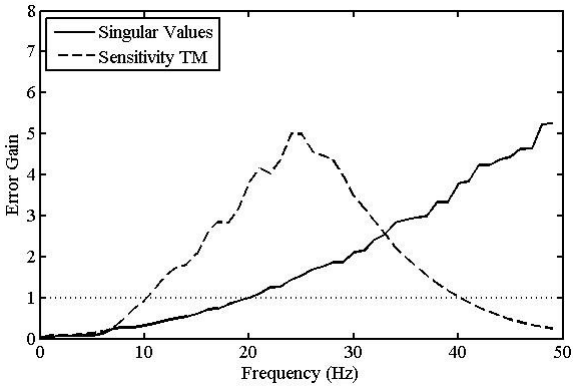

c. Random Disturbance




d. Combined Disturbance

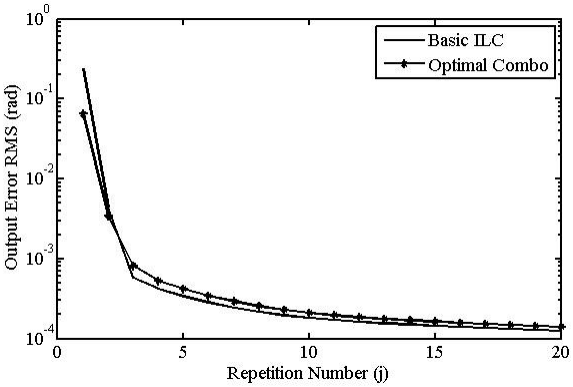



e. Previous Cycle Error Sensitivity TM

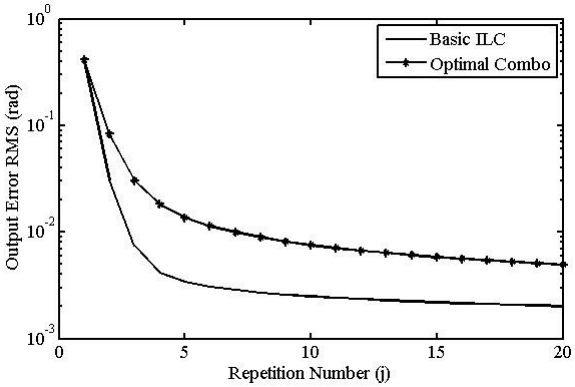



f. Disturbance Difference Sensitivity TM

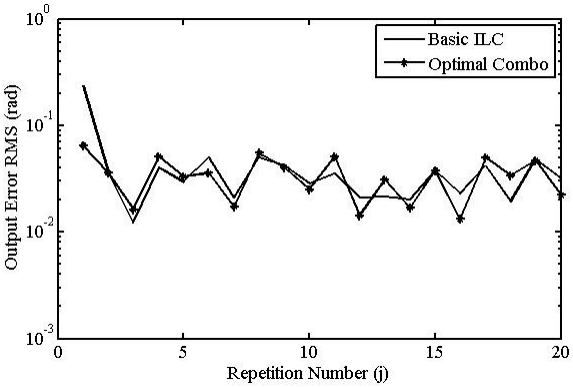
**Figure 4-16. Dynamic Q Based Finite-Time FBC and ILC for Ill-Conditioned Model**



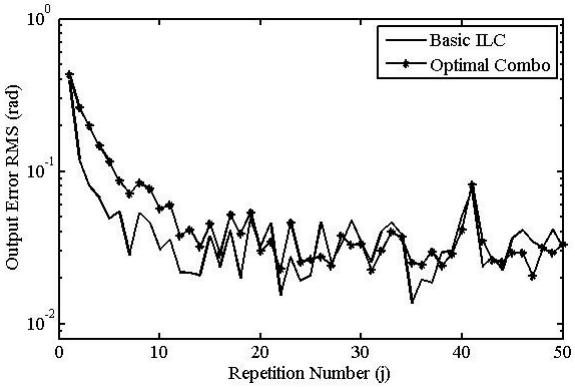

a. Ideal Case with No Disturbance



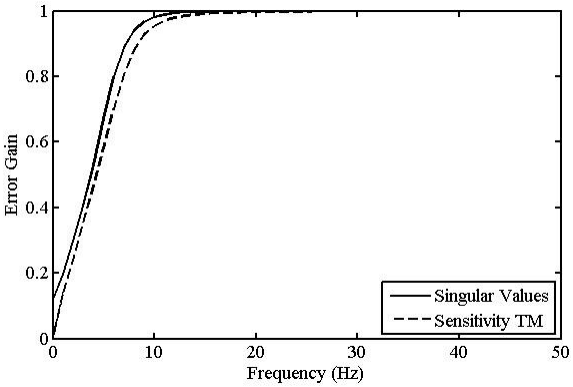

b. Repeating Disturbance



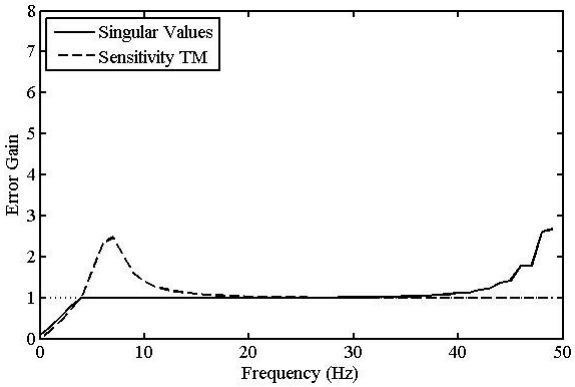

c. Random Disturbance




d. Combined Disturbance



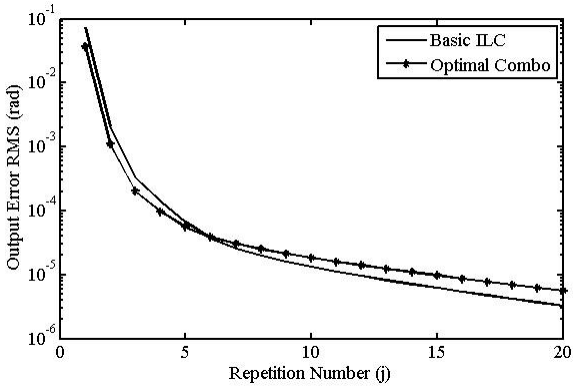

e. Previous Cycle Error Sensitivity TM



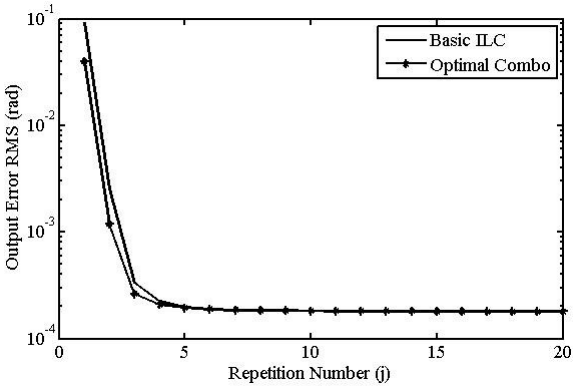

f. Disturbance Difference Sensitivity TM

**Figure 4-17. Static Q Based Conventional PID FBC and ILC for Ill-Conditioned Model**

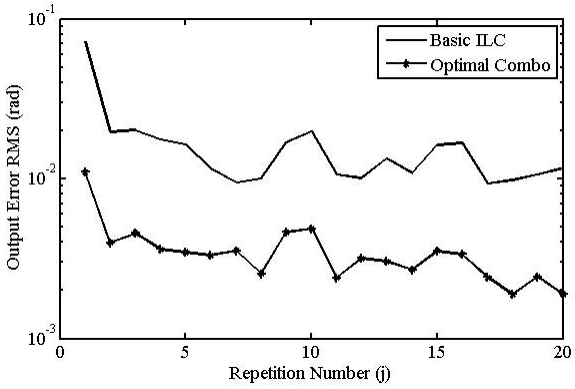
## Spine Model



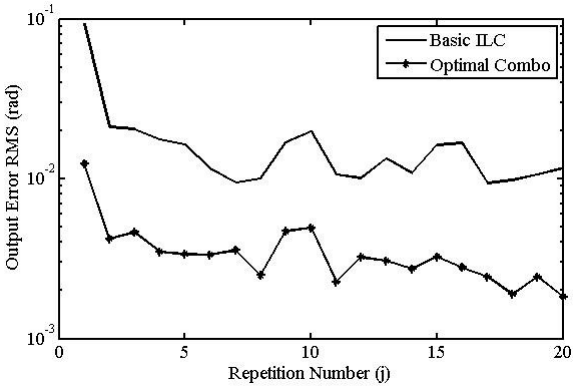

a. Ideal Case with No Disturbance



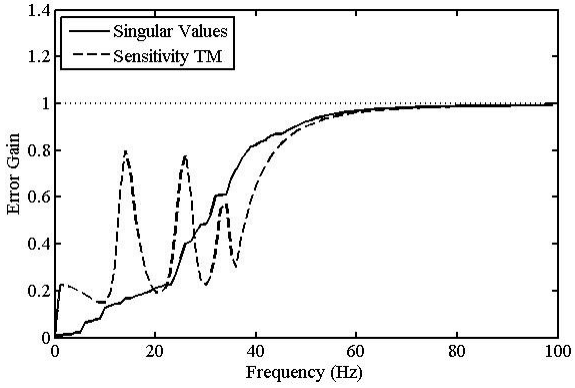

b. Repeating Disturbance



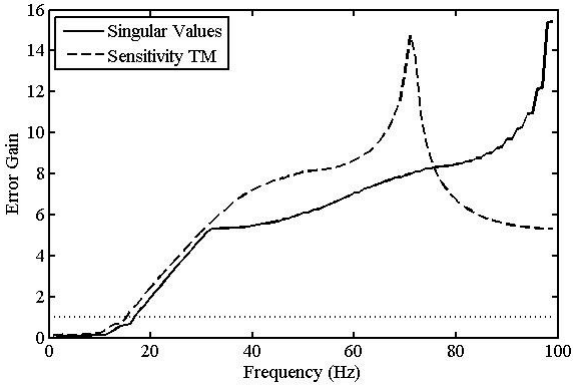

c. Random Disturbance




d. Combined Disturbance

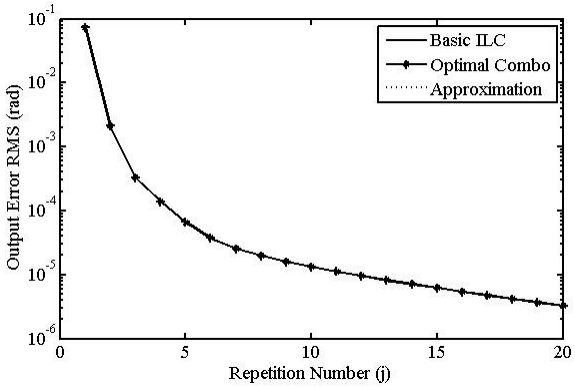



e. Previous Cycle Error Sensitivity TM

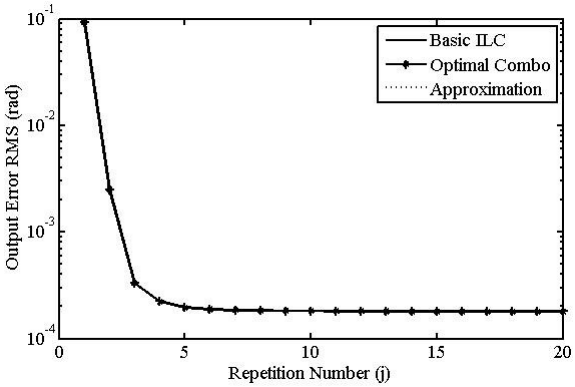



f. Disturbance Difference Sensitivity TM

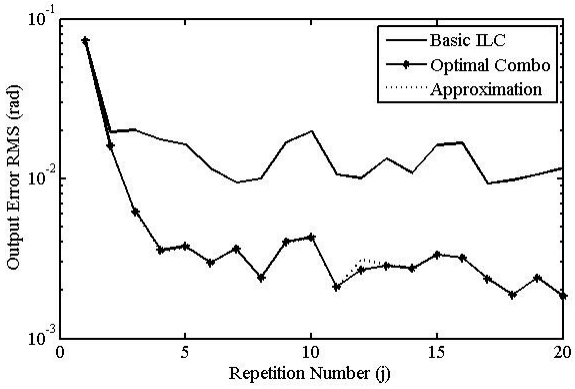
**Figure 4-18. Static Q Based Finite-Time FBC and ILC for Spine Model**



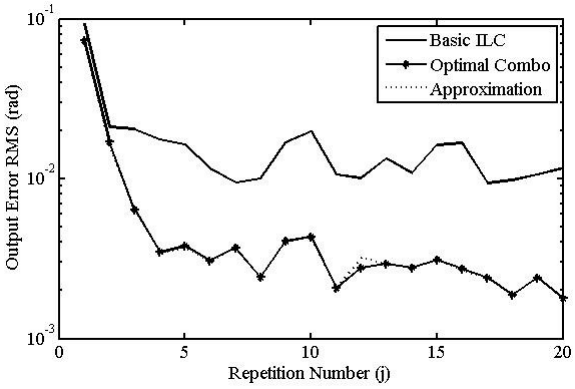

a. Ideal Case with No Disturbance



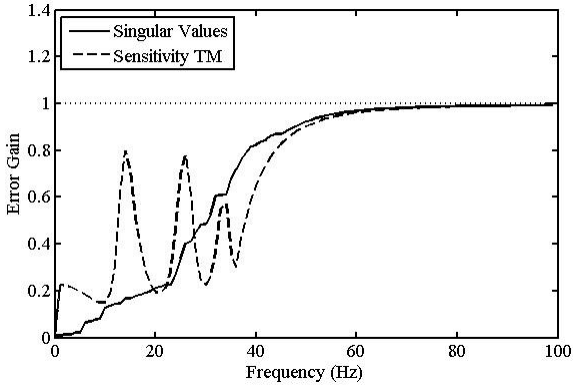

b. Repeating Disturbance



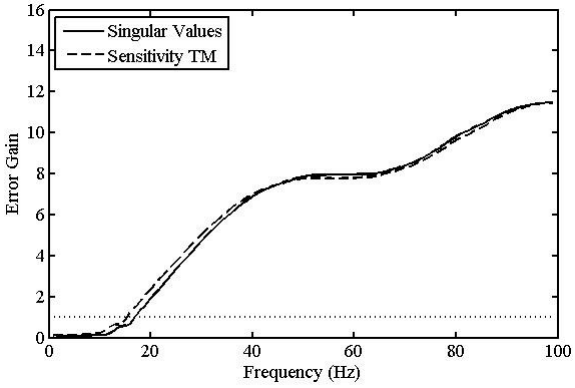

c. Random Disturbance




d. Combined Disturbance

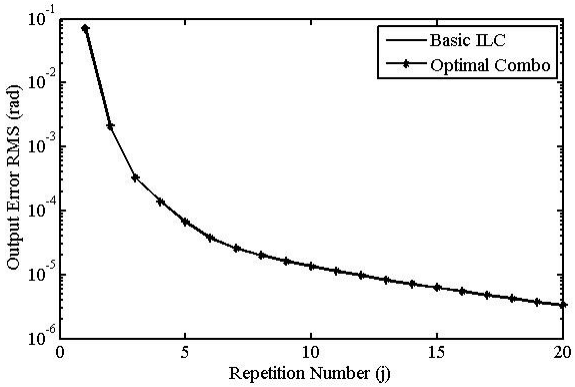



e. Previous Cycle Error Sensitivity TM

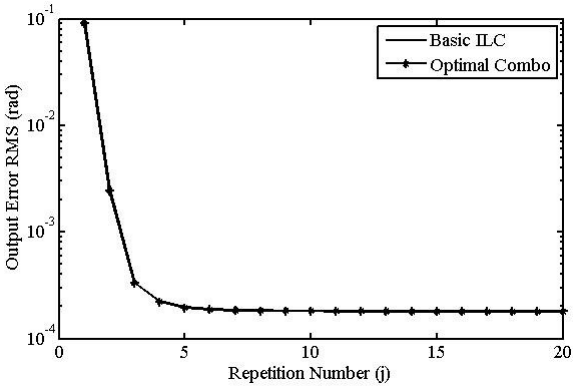



f. Disturbance Difference Sensitivity TM

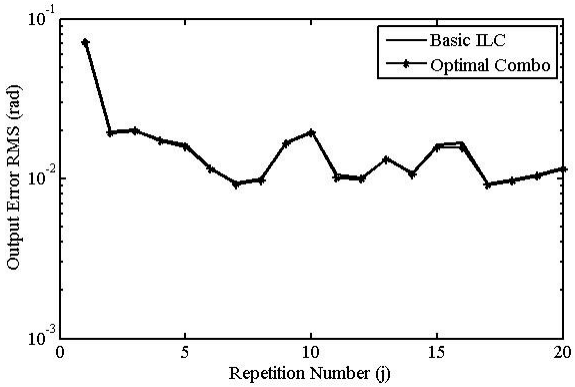
**Figure 4-19. Dynamic Q Based Finite-Time FBC and ILC for Spine Model**



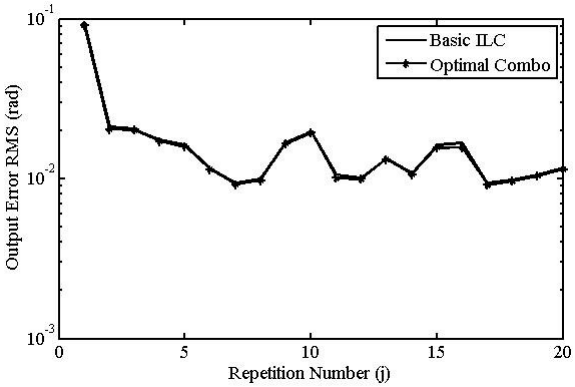

a. Ideal Case with No Disturbance



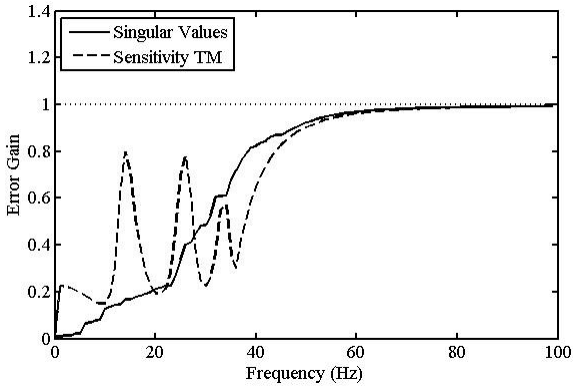

b. Repeating Disturbance



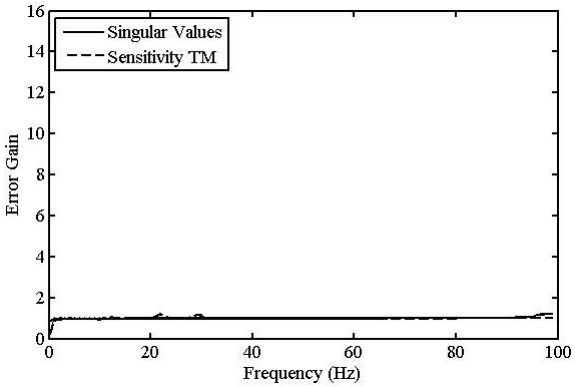

c. Random Disturbance




d. Combined Disturbance




e. Previous Cycle Error Sensitivity TM




f. Disturbance Difference Sensitivity TM

**Figure 4-20. Static Q Based Conventional PID FBC and ILC for Spine Model**

# CHAPTER FIVE: Configurations for Feedback and Iterative Learning Combination Controllers

It should be clear from Chapter 4 that for practical applications in which ILC can be considered, use of complementary FBC will be advantageous. Simultaneously designed combination controllers have been shown to provide additional tracking and disturbance rejection capabilities over ILC alone. Optimal combinations controllers for learning control applications have been explored in prior studies, however, an exhaustive investigation of various configuration options is lacking in the literature. As stated in Chapter 1, detailed evaluations of possible block diagrams for RC and the effects of placing the repetitive controller inside or outside the feedback loop are available.[70-72] A similar comprehensive assessment is performed in this chapter for ILC by taking a detailed look at different configuration options for feedback and learning control combinations. Some novel designs for the controller implementation are also presented.

This chapter is organized in the following manner. In the first section, the learning control problem is formulated to support time-domain analysis for the FBC portion, and repetition-domain analysis for the ILC portion, of the combination controller. Assessment of the various configurations begins in the second section with the most basic control scheme, the open-loop ILC configuration. The first sub-section of section two presents the corresponding optimal controllers for this basic configuration, followed by a derivation of the expected error propagation for each of the associated optimal controllers in the second sub-section. Section three continues the assessment by considering the first of two closed-loop FBC and ILC configurations. Closed-loop FBC

and ILC configuration one considers controller types where the learning signal supplements the FBC signal to improve the overall input to the plant. The first sub-section of section three presents the optimal controllers for closed-loop configuration one. Next, the expected error propagation is similarly established for the corresponding optimal controllers in the second sub-section. Section four furthers the assessment by investigating closed-loop FBC and ILC configuration two, which considers controller types where the learning signal supplements the desired signal or the command, effectively modifying the command to the existing feedback control system. Again, the two sub-sections of this section establish the optimal controllers and expected error performance specific to this type of control configuration. The assessment is concluded in the remaining sections by performing an analytical comparison of the configurations starting with an ideal setup using an inverse feedback controller, then a more practical setup using a classic PI controller, and finally conducting numerical simulations to demonstrate the findings.

## 5.1: Basic Open-Loop ILC Configuration

In order to rigorously assess the extent to which the incorporation of feedback control can improve performance in learning control applications, it is necessary to first establish how the basic open-loop ILC configuration performs. A block diagram of this elementary configuration is given in Figure 5-1 and it should be clear that there is no distinction between the plant input $\underline{u}_j$ and the learning control signal $\underline{v}_j$ in this setup.

### 5.1.1: Optimal Open-Loop Learning Controller Design

The input to output relationship is described entirely by Eq. (1.3) in the time-domain for any single iteration and by Eq. (1.8) in the repetition-domain for the duration of the

application. With this setup, the designer has the option to minimize one of two cost functions to optimize performance

$$J_1 = \left[\underline{e}_j\right]^T Q_e \underline{e}_j + \left[\underline{u}_j\right]^T Q_u \underline{u}_j \tag{5.1}$$

$$J_2 = \left[\underline{e}_j\right]^T Q_e \underline{e}_j + \left[\delta_j \underline{u}\right]^T Q_u \delta_j \underline{u} \tag{5.2}$$

Here, from a simple comparison of Eq. (5.1) and Eq. (5.2), it should be clear that the only difference between the two cost functions is that $J_1$ aims to minimize the current cycle control actions and $J_2$ aims to minimize the change in the control actions in repetition. Relative to each other, only $J_2$ can truly be minimized to zero in an optimal fashion, since minimizing $J_1$ to zero requires that the plant input $\underline{u}_j$ to be heavily penalized or removed entirely, which is undesirable. For both cost functions, the aim is to determine the optimal updates to the control actions. To do this, the cost functions must first be modified into the correct format which explicitly shows the implicit relationships between the control variable to be optimized. That control variable will be $\delta_j \underline{v}$, or updates to the learning signal, for the optimizations conducted in this work. For the basic open-loop ILC configuration, the fundamental relationship between the control variable $\delta_j \underline{v}$ and the other critical parameters of interest, $\delta_j \underline{u}$ and $\delta_j \underline{e}$, in $J_1$ and $J_2$ are

$$\partial \delta_j \underline{u} \Big/ \partial \delta_j \underline{v} = I \tag{5.3}$$

$$\partial \delta_j \underline{e} \Big/ \partial \delta_j \underline{v} = -P \tag{5.4}$$

The critical relation given in Eq. (5.3) can be verified using the simple fact that $\underline{u}_j = \underline{v}_j$, so $\delta_j \underline{u} = \delta_j \underline{v}$ for this configuration. The other critical relation, given in Eq. (5.4), can be concluded using Eqs. (5.3) and (1.7), presented in the problem formulation section. Both of these fundamental results will be necessary to simplify and complete the optimization

process. It is useful to explicitly show the critical parameters of interest, so the difference operator is used to replace the current cycle error $\underline{e}_j$ according to the relationship $\delta_j\underline{e} = \underline{e}_j - \underline{e}_{j-1}$ and the current cycle input $\underline{u}_j$ according to the relationship $\delta_j\underline{u} = \underline{u}_j - \underline{u}_{j-1}$ as follows

$$J_1 = \left[\delta_j\underline{e} + \underline{e}_{j-1}\right]^T Q_e \left[\delta_j\underline{e} + \underline{e}_{j-1}\right] + \left[\delta_j\underline{u} + \underline{u}_{j-1}\right]^T Q_u \left[\delta_j\underline{u} + \underline{u}_{j-1}\right] \tag{5.5}$$

$$J_2 = \left[\delta_j\underline{e} + \underline{e}_{j-1}\right]^T Q_e \left[\delta_j\underline{e} + \underline{e}_{j-1}\right] + \left[\delta_j\underline{u}\right]^T Q_u \delta_j\underline{u} \tag{5.6}$$

Now, the cost functions given in Eqs. (5.5) and (5.6) are ready to be optimized by taking the partial derivative with respect to $\delta_j\underline{v}$, through appropriate use of both Eqs. (5.3) and (5.4), and setting to zero

$$\partial J_1 \Big/ \partial \delta_j\underline{v} = 2P^T Q_e P \delta_j\underline{v} - 2P^T Q_e \underline{e}_{j-1} + 2Q_u \delta_j\underline{v} + 2Q_u \underline{u}_{j-1} = 0 \tag{5.7}$$

$$\partial J_2 \Big/ \partial \delta_j\underline{v} = 2P^T Q_e P \delta_j\underline{v} - 2P^T Q_e \underline{e}_{j-1} + 2Q_u \delta_j\underline{v} = 0 \tag{5.8}$$

It is worthwhile to note that $\delta_j\underline{d}$ is not present in Eqs. (5.7) and (5.8), since the fundamental relationship given in Eq. (5.4) was used. This is advantageous since $\delta_j\underline{d}$ is generally unknown and difficult, if not impossible, to accurately model and predict. If a reliable estimate of $\delta_j\underline{d}$ is available, it can be utilized by redoing the optimization, starting from Eqs. (5.5) and (5.6), and using Eq. (1.7) to explicitly introduce $\delta_j\underline{d}$ into the derivation. If such a derivation was done, the resultant partial derivatives would be

$$\partial J_1 \Big/ \partial \delta_j\underline{v} = 2P^T Q_e P \delta_j\underline{v} - 2P^T Q_e \underline{e}_{j-1} + 2P^T Q_e \delta_j\underline{d} + 2Q_u \delta_j\underline{v} + 2Q_u \underline{u}_{j-1} \tag{5.9}$$

$$\partial J_2 \Big/ \partial \delta_j\underline{v} = 2P^T Q_e P \delta_j\underline{v} - 2P^T Q_e \underline{e}_{j-1} + 2P^T Q_e \delta_j\underline{d} + 2Q_u \delta_j\underline{v} \tag{5.10}$$

It is important to note in the majority of practical applications, $\delta_j\underline{d}$ is typically neglected in such update equations without consequence. In fact these results, allowing for the use of

a reliable estimate of $\delta_j \underline{d}$, are included only for completeness. The more generally applicable minimizations in the vast majority of implementations are not a function of unknown disturbances. Therefore, only the results obtained in Eqs. (5.7) and (5.8) are considered further. These two minimizations are identical except for the last term in Eq. (5.7), which is a direct result of using a cost function that penalizes the current cycle control actions, as opposed to the iterative updates to the control actions. Here the basic open-loop ILC configuration will produce optimal control with respect to $J_1$ if

$$\delta_j \underline{v} = (P^T Q_e P + Q_u)^{-1}\left[P^T Q_e \underline{e}_{j-1} - Q_u \underline{u}_{j-1}\right] \tag{5.11}$$

$$\underline{v}_j = \underline{v}_{j-1} + (P^T Q_e P + Q_u)^{-1} P^T Q_e \underline{e}_{j-1} - (P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1}$$

The term $(P^T Q_e P + Q_u)^{-1} P^T Q_e$ will reappear in numerous subsequent derivations, so $L^*$ will be used to represent it purely for mathematical convenience

$$\begin{aligned} \delta_j \underline{v} &= (P^T Q_e P + Q_u)^{-1} P^T Q_e \underline{e}_{j-1} - (P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} \\ &= L^* \underline{e}_{j-1} - (P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} \end{aligned} \tag{5.12}$$

This is the recommended optimal rule, with respect to $J_1$, for updating the learning signal. With respect to $J_2$, the basic open-loop ILC configuration will produce optimal control if

$$\begin{aligned} \delta_j \underline{v} &= (P^T Q_e P + Q_u)^{-1} P^T Q_e \underline{e}_{j-1} \\ &= L^* \underline{e}_{j-1} \end{aligned} \tag{5.13}$$

Both of these optimal control designs only apply previous cycle feedback, so current cycle control is not possible. This means that optimal performance can only be expected in repetition and not in run, or individual iterations, which is clearly limiting. However, these designs alone are not sufficient to adequately predict the overall performance limitations when implemented, since they only describe how the control actions should be updated and not the resulting error.

### 5.1.2: Error Propagation of Optimal Open-Loop Learning Controller Designs

In order to thoroughly assess how the incorporation of current cycle feedback into the control scheme improves performance, it is necessary to determine the error performance of the basic open-loop ILC configuration using an optimized controller. To do this for the optimal controller resulting from $J_1$, start with Eq. (1.7) and plug in for the optimal updates to the plant input using Eq. (5.12), and recalling $\delta_j \underline{u} = \delta_j \underline{v}$ in this configuration, as shown

$$\delta_j \underline{e} = -P\left[L^* \underline{e}_{j-1} - (P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1}\right] - \delta_j \underline{d} \tag{5.14}$$

After expansion of the difference operator on error and some expansion of terms, it is clear to see that Eq. (5.14) can be rewritten as

$$\left(\underline{e}_j - \underline{e}_{j-1}\right) = -PL^* \underline{e}_{j-1} + P(P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} - \delta_j \underline{d} \tag{5.15}$$

Some additional arithmetic to collect like terms will produce the desired result

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} + P(P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} - \delta_j \underline{d} \tag{5.16}$$

It should be clear, from Eq. (5.16), that the optimal controller produced by minimizing $J_1$ can never result in error free control. Even if inverse control was attempted, by setting $L^* = P^{-1}$, zero error is not attainable because $\underline{u}_{j-1}$ and $\delta_j \underline{d}$ are non-zero in general. Now, the same derivation is done for the optimal controller resulting from $J_2$, starting again with Eq. (1.7) and plugging in the optimal update rule Eq. (5.13), once more recalling $\delta_j \underline{u} = \delta_j \underline{v}$, as follows

$$\delta_j \underline{e} = -P\left(L^* \underline{e}_{j-1}\right) - \delta_j \underline{d} \tag{5.17}$$

Following some simple standard arithmetic to collect like terms and rearrange the equation, it is apparent that Eq. (5.17) simplifies to

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - \delta_j \underline{d} \tag{5.18}$$

The error performance predicted by Eq. (5.18) shows that the basic open-loop learning controller optimized with respect to $J_2$ also does not generally have the potential to produce perfect, error free control. Even using inverse control, which is generally not recommended, $\delta_j \underline{d}$ imposes a strict limit on attainable performance. If the disturbance environment is such that only repeating disturbances are present, or $\underline{\Delta}_j = \underline{d}$, then $\delta_j \underline{\Delta} = \delta_j \underline{d} = 0$ and zero error is possible if, and only if,

$$|\lambda_i(I - PL^*)| < 1 \qquad \forall\, i \tag{5.19}$$

Here, Eq. (5.19) states that all possible tracking errors will converge if all of the eigenvalues of the corresponding propagation matrix are less than one in magnitude. Similarly, the transient behavior of the controller, in the same disturbance environment, is determined by the singular values of the propagation matrix. In order to ensure well behaved transients

$$\sigma_i(I - PL^*) < 1 \qquad \forall\, i \tag{5.20}$$

It is important to reiterate that Eqs. (5.19) and (5.20) describe the overall stability and transient behavior for disturbance environments consisting of a repeating disturbance only. The more general disturbance environment represented by Eq. (1.4) is the one that will be generally used in this chapter. In this disturbance environment, Eqs. (5.16) and (5.18) show that the true limitations of the basic open-loop ILC configuration, regardless of optimization, stem from its inability to compensate for non-repeating, stochastic disturbances. It should be clear that when $\delta_j \underline{d} \neq 0$, it can be considered a non-zero forcing function for current cycle error. When such a driver is present, the incorporation of feedback control can considerably improve the overall performance.

## 5.2: Closed-Loop FBC and ILC Configuration One

There are essentially two fundamentally distinct ways in which a feedback and learning combination controller can be configured, one with the learning signal incorporated inside the feedback loop and one with the learning signal incorporated outside the feedback loop. In this section, the configuration with the learning signal incorporated inside the loop will be evaluated. Though there are different ways to input the learning signal in this manner, Figure 5-2 shows the recommended configuration for evaluation purposes. This is because the other ways can usually be considered special cases of this configuration. Therefore, the results of this investigation can be appropriately modified for use in such cases as desired.

### 5.2.1: Optimal Closed-Loop Combination Controller Design for Configuration One

As with the basic open-loop configuration, cost functions will once again be utilized to optimize performance of the control implementation. However, before the cost functions are introduced, it is necessary to properly formulate the input signal to be optimized. Unlike in Figure 5-1, the plant input $\underline{u}_j$ and learning control signal $\underline{v}_j$ in Figure 5-2 are clearly not the same. For closed-loop FBC and ILC configuration one, hereafter referred to simply as closed-loop configuration one, the plant input is

$$\underline{u}_j = \underline{u}_{FB} + \underline{v}_j \tag{5.21}$$

Here, $\underline{u}_{FB}$ is the output of the feedback controller, which is defined with respect to the configuration as

$$\underline{u}_{FB} = R\underline{u}_{FB} + G\underline{e}_j \tag{5.22}$$

Using Eq. (5.21) to solve for $\underline{u}_{FB}$ and then plugging that result into Eq. (5.22) will give

$$\left(\underline{u}_j - \underline{v}_j\right) = R\left(\underline{u}_j - \underline{v}_j\right) + G\underline{e}_j \tag{5.23}$$

Some additional arithmetic, to expand Eq. (5.23), collect like terms, and simplify the resultant, produces the desired final expression

$$\begin{aligned} \underline{u}_j &= (I - R)^{-1} G \underline{e}_j + \underline{v}_j \\ &= C \underline{e}_j + \underline{v}_j \end{aligned} \tag{5.24}$$

Note that matrix $C$ is introduced to represent $(I - R)^{-1} G$, the overall time-domain FBC gain matrix, for the same reason $L^*$ was introduced in Eq. (5.12), ease of use in subsequent derivations. It is also worth noting that there is an underlying assumption in Eq. (5.24) that $(I - R)^{-1}$ exists, which is generally valid since $R$ is part of the FBC and can typically be designed to ensure the assumption is valid. It can be seen from Eq. (5.24) that with this type of configuration, the ILC signal can be thought to supplement the FBC signal to improve the overall input to the plant. Before introducing the various cost functions which are considered to optimize this control implementation, the fundamental relationships between the control variable $\delta_j \underline{v}$ and the other critical parameters of interest, $\delta_j \underline{u}$ and $\delta_j \underline{e}$, must be established for close-loop configuration one. These relationships will be analogous to those specified in Eqs. (5.3) and (5.4) for the basic open-loop ILC configuration. In order to derive these essential relations, start with Eq. (5.24) and apply the difference operator $\delta_j$ as shown

$$\delta_j \underline{u} = C \delta_j \underline{e} + \delta_j \underline{v} \tag{5.25}$$

Now, using Eq. (1.7), it is possible to rewrite Eq. (5.25) by substituting in for the unknown iterative changes to the error, which are to be determined

$$\delta_j \underline{u} = C\left(-P \delta_j \underline{u} - \delta_j \underline{d}\right) + \delta_j \underline{v} \tag{5.26}$$

After some standard arithmetic to collect like terms and simplify the equation, it can be seen that Eq. (5.26) can be rewritten as

$$\begin{aligned}\delta_j \underline{u} &= (I + CP)^{-1}\delta_j \underline{v} - (I + CP)^{-1} C \delta_j \underline{d} \\ &= K\delta_j \underline{v} - KC\delta_j \underline{d}\end{aligned} \tag{5.27}$$

Here, matrix $K$ is introduced for two reasons, mathematical convenience and also because it turns out to be a key parameter in the overall investigation of the control configurations considered. The reason $K$ is key to this investigation is that it can be thought of as the single parameter used to describe the overall repetition-domain feedback control (RFBC) actions. It should be highlighted that $K$ is assumed to exist, and this is generally valid for the same reasons as with Eq. (5.24), since $C$ represents the FBC and can similarly be designed to ensure the assumption is valid. This equation gives the relation between $\delta_j \underline{v}$ and $\delta_j \underline{u}$, which is necessary to establish the first fundamental relationship for optimization of the control actions produced by closed-loop configuration one. Before deriving the second essential relation, a useful proof is presented which will allow Eq. (5.27) to be rewritten in another equivalent form. Let $\{E, F\} \in \mathbb{R}^{p\times p}$, where $E$ is selected to ensure $(I + EF)^{-1}$ and $(I + FE)^{-1}$ both exist, then

$$\begin{aligned}(I + EF)^{-1}E &= E(I + FE)^{-1} \\ E(I + FE) &= (I + EF)E \\ E + EFE &= E + EFE\end{aligned} \tag{5.28}$$

Now, to derive the second essential relation, start with Eqs. (1.6) and (1.8), or Eq. (1.7) directly, along with Eq. (5.27) to show

$$\delta_j \underline{e} = -P\left(K\delta_j \underline{v} - KC\delta_j \underline{d}\right) - \delta_j \underline{d} \tag{5.29}$$

Again, following some standard arithmetic to expand the product, collecting like terms, and then plug in for the second instance of $K$, it can be seen that Eq. (5.29) can be rewritten as

$$\delta_j \underline{e} = -PK\delta_j \underline{v} - [I - P(I + CP)^{-1}C]\delta_j \underline{d} \tag{5.30}$$

In order to simplify Eq. (5.30) further, another proof is necessary. This proof will make use of the prior proof, given in Eq. (5.28) with substitutions $E = C$ and $F = P$, to demonstrate that the last term of Eq. (5.30) can be expressed as

$$\begin{aligned} I - P[(I + CP)^{-1}C] &= (I + PC)^{-1} \\ I - P[C(I + PC)^{-1}] &= (I + PC)^{-1} \\ I(I + PC) - PC &= (I + PC)^{-1}(I + PC) \\ I(I + PC) - PC &= I \\ I + PC &= I + PC \end{aligned} \tag{5.31}$$

Through the use of Eq. (5.31), it should be clear that Eq. (5.30) can be further simplified into its final desired form

$$\delta_j \underline{e} = -PK\delta_j \underline{v} - (I + PC)^{-1}\delta_j \underline{d} \tag{5.32}$$

This equation gives the relation between $\delta_j \underline{v}$ and $\delta_j \underline{e}$, which is necessary to establish the second fundamental relationship for optimization of the control actions produced by closed-loop configuration one. The two fundamental relationships, analogous to Eqs. (5.3) and (5.4), can now be found using Eqs. (5.27) and (5.32) as follows

$$\partial \delta_j \underline{u} \Big/ \partial \delta_j \underline{v} = K \tag{5.33}$$

$$\partial \delta_j \underline{e} \Big/ \partial \delta_j \underline{v} = -PK = -P_E \tag{5.34}$$

The matrix $P_E$ in Eq. (5.34) can be considered the effective repetition-domain plant for this configuration, which is closely related to the general plant as given in Eq. (5.4) from the previous section. From a comparison of Eqs. (5.33) and (5.34) to Eqs. (5.3) and (5.4), it is apparent that the only difference between these two sets of fundamental

relationships is $K$, which explicitly reflects the influence of the feedback controller in the repetition-domain.

With the formulation of the input signal and fundamental relationships, it is now appropriate to introduce the various cost functions which can be used for optimization. The first two cost functions which can be considered are the same ones used for the basic configuration, $J_1$ and $J_2$ as presented in Eq. (5.1) and Eq. (5.2) respectively. The final two cost functions are

$$J_3 = \left[\underline{e}_j\right]^T Q_e \underline{e}_j + \left[\underline{v}_j\right]^T Q_u \underline{v}_j \tag{5.35}$$

$$J_4 = \left[\underline{e}_j\right]^T Q_e \underline{e}_j + \left[\delta_j \underline{v}\right]^T Q_u \delta_j \underline{v} \tag{5.36}$$

By comparing $J_1$ to $J_3$ and $J_2$ to $J_4$, it can be seen that the only difference between these cost functions is use of the plant input or learning signal as the control variable. However, when trying to find the optimal controller, the variable used to minimize all of these cost functions should be $\delta_j \underline{v}$, the update to the learning signal. So, as with the basic open-loop configuration, $J_1$ will also be the first cost function optimized here. To do this, start with Eqs. (5.5) and take the partial derivative with respect to $\delta_j \underline{v}$, through appropriate use of Eqs. (5.33) and (5.34), then set to zero

$$\partial J_1 \Big/ \partial \delta_j \underline{v} = 2{P_E}^T Q_e P_E \delta_j \underline{v} - 2{P_E}^T Q_e \underline{e}_{j-1} + 2K^T Q_u K \delta_j \underline{v} + 2K^T Q_u \underline{u}_{j-1} = 0 \tag{5.37}$$

Through use of the definition given in Eq. (5.34) for effective plant $P_E$, it is possible to simplify Eq. (5.37) into a more suitable form for comparison with the other designs

$$\begin{aligned} &2[PK]^T Q_e [PK]\delta_j \underline{v} - 2[PK]^T Q_e \underline{e}_{j-1} + 2K^T Q_u K \delta_j \underline{v} + 2K^T Q_u \underline{u}_{j-1} = 0 \\ &2K^T (P^T Q_e P) K \delta_j \underline{v} - 2K^T P^T Q_e \underline{e}_{j-1} + 2K^T Q_u K \delta_j \underline{v} + 2K^T Q_u \underline{u}_{j-1} = \\ &2K^T \left[ (P^T Q_e P) K \delta_j \underline{v} - P^T Q_e \underline{e}_{j-1} + Q_u K \delta_j \underline{v} + Q_u \underline{u}_{j-1} \right] = \end{aligned} \tag{5.38}$$

Now to determine the optimal update to the learning signal with respect to $J_1$ for closed-loop configuration one, some additional arithmetic is required to eliminate common terms, collect like terms, and solve

$$\begin{aligned} 0 &= (P^T Q_e P) K \delta_j \underline{v} - P^T Q_e \underline{e}_{j-1} + Q_u K \delta_j \underline{v} + Q_u \underline{u}_{j-1} \\ (P^T Q_e P + Q_u) K \delta_j \underline{v} &= P^T Q_e \underline{e}_{j-1} - Q_u \underline{u}_{j-1} \\ \delta_j \underline{v} &= K^{-1} (P^T Q_e P + Q_u)^{-1} \left( P^T Q_e \underline{e}_{j-1} - Q_u \underline{u}_{j-1} \right) \end{aligned} \tag{5.39}$$

By using the same representation given in Eqs. (5.12) and (5.13), the final result of Eq. (5.39) can be rewritten in an analogous form

$$\begin{aligned} \delta_j \underline{v} &= K^{-1} (P^T Q_e P + Q_u)^{-1} P^T Q_e \underline{e}_{j-1} - K^{-1} (P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} \\ &= K^{-1} L^* \underline{e}_{j-1} - K^{-1} (P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} \end{aligned} \tag{5.40}$$

This is the recommended optimal rule, with respect to $J_1$, for updating the learning signal. The optimal updates to the learning signal for closed-loop configuration one, with respect to $J_2$, can be found by taking the partial derivative of Eq. (5.6) with respect to $\delta_j \underline{v}$ as shown

$$\partial J_2 \Big/ \partial \delta_j \underline{v} = 2 {P_E}^T Q_e P_E \delta_j \underline{v} - 2 {P_E}^T Q_e \underline{e}_{j-1} + 2 K^T Q_u K \delta_j \underline{v} = 0 \tag{5.41}$$

Once again, these partial derivatives make proper use of Eqs. (5.33) and (5.34) as required. From a comparison of Eqs. (5.37) and (5.41), it should be clear that the only difference between the final results will be the effect of $\underline{u}_{j-1}$, which is present in the former equation but absent in the latter. Given this, the final form of Eq. (5.41) can be written as

$$\begin{aligned} \delta_j \underline{v} &= K^{-1} (P^T Q_e P + Q_u)^{-1} P^T Q_e \underline{e}_{j-1} \\ &= K^{-1} L^* \underline{e}_{j-1} \end{aligned} \tag{5.42}$$

Both optimal control designs given in Eqs. (5.40) and (5.42) made use of cost functions where the control variable was related to the plant input directly. Now, the other two cost functions, where the control variable is directly related to the learning signal, are considered. As with $J_1$ and $J_2$, it is useful to rewrite $J_3$ and $J_4$ to explicitly show the critical parameters of interest. So the difference operator is again used, as in Eqs. (5.5) and (5.6), to replace the current cycle error $\underline{e}_j$ and the current cycle learning signal $\underline{v}_j$ in Eqs. (5.35) and (5.36) as follows

$$J_3 = \left[\delta_j \underline{e} + \underline{e}_{j-1}\right]^T Q_e \left[\delta_j \underline{e} + \underline{e}_{j-1}\right] + \left[\delta_j \underline{v} + \underline{v}_{j-1}\right]^T Q_u \left[\delta_j \underline{v} + \underline{v}_{j-1}\right] \tag{5.43}$$

$$J_4 = \left[\delta_j \underline{e} + \underline{e}_{j-1}\right]^T Q_e \left[\delta_j \underline{e} + \underline{e}_{j-1}\right] + \left[\delta_j \underline{v}\right]^T Q_u \delta_j \underline{v} \tag{5.44}$$

Now, the cost functions given in Eqs. (5.43) and (5.44) are ready to be optimized by taking the partial derivative with respect to $\delta_j \underline{v}$, through appropriate use of Eq. (5.34), and setting to zero

$$\partial J_3 \Big/ \partial \delta_j \underline{v} = 2{P_E}^T Q_e P_E \delta_j \underline{v} - 2{P_E}^T Q_e \underline{e}_{j-1} + 2Q_u \delta_j \underline{v} + 2Q_u \underline{v}_{j-1} = 0 \tag{5.45}$$

$$\partial J_4 \Big/ \partial \delta_j \underline{v} = 2{P_E}^T Q_e P_E \delta_j \underline{v} - 2{P_E}^T Q_e \underline{e}_{j-1} + 2Q_u \delta_j \underline{v} = 0 \tag{5.46}$$

Similar to $J_1$ and $J_2$, the minimizations of $J_3$ and $J_4$ are also identical except for the last term in Eq. (5.45), which is a direct result of penalizing the current cycle learning signal, as opposed to the iterative updates to the learning signal. Here, closed-loop configuration one will produce optimal control with respect to $J_3$ if

$$\delta_j \underline{v} = \left({P_E}^T Q_e P_E + Q_u\right)^{-1} {P_E}^T Q_e \underline{e}_{j-1} - \left({P_E}^T Q_e P_E + Q_u\right)^{-1} Q_u \underline{v}_{j-1} \tag{5.47}$$

With respect to $J_3$, Eq. (5.47) is the optimal update rule for the learning signal. Now, for $J_4$, the closed-loop configuration one will produce optimal control if the solution to Eq. (5.46) is used

$$\delta_j \underline{v} = \left({P_E}^T Q_e P_E + Q_u\right)^{-1} {P_E}^T Q_e \underline{e}_{j-1} \tag{5.48}$$

All of the optimal control designs for this configuration, given in Eqs. (5.40), (5.42), (5.47), and (5.48), apply both current and previous cycle feedback. Before trying to predict the overall performance, some notable connections between the first set of cost functions, $J_1$ and $J_2$, and the second set, $J_3$ and $J_4$, should be stated. By comparing Eqs. (5.40) and (5.47), it should be apparent that the main difference between these two designs is Eq. (5.40) operates on the original plant $P$ and Eq. (5.47) operates on the effective plant $P_E$ to generate learning signal updates. This can also be seen when comparing Eqs. (5.42) and (5.48), where the results would be identical if the $K$ term in Eq. (5.42) was neglected and the original plant and effect plant were equivalent. This is useful to know when trying to decide which cost function to use and predict overall performance. However, these designs alone do not provide the necessary insight into the overall performance limitations to make educated design decisions, since they do not formulate the resulting error propagation.

### 5.2.2: Error Propagation of Optimal Closed-Loop Configuration One Controller Designs

To thoroughly assess how the incorporation of current cycle feedback into the control scheme using closed-loop configuration one, it is necessary to determine the error performance using an optimized controller. To do this for the optimal controller resulting from $J_1$, start with Eq. (5.32) and plug in for the optimal updates to the learning signal using Eq. (5.40) as shown

$$\delta_j \underline{e} = -PK\left[K^{-1}L^* \underline{e}_{j-1} - K^{-1}(P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1}\right] - (I + PC)^{-1} \delta_j \underline{d} \tag{5.49}$$

After expansion of the difference operator on error and some expansion of terms, it is clear to see that Eq. (5.49) can be rewritten as

$$\left(\underline{e}_j - \underline{e}_{j-1}\right) = -PL^*\underline{e}_{j-1} + P(P^TQ_eP + Q_u)^{-1}Q_u\underline{u}_{j-1} - (I + PC)^{-1}\delta_j\underline{d} \tag{5.50}$$

Some additional arithmetic to collect like terms will produce the desired result

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} + P(P^TQ_eP + Q_u)^{-1}Q_u\underline{u}_{j-1} - (I + PC)^{-1}\delta_j\underline{d} \tag{5.51}$$

Based on Eq. (5.51), the optimal controller produced by minimizing $J_1$ can never result in error free control. Even if inverse control using previous cycle feedback, by setting $L^* = P^{-1}$, and high gain current cycle feedback, by setting $C = \{C \mid PC \gg I\}$, was attempted, zero error is not attainable because $\underline{u}_{j-1}$ is non-zero in general. Now, the same derivation is done for the optimal controller resulting from $J_2$, starting again with Eq. (5.32) and plugging in Eq. (5.42) as follows

$$\delta_j\underline{e} = -PK\left(K^{-1}L^*\underline{e}_{j-1}\right) - (I + PC)^{-1}\delta_j\underline{d} \tag{5.52}$$

Following some simple standard arithmetic to collect like terms and rearrange the equation, it is apparent that Eq. (5.52) simplifies to

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - (I + PC)^{-1}\delta_j\underline{d} \tag{5.53}$$

The performance predicted by Eq. (5.53) shows that closed-loop configuration one optimized with respect to $J_2$ does theoretically have the potential to produce perfect, error free control. However, in order to achieve this potential, high gain control using current cycle feedback is necessary, which is not recommended in general. Next, the same steps are followed for the optimal controller resulting from $J_3$, similarly starting with Eq. (5.32) and plugging in Eq. (5.47) in this derivation

$$\begin{aligned}\delta_j\underline{e} = -PK\left[\left(P_E{}^TQ_eP_E + Q_u\right)^{-1}P_E{}^TQ_e\underline{e}_{j-1} - \left(P_E{}^TQ_eP_E + Q_u\right)^{-1}Q_u\underline{v}_{j-1}\right] \\ - (I + PC)^{-1}\delta_j\underline{d}\end{aligned} \tag{5.54}$$

Again some simple standard arithmetic must be done to collect like terms and rearrange the equation. Following this, Eq. (5.54) can be rewritten in its final form

$$\underline{e}_j = \left[I - PK\left(P_E{}^T Q_e P_E + Q_u\right)^{-1} P_E{}^T Q_e\right]\underline{e}_{j-1} + PK\left(P_E{}^T Q_e P_E + Q_u\right)^{-1} Q_u \underline{v}_{j-1} - (I + PC)^{-1}\delta_j \underline{d} \tag{5.55}$$

Similar to Eq. (5.51), here Eq. (5.55) shows the optimal controller produced by minimizing $J_3$ can never result in error free control. This is true even if $K\left(P_E{}^T Q_e P_E + Q_u\right)^{-1} P_E{}^T Q_e \to P^{-1}$ and $C = \{C \mid PC \gg I\}$ because $\underline{v}_{j-1}$ is non-zero in general. The final error performance analyzed for close-loop configuration one is for the optimal controller resulting from minimizing $J_4$ and this is done starting, once again, with Eq. (5.32) and then plugging in Eq. (5.48) as shown

$$\delta_j \underline{e} = -PK\left[\left(P_E{}^T Q_e P_E + Q_u\right)^{-1} P_E{}^T Q_e \underline{e}_{j-1}\right] - (I + PC)^{-1}\delta_j \underline{d} \tag{5.56}$$

Following the same simple standard arithmetic to collect like terms and rearrange the equation, it can be shown that Eq. (5.56) simplifies to

$$\underline{e}_j = \left[I - PK\left(P_E{}^T Q_e P_E + Q_u\right)^{-1} P_E{}^T Q_e\right]\underline{e}_{j-1} - (I + PC)^{-1}\delta_j \underline{d} \tag{5.57}$$

As with Eq. (5.53), it should be apparent from Eq. (5.57) that the error performance of the optimal controller with respect to $J_4$ has the potential to be perfect. However, to realize this possibility, high gain current cycle feedback with $C = \{C \mid PC \gg I\}$ is necessary, but is typically not recommended for practice. Before moving on to consider another configuration, it is worthwhile to provide an example of how the preceding results using closed-loop configuration one can be modified and applied to a variant setup incorporating the learning signal inside the feedback loop. Consider a configuration where the pickoff point for the input feedback matrix $R$, as shown in Figure 5-2, was moved such that the plant input $\underline{u}_j$ was feedback instead of $\underline{u}_{FB}$, as Eq. (5.22) requires. This setup can easily be evaluated using the same results as obtained above

and simply replacing $\underline{v}_j$ with $(I-R)^{-1}\underline{v}_j$ as appropriate. Though other such variant setups to input the learning signal inside the feedback loop exist, Figure 5-2 shows the recommended configuration for evaluation purposes, since the other setups can usually just be considered special cases of this configuration.

## 5.3: Closed-loop FBC and ILC Configuration Two

The second fundamentally distinct way in which a feedback and learning combination controller can be configured is with the learning signal incorporated outside the feedback loop. In this section, a general configuration of this type will be evaluated. Though there are different ways to input the learning signal in this manner, Figure 5-3 shows the configuration evaluated in this investigation. Similar to closed-loop configuration one, the results established in this section can also be appropriately modified and applied to a variant setup incorporating the learning signal outside the feedback loop, if desired.

### 5.3.1: Optimal Closed-Loop Combination Controller Design for Configuration Two

As with the basic open-loop configuration and closed-loop configuration one, cost functions will similarly be utilized here to optimize performance of the control implementation. However, before the cost functions are introduced, the input signal to be optimized for this configuration must be properly formulated. Similar to closed-loop configuration one in Figure 5-2, the plant input $\underline{u}_j$ and learning control signal $\underline{v}_j$ in Figure 5-3 are also not the same. For closed-loop FBC and ILC configuration two, which will be referred to simply as closed-loop configuration two henceforth, the plant input is

$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j^* \tag{5.58}$$

Here, $\underline{e}_j^*$ is the input to the feedback controller, which can be defined with respect to the configuration as

$$\begin{aligned} \underline{e}_j^* &= \underline{y}^* - \underline{y}_j + \underline{v}_j \\ &= \underline{e}_j + \underline{v}_j \end{aligned} \tag{5.59}$$

Using Eq. (5.59) it is possible to define the plant input given in Eq. (5.58) in terms of the current cycle tracking error $\underline{e}_j$, which is desired since that is what the controller aims to minimize

$$\underline{u}_j = R\underline{u}_j + G\left(\underline{e}_j + \underline{v}_j\right) \tag{5.60}$$

Following some standard arithmetic, to expand Eq. (5.60), collect like terms, and simplify, the final expression is obtained

$$\begin{aligned} \underline{u}_j &= (I - R)^{-1}G\underline{e}_j + (I - R)^{-1}G\underline{v}_j \\ &= C\underline{e}_j + C\underline{v}_j \end{aligned} \tag{5.61}$$

Note the same matrix $C$ introduced in Eq. (5.24) is similarly used in place of $(I - R)^{-1}G$ purely for mathematical convenience in subsequent derivations. It is also reiterated that the underlying assumption of the existence of $(I - R)^{-1}$ is repeated here, which is again valid since $R$ is part of the FBC and can typically be designed to validate the assumption. It should be apparent from inspection of Eqs. (5.59) and (5.61) that with this type of configuration, the ILC signal can be thought to supplement the reference signal $\underline{y}^*$, effectively modifying the command to the existing feedback control system. As with closed-loop configuration one, before considering the same cost functions for controller optimization, analogous fundamental relationships relating $\delta_j\underline{v}$ to $\delta_j\underline{u}$ and $\delta_j\underline{e}$ must be established for close-loop configuration two. These relationships will parallel those specified in Eqs. (5.33) and (5.34) for closed-loop configuration one. In order to derive these essential relations, start with Eq. (5.61) and apply the difference operator $\delta_j$ as shown

$$\delta_j \underline{u} = C\delta_j \underline{e} + C\delta_j \underline{v} \tag{5.62}$$

As with closed-loop configuration one, using Eq. (1.7), it is possible to rewrite Eq. (5.62) by substituting in for the unknown iterative changes to the error

$$\delta_j \underline{u} = C\left(-P\delta_j \underline{u} - \delta_j \underline{d}\right) + C\delta_j \underline{v} \tag{5.63}$$

Again, after some simple arithmetic to collect like terms and rearrange the equation, it can be seen that Eq. (5.63) can be written as

$$\begin{aligned} \delta_j \underline{u} &= (I + CP)^{-1} C\delta_j \underline{v} - (I + CP)^{-1} C\delta_j \underline{d} \\ &= KC\delta_j \underline{v} - KC\delta_j \underline{d} \end{aligned} \tag{5.64}$$

The same matrix $K$ first used in Eq. (5.27) is similarly utilized here to represent $(I + CP)^{-1}$ for the same reasons it was initially introduced. Also, $K$ is again assumed to exist here, which is acceptable since $C$ equally represents the FBC in this derivation and can be designed to satisfy the assumption. The equation given in Eq. (5.64) defines the relation between $\delta_j \underline{v}$ and $\delta_j \underline{u}$, which is necessary to establish the first fundamental relationship for optimization of the control actions produced by closed-loop configuration two. Now, to derive the second essential relation, start with Eqs. (1.6) and (1.8), or Eq. (1.7) directly, along with Eq. (5.64) to show

$$\delta_j \underline{e} = -P\left(KC\delta_j \underline{v} - KC\delta_j \underline{d}\right) - \delta_j \underline{d} \tag{5.65}$$

Through some arithmetic manipulation to expand the product, collecting like terms, and then plug in for the second instance of $K$, as done in Eq. (5.30), it is clear that Eq. (5.65) can be rewritten as

$$\delta_j \underline{e} = -PKC\delta_j \underline{v} - [I - P(I + CP)^{-1} C]\delta_j \underline{d} \tag{5.66}$$

Through use of the proof presented in Eq. (5.31), it can be shown that Eq. (5.66) also simplifies, just like Eq. (5.30), into the following concise form

$$\delta_j \underline{e} = -PKC\delta_j \underline{v} - (I + PC)^{-1}\delta_j \underline{d} \tag{5.67}$$

This equation gives the relation between $\delta_j \underline{v}$ and $\delta_j \underline{e}$, analogous to Eq. (5.32), which is necessary to establish the second fundamental relationship for optimization of the control actions produced by closed-loop configuration two. The two fundamental relationships, parallel to Eqs. (5.33) and (5.34), can now be found using Eqs. (5.64) and (5.67) as shown

$$\partial \delta_j \underline{u} \Big/ \partial \delta_j \underline{v} = KC = K_E \tag{5.68}$$

$$\partial \delta_j \underline{e} \Big/ \partial \delta_j \underline{v} = -PKC = -PK_E \tag{5.69}$$

The matrix $K_E$ in Eqs. (5.68) and (5.69) can be considered the effective RFBC implementation for this configuration, closely related to the analogous feedback controller given in Eq. (5.33) from the previous section. A quick comparison of Eqs. (5.33) and (5.34) to Eqs. (5.68) and (5.69) shows that the only difference between these two sets of fundamental relationships is that $C$ is explicitly present in the latter set. This directly reflects the effect of using a configuration that incorporates the learning signal outside the feedback loop, because now the feedback controller $C$ has direct influence on reception dynamics.

With the formulation of the fundamental relationships for the control variable and the general input signal, the same cost functions considered for closed-loop configuration one can again be used for optimization here. When trying to find the optimal controller, as done previously, the variable used to minimize the cost functions is $\delta_j \underline{v}$, the update to the learning signal. Once again $J_1$ will be the first cost function optimized here. To do this, start with Eqs. (5.5) and take the partial derivative with respect to $\delta_j \underline{v}$, in the same

way as for closed-loop configuration one but this time through appropriate use of Eqs. (5.68) and (5.69), as shown

$$\partial J_1 \Big/ \partial \delta_j \underline{v} = 2[PK_E]^T Q_e [PK_E] \delta_j \underline{v} - 2[PK_E]^T Q_e \underline{e}_{j-1} + 2{K_E}^T Q_u K_E \delta_j \underline{v} + 2{K_E}^T Q_u \underline{u}_{j-1} \quad (5.70)$$

Through use of the definition given in Eq. (5.68) for effective feedback $K_E$, it is possible to simplify Eq. (5.70) to facilitate comparison with the other designs

$$\begin{aligned} 2[PK_E]^T Q_e [PK_E] \delta_j \underline{v} - 2[PK_E]^T Q_e \underline{e}_{j-1} + 2{K_E}^T Q_u K_E \delta_j \underline{v} + 2{K_E}^T Q_u \underline{u}_{j-1} = 0 \\ 2{K_E}^T \left[ (P^T Q_e P) K_E \delta_j \underline{v} - P^T Q_e \underline{e}_{j-1} + Q_u K_E \delta_j \underline{v} + 2 Q_u \underline{u}_{j-1} \right] = \\ 2[KC]^T \left[ (P^T Q_e P + Q_u)[KC] \delta_j \underline{v} - P^T Q_e \underline{e}_{j-1} + Q_u \underline{u}_{j-1} \right] = \end{aligned} \quad (5.71)$$

In order determine the optimal update to the learning signal with respect to $J_1$ for closed-loop configuration two, some additional restrictions must first be placed on the structure of $C$, which represents $(I - R)^{-1} G$ or the overall time-domain FBC gain matrix. Recall that $C$ is already restricted by $K$, where $K$ is used to represent $(I + CP)^{-1}$ or the overall RFBC gain matrix, because it was originally assumed that $C$ was designed to ensure $K$ existed. From inspection of Eq. (5.71), it should be apparent that $C^{-1}$ must also exist in order to simplify the equation further. This necessitates the existence of $G^{-1}$, which restricts the structure of $G$, and consequently $C$, to be of lower triangular form and also requires both matrices to be non-singular. If the desired error feedback is time-invariant, this means that $G$ must be a lower triangular Toeplitz matrix. Similarly, if the input feedback is also time-invariant, this implies $R$ can be a lower triangular or lower sub-triangular Toeplitz matrix. In both cases, $C$ is restricted to be a lower triangular Toeplitz matrix. Such a restriction results in an overall time-domain FBC implementation that is non-causal by one time-step. While strict causality must be followed for FBC, since it utilizes real-time current cycle feedback, earlier work established design approaches to

transform FBC that is non-causal by one-time step into strictly causal, implementable feedback and learning combination controllers. It can therefore be assumed that restricting $G$, and by association $C$, to be of lower triangular structure is acceptable. It is further assumed that these gain matrices were both designed to be non-singular when used in closed-loop configuration two. Now with some additional arithmetic to eliminate common terms, collect like terms, and solve, the following is obtained

$$\begin{aligned} 0 &= (P^T Q_e P + Q_u)[KC]\delta_j \underline{v} - P^T Q_e \underline{e}_{j-1} + Q_u \underline{u}_{j-1} \\ (P^T Q_e P + Q_u)KC\delta_j \underline{v} &= P^T Q_e \underline{e}_{j-1} - Q_u \underline{u}_{j-1} \\ \delta_j \underline{v} &= [C^{-1}K^{-1}](P^T Q_e P + Q_u)^{-1}\left(P^T Q_e \underline{e}_{j-1} - Q_u \underline{u}_{j-1}\right) \end{aligned} \tag{5.72}$$

By using the same representation given in Eqs. (5.12) and (5.40), the final result of Eq. (5.72) can be rewritten in a similar form to facilitate comparison

$$\begin{aligned} \delta_j \underline{v} &= [C^{-1}K^{-1}](P^T Q_e P + Q_u)^{-1} P^T Q_e \underline{e}_{j-1} - [C^{-1}K^{-1}](P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} \\ &= [C^{-1}K^{-1}]L^* \underline{e}_{j-1} - [C^{-1}K^{-1}](P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} \\ &= {K_E}^{-1} L^* \underline{e}_{j-1} - {K_E}^{-1}(P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} \end{aligned} \tag{5.73}$$

Notice $K_E = KC$ was substituted back into the final expression. This is the recommended optimal rule, with respect to $J_1$, for updating the learning signal. The optimal updates to the learning signal for closed-loop configuration two, with respect to $J_2$, can be found by taking the partial derivative of Eq. (5.6) with respect to $\delta_j \underline{v}$, again through appropriate use of Eqs. (5.68) and (5.69), as follows

$$\partial J_2 \Big/ \partial \delta_j \underline{v} = 2[PK_E]^T Q_e [PK_E]\delta_j \underline{v} - 2[PK_E]^T Q_e \underline{e}_{j-1} + 2{K_E}^T Q_u K_E \delta_j \underline{v} = 0 \tag{5.74}$$

The final version of Eq. (5.74) should resemble Eqs. (5.13) and (5.42) closely, in that all three minimizations should depend only on $\underline{e}_{j-1}$ and $L^*$ should be explicitly present. So, Eq. (5.74) can be modified accordingly

$$\begin{aligned}\delta_j\underline{v} &= [C^{-1}K^{-1}](P^TQ_eP + Q_u)^{-1}P^TQ_e\underline{e}_{j-1} \\ &= {K_E}^{-1}L^*\underline{e}_{j-1}\end{aligned} \tag{5.75}$$

Next, the optimal controller for closed-loop configuration two with respect to $J_3$ is developed. This is done using the same steps followed for $J_1$ and $J_2$, but starting with Eq. (5.43) to take the partial derivative with respect to $\delta_j\underline{v}$ and setting to zero

$$\partial J_3 \Big/ \partial\delta_j\underline{v} = 2[PK_E]^TQ_e[PK_E]\delta_j\underline{v} - 2[PK_E]^TQ_e\underline{e}_{j-1} + 2Q_u\delta_j\underline{v} + 2Q_u\underline{v}_{j-1} = 0 \tag{5.76}$$

It should be noted that the partial derivative to obtain Eq. (5.76) made use of Eq. (5.69) as appropriate. For closed-loop configuration two, optimal control with respect to $J_3$ will result if

$$\delta_j\underline{v} = ([PK_E]^TQ_e[PK_E] + Q_u)^{-1}[PK_E]^TQ_e\underline{e}_{j-1} - ([PK_E]^TQ_e[PK_E] + Q_u)^{-1}Q_u\underline{v}_{j-1} \tag{5.77}$$

With respect to $J_3$, Eq. (5.77) is the optimal update rule for the learning signal. From inspection of this update rule and comparison to Eq. (5.47), it is apparent that the difference between these two rules is the overall RFBC implementation. For Eq. (5.47), the update depends on the general RFBC gain matrix $K$ through effective plant $P_E$, while in Eq. (5.77) the update depends on $K_E$, the effective RFBC gain matrix. Finally, for $J_4$, the closed-loop configuration two optimal controller is found starting with Eq. (5.44) and going through the same steps as before

$$\partial J_4 \Big/ \partial\delta_j\underline{v} = 2[PK_E]^TQ_e[PK_E]\delta_j\underline{v} - 2[PK_E]^TQ_e\underline{e}_{j-1} + 2Q_u\delta_j\underline{v} = 0 \tag{5.78}$$

Again following some standard arithmetic, to eliminate common terms, collect like terms, and rearrange the equation into a form consistent with previous results, yields

$$\delta_j\underline{v} = ([PK_E]^TQ_e[PK_E] + Q_u)^{-1}[PK_E]^TQ_e\underline{e}_{j-1} \tag{5.79}$$

This is the optimal update rule, with respect to $J_4$, for the learning signal when implementing closed-loop configuration two. From a comparison of Eq. (5.48) to Eq. (5.79), it should be clear that the differences between these two rules again come down to the repetition-domain feedback controller. So for $J_3$ and $J_4$, an understanding of the differences between $K$ and $K_E$ can provide some insight into the expected differences in performance between closed-loop configurations one and two. This is useful to know when trying to decide which cost function, and configuration, to use and quickly attempt to predict overall performance. However, as with closed-loop configuration one, these designs alone are insufficient to fully predict the overall performance limitations, since they do not directly describe the expected error.

### 5.3.2: Error Propagation of Optimal Closed-Loop Configuration Two Controller Designs

To facilitate the evaluation of closed-loop configuration two and the overall impact of incorporating the learning signal with such a setup, outside the feedback loop, the error performance using optimized controllers must be established. For the optimal controller resulting from $J_1$, this is done starting with Eq. (5.67) and plugging in for the optimal updates to the learning signal using Eq. (5.73) as shown

$$\delta_j \underline{e} = -PK_E\left[{K_E}^{-1}L^*\underline{e}_{j-1} - {K_E}^{-1}(P^TQ_eP + Q_u)^{-1}Q_u\underline{u}_{j-1}\right] - (I + PC)^{-1}\delta_j\underline{d} \qquad (5.80)$$

It should be noted that $K_E = KC$ was substituted into Eq. (5.67) when used here. After expansion of the difference operator on error and multiplying through to cancel like terms, it can be seen that Eq. (5.80) can be rewritten as

$$\left(\underline{e}_j - \underline{e}_{j-1}\right) = -PL^*\underline{e}_{j-1} + P(P^TQ_eP + Q_u)^{-1}Q_u\underline{u}_{j-1} - (I + PC)^{-1}\delta_j\underline{d} \qquad (5.81)$$

Following some simple arithmetic manipulation to collect like terms and rearrange the equation, the final result is obtained

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} + P(P^T Q_e P + Q_u)^{-1} Q_u \underline{u}_{j-1} - (I + PC)^{-1}\delta_j \underline{d} \tag{5.82}$$

A quick comparison of Eq. (5.82) to Eq. (5.51) reveals the error performance using $J_1$ is identical for both closed-loop configurations. This means the optimal controller produced by minimizing $J_1$ can never result in error free control with either configuration, due to the presence of $\underline{u}_{j-1}$ in both error propagation expressions. It also suggests that the optimization process, using $J_1$ in the manner done here, is independent of the configurations. Now, following the same steps for the optimal controller obtained by minimizing $J_2$, similarly starting with Eq. (5.67) and this time plugging in Eq. (5.75), results in

$$\delta_j \underline{e} = -PK_E\left({K_E}^{-1} L^* \underline{e}_{j-1}\right) - (I + PC)^{-1}\delta_j \underline{d} \tag{5.83}$$

After going through similar arithmetic to collect like terms, it can be seen that Eq. (5.83) will simplify to

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - (I + PC)^{-1}\delta_j \underline{d} \tag{5.84}$$

As with the performance using $J_1$, the error performance achieved using $J_2$ for closed-loop configuration one, as described by Eq. (5.53), is identical to the error performance predicted here by Eq. (5.84) for closed-loop configuration two. This shows that both closed-loop configurations, when optimized with respect to $J_2$, have the theoretical potential to produce error free control. It also similarly suggests that the optimization process, using $J_2$ in the manner done here, is independent of the configurations. Next, the same derivation is done for the optimal controller resulting from $J_3$, starting again with Eq. (5.67) and plugging in Eq. (5.77) as follows

$$\begin{aligned}\delta_j \underline{e} = -PK_E\big[&([PK_E]^T Q_e [PK_E] + Q_u)^{-1}[PK_E]^T Q_e \underline{e}_{j-1} \\ &- ([PK_E]^T Q_e [PK_E] + Q_u)^{-1} Q_u \underline{v}_{j-1}\big] - (I + PC)^{-1}\delta_j \underline{d}\end{aligned} \tag{5.85}$$

Following some necessary simple arithmetic to collect like terms and rearrange the equation, it is apparent that Eq. (5.85) simplifies to

$$\begin{aligned}\underline{e}_j = {} & [I - PK_E([PK_E]^T Q_e[PK_E] + Q_u)^{-1}[PK_E]^T Q_e]\underline{e}_{j-1} \\ & + PK_E([PK_E]^T Q_e[PK_E] + Q_u)^{-1} Q_u \underline{v}_{j-1} - (I + PC)^{-1}\delta_j \underline{d}\end{aligned} \tag{5.86}$$

It can be seen by comparing Eq. (5.55) to Eq. (5.86) that the difference between these two error propagation expressions is again related to the repetition-domain feedback controller. Recall that $P_E = PK$, as defined in Eq. (5.34), and $K_E = KC$, as originally defined in Eq. (5.68), so it is clear that if not for the difference between $K$ and $K_E$, these error performance expressions would be identical. Therefore, as with $J_1$, the optimal controller produced by minimizing $J_3$ is also not capable of producing error free control with either configuration, due to the presence of $\underline{v}_{j-1}$ in both error propagation expressions. Finally, the last error performance analysis remaining for close-loop configuration two is using the optimal controller resulting from minimizing $J_4$ and this, as before, starts with Eq. (5.67) and then plugging in Eq. (5.79) as shown

$$\delta_j \underline{e} = -PK_E\left[([PK_E]^T Q_e[PK_E] + Q_u)^{-1}[PK_E]^T Q_e \underline{e}_{j-1}\right] - (I + PC)^{-1}\delta_j \underline{d} \tag{5.87}$$

As with all the prior error propagation derivations, additional arithmetic must be done to collect like terms and rearrange the equation. Following this, Eq. (5.87) is finalized as

$$\underline{e}_j = [I - PK_E([PK_E]^T Q_e[PK_E] + Q_u)^{-1}[PK_E]^T Q_e]\underline{e}_{j-1} - (I + PC)^{-1}\delta_j \underline{d} \tag{5.88}$$

As with Eq. (5.57), it should be apparent from Eq. (5.88) that the error performance of the optimal controller with respect to $J_4$ has the potential to produce zero error control. In fact, just like the controllers optimized with respect to $J_3$, these two error performance expressions would be identical if not for the differences in the RFBC implementation. Now that the expected error has been fully described and performance limitations have

been established for all the different combinations, a detailed analytical comparison is conducted to complete the overall assessment of the three configurations considered.

### 5.4: Analytical Comparison of the Configurations

To begin the analytical comparison, assume that the physical plant to be controlled can be modeled using a $P$ matrix, which is known to perfectly model the physical plant. Through use of this $P$ model, it is additionally assumed that the output and output error of the physical plant, and as necessary its operating environment, can be fully described using Eqs. (1.2) - Eq. (1.8) with complete accuracy. This is clearly not very realistic or practical, but for the purposes of a hypothetical comparison, such an ideal situation can be assumed. Using the error propagation given in Eq. (5.16), for the basic open-loop configuration in this ideal scenario, the following transfer matrices result if equal weighting, or $Q_e = Q_u = 0.5I$, is applied

$$\begin{aligned} TF1a &= I - PL^* = I - P(P^T Q_e P + Q_u)^{-1} P^T Q_e \\ &= I - P(P^T P + I)^{-1} P^T \end{aligned} \tag{5.89}$$

$$\begin{aligned} TF1b &= P(P^T Q_e P + Q_u)^{-1} Q_u \\ &= P(P^T P + I)^{-1} \end{aligned} \tag{5.90}$$

To evaluate the closed-loop configurations for the ideal situation, the following current cycle feedback control implementation can be considered

$$R = \frac{2\alpha - 1}{\alpha} I \tag{5.91}$$

$$G = P^{-1} \tag{5.92}$$

To consider this type of theoretical controller, $P$ must be non-singular and well-conditioned with a stable inverse. The current cycle input gain matrix defined in Eq.

(5.91) and current cycle error feedback gain matrix defined in Eq. (5.92) can now be used to generate

$$C = (I - R)^{-1}G = \frac{\alpha}{1-\alpha}P^{-1} \tag{5.93}$$

$$K = (I + CP)^{-1} = (1 - \alpha)I \tag{5.94}$$

$$K_E = (I + CP)^{-1}C = \alpha P^{-1} \tag{5.95}$$

Here, $C$ in Eq. (5.93) is the overall time-domain FBC gain matrix, $K$ in Eq. (5.94) is the RFBC gain matrix, and $K_E$ in Eq. (5.95) is the effective RFBC gain matrix. In addition to these three key closed-loop configuration parameters, it is useful to define two additional parameters

$$P_E = PK = (1 - \alpha)P \tag{5.96}$$

$$PK_E = \alpha I \tag{5.97}$$

The parameter $P_E$ can be considered the effective repetition-domain plant for closed-loop configuration one, and $PK_E$ can additionally be considered the effective repetition-domain plant for closed-loop configuration two. For closed-loop configuration one, using the error propagation given in Eq. (5.55) and effective repetition-domain plant defined in Eq. (5.96), the following transfer matrices results if the same weighting, or $Q_e = Q_u = 0.5I$, is again applied

$$\begin{aligned} TF2a &= I - PK\left({P_E}^T Q_e P_E + Q_u\right)^{-1} {P_E}^T Q_e \\ &= I - (1-\alpha)^2 P[(1-\alpha)^2 P^T P + I]^{-1} P^T \end{aligned} \tag{5.98}$$

$$\begin{aligned} TF2b &= PK\left({P_E}^T Q_e P_E + Q_u\right)^{-1} Q_u \\ &= (1-\alpha)P[(1-\alpha)^2 P^T P + I]^{-1} \end{aligned} \tag{5.99}$$

$$TF2c = (I + PC)^{-1} = (1 - \alpha)I \tag{5.100}$$

For configuration two, using Eq. (5.86) along with the effective repetition-domain plant defined in Eq. (5.97), and the same weighting once more, will result in the following

$$TF3a = I - PK_E([PK_E]^T Q_e [PK_E] + Q_u)^{-1}[PK_E]^T Q_e = \frac{1}{\alpha^2 + 1} I \tag{5.101}$$

$$TF3b = PK_E([PK_E]^T Q_e [PK_E] + Q_u)^{-1} Q_u = \frac{\alpha}{\alpha^2 + 1} I \tag{5.102}$$

$$TF3c = (I + PC)^{-1} = (1 - \alpha)I \tag{5.103}$$

These transfer matrices, based on the ideal controller implementation, enable some general comparisons to be made and additional insight to be gained about the configurations considered. In terms of previous cycle error contributions to the current cycle error, it can be seen from a comparison of Eqs. (5.98) and (5.101) that only closed-loop configuration two has the potential to completely eliminate the influence of plant dynamics in the error propagation. In terms of previous cycle learning signal input contributions to current cycle error, a quick comparison of Eqs. (5.99) and (5.102) similarly shows that only closed-loop configuration two has the potential to completely eliminate the influence of plant dynamics in the error propagation. Neither of these observations means that closed-loop configuration two is superior to closed-loop configuration one in general. However, for situations where similar inverse time-domain feedback controllers are considered for implementation, the observations do suggest closed-loop configuration two provides more direct control. This is because the error

propagation becomes solely a function of the designer specified gain $\alpha$, as defined in Eq. (5.91), only for closed-loop configuration two. Now, in terms of random disturbance $\delta_j \underline{d}$ rejection capability, both closed-loop configurations provide identical control. This can be seen from Eqs. (5.100) and (5.103). However, there is a clear tradeoff between random disturbance rejection capability and previous cycle error contribution for closed-loop configuration one, but not for closed-loop configuration two. The disturbance rejection capability improves for both configurations as $\alpha \to 1$, however this reduces the ability of the repetition-domain feedback controller to suppress previous cycle error contributions only for closed-loop configuration one. Another inspection of Eqs. (5.98) and (5.101) is sufficient to verify this.

Now, if a more practical situation is considered, where the $P$ model does not perfectly represent the physical plant, a more classic current cycle feedback control implementation should be considered. The designer can choose from a variety of FBC implementations, like the well-known and commonly used proportional-integral (PI) controller. Such a controller can be implemented using the following

$$u(k) = u(k-1) + K_2 e(k) + K_1 e(k-1) \tag{5.104}$$

$$R = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \\ 1 & 0 & 0 & \ddots & \vdots \\ 0 & 1 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 0 \end{bmatrix} \qquad G = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 \\ K_2 & 0 & 0 & 0 & \ddots & \vdots \\ K_1 & K_2 & 0 & 0 & \ddots & \vdots \\ 0 & K_1 & K_2 & 0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & K_1 & K_2 & 0 \end{bmatrix} \tag{5.105}$$

With this more traditional controller, $C$, $K$, $K_E$, $P_E$, and $PK_E$ must all be redefined using $R$ and $G$ as defined following Eqs. (5.104) and (5.105) above. As a result, the analytical comparison is more involved and requires the aid of numerical simulation software to generate plots of the corresponding transfer functions. The transfer matrices for such an

implementation will be much closer to the original forms given in Eqs. (5.55) and (5.86), even when equal weighting, or $Q_e = Q_u = 0.5I$, is used in the cost functions. For closed-loop configuration one

$$\begin{aligned} TF2a &= I - PK\left({P_E}^T Q_e P_E + Q_u\right)^{-1} {P_E}^T Q_e \\ &= I - P_E([P_E]^T[P_E] + I)^{-1}[P_E]^T \end{aligned} \tag{5.106}$$

$$\begin{aligned} TF2b &= PK\left({P_E}^T Q_e P_E + Q_u\right)^{-1} Q_u \\ &= P_E([P_E]^T[P_E] + I)^{-1} \end{aligned} \tag{5.107}$$

$$TF2c = (I + PC)^{-1} \tag{5.108}$$

Similarly, for closed-loop configuration two

$$\begin{aligned} TF3a &= I - PK_E([PK_E]^T Q_e[PK_E] + Q_u)^{-1}[PK_E]^T Q_e \\ &= I - PK_E([PK_E]^T[PK_E] + I)^{-1}[PK_E]^T \end{aligned} \tag{5.109}$$

$$\begin{aligned} TF3b &= PK_E([PK_E]^T Q_e[PK_E] + Q_u)^{-1} Q_u \\ &= PK_E([PK_E]^T[PK_E] + I)^{-1} \end{aligned} \tag{5.110}$$

$$TF3c = (I + PC)^{-1} \tag{5.111}$$

Using the first pair of transfer matrices Eqs. (5.89) and (5.90), the second set of transfer matrices Eqs. (5.98) – (5.103), and the final set of transfer matrices Eqs. (5.106) – (5.111), the analytical comparison can be completed using numerical simulation.

## 5.5: Numerical Simulation

Numerical simulations were conducted on MATLAB 7.7 to complete the comparison of the basic open-loop configuration and two closed-loop configurations considered. A well-conditioned system model is used to effectively consider control problems in which a designer has the option to consider both the ideal inverse controller and traditional PI feedback controller discussed above. It is very important to remember that in order to be

able to use an inverse controller in practice, a relatively accurate system model is generally required. The necessary $P$ model is generated using a third order system with the following continuous time transfer function

$$Y(s) = G(s)U(s) = \left(\frac{s^2 + 26s + 170}{100}\right)\left(\frac{a}{s+a}\right)\left(\frac{{\omega_n}^2}{s^2 + 2\,\zeta\,\omega_n s + {\omega_n}^2}\right)U(s) \tag{5.112}$$

For the numerical simulations $a = 8.8$, $\omega_n = 37$, $\zeta = 0.5$, and the continuous time model was discretized with a ZOH using a 100 Hz sample rate for 2 seconds. This means that $p = 200$ as defined in Eq. (1.2) in the introductory chapter. Three different values for $\alpha$ were simulated, $\alpha = 0.1$, $\alpha = 0.5$, and $\alpha = 0.9$, for the inverse controller comparisons of the configurations. The basic open-loop configuration can be evaluated using Eqs. (5.3), (5.4), (5.89), and (5.90) to generate sensitivity transfer matrix plots and singular value plots. Next, closed-loop configuration one can be assessed using Eqs. (5.94), (5.96), (5.98) – (5.100), and (5.106) – (5.108) to similarly generate sensitivity transfer matrix and singular value plots. Finally, Eqs. (5.95), (5.97), (5.101) – (5.103), and (5.109) – (5.111) can be used to generate the sensitivity transfer matrix and singular value plots for closed-loop configuration two. Using these formulations, plots are presented in two separate groups. The first group shows a performance comparison based on sensitivity transfer matrix plots and the second group shows a performance comparison based on plots of singular values. For the sensitivity transfer matrix plots, Figures 5-4, 5-5, and 5-6 show the results for all three configurations using the inverse controller with $\alpha = 0.1$, $\alpha = 0.5$, and $\alpha = 0.9$ respectively. The sensitivity transfer matrix plot for the classic PI controller, with $K_2 = 1.15$ and $K_1 = -0.7$, is given in Figure 5-7 for the more practical situation. Next, the singular value plots are similarly given in Figures 5-8, 5-9, and 5-10

for the inverse controller simulations, again with $\alpha = 0.1$, $\alpha = 0.5$, and $\alpha = 0.9$ respectively. Finally, the singular value plot for the classic PI controller is provided in Figure 5-11 to complete the comparison of the configurations. It is worthwhile to mention that while the sensitivity transfer matrix and singular value plots are closely related, the sensitivity transfer matrix is the recommended tool for evaluation and comparison purposes. This is because it offers the designer a sense of the frequency response for the various parameters and transfer functions analyzed that the singular value plots do not convey, since they are not frequency dependent. The singular value plots are nevertheless important because they are simpler to create and provide some quick insight into the upper bounds of the sensitivity transfer matrix. In fact, several basic design decisions can easily be made using Figures 5-8 through 5-11, especially in situations where the plots of interest depict all singular values being less than one. Much of the required information necessary to make combination controller design decisions can be obtained from the (a) and (b) plots of each figure, since these contain the effective RFBC and the effective plant gains. Ideally, the effective plant will have high gain over the frequencies which make up the signal to be followed and low gains outside of those frequencies. This can be done through the appropriate design of the effective RFBC, as suggested in the various given figures. The error performance information, another critical factor to consider when designing the combination controller, is contained most explicitly in the (c), (d), and (e) plots of every figure. For the ideal implementation, unlike with the (a) and (b) plots, these plots would be zero for every frequency from zero up to Nyquist frequency. This is not possible using time-domain FBC due to the well know waterbed effect, which essentially states rejection over all frequencies is not possible. If a design targets a specific band of frequencies to reject, it

consequently results in amplification of frequencies outside that band. Since the random disturbance rejection capability completely relies on time-domain FBC, most practical controllers will produce plots similar to Figure 5-7c, where disturbance rejection only occurs over a small band, in this case from 0 to about 16.5 Hz. This is still far better than the basic open-loop configuration where all frequencies are passed and the controller has no ability to handle random disturbances. When comparing closed-loop configuration one and two for the more practical situation, it appears the most favorable design will be application specific based on Figure 5-7, with plot (d) providing the strongest case for this. If the application uses a desired trajectory with all frequency content above 16.5 Hz, it appears closed-loop configuration one is most favorable. On the other hand, if a desired trajectory with primarily low frequency content below 16.5 Hz is utilized, then closed-loop configuration two should be used. Either way, these types of plots and their corresponding parameters and transfer matrices provide very useful insight. This information can be paramount in making good design decisions with regards to which configuration, cost function, and overall time-domain and repetition-domain feedback control schemes to implement.

## 5.6: Summary of Findings

Three controller configurations have been thoroughly investigated in this work specifically to facilitate implementation in any iterative learning control application. The first configuration considered was the basic open-loop ILC scheme which is an open-loop implementation in time, but a closed-loop controller in repetition. This type of open-loop approach can be considered straightforward to setup in practice and relatively easy to optimize through the use of a simple cost function, since there is no distinction between the plant input and learning signal. The main weakness of controller

configurations of this type is that they are ill-equipped to handle stochastic, non-repeating disturbances. To address this limitation, two configurations which attempt to merge FBC and ILC into an optimal combination controller were evaluated. There are two fundamentally distinct ways in which a feedback and learning combination controller can be configured, one with the learning signal incorporated inside the feedback loop and one with the learning signal incorporated outside the feedback loop. Closed-loop FBC and ILC configuration one examines combination controllers that integrate the learning signal inside the feedback loop. Closed-loop FBC and ILC configuration two examines combination controllers that integrate the learning signal outside the feedback loop. For both of these configurations, the learning signal and plant input are distinct. When optimizing these types of controllers using cost functions, the variable used to minimize the costs was the update to the learning signal. There were four cost functions available for consideration when attempting to optimize the two closed-loop configurations. For the basic open-loop configuration, only two cost functions were considered due to the redundancy that exists when the plant input and learning signal are the same. It turned out that the optimal controllers for these different configurations are closely related and similar in structure, with the differences being attributed to a few key parameters. The error performance between the developed optimal controllers was found to be comparable, and the selection of configuration, cost function, and overall control scheme was application-dependent. However, detailed formulation is provided for each configuration to support the overall design and implementation process.

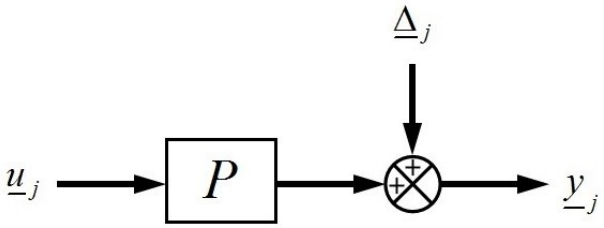



**Figure 5-1. Basic Open-Loop ILC Configuration**

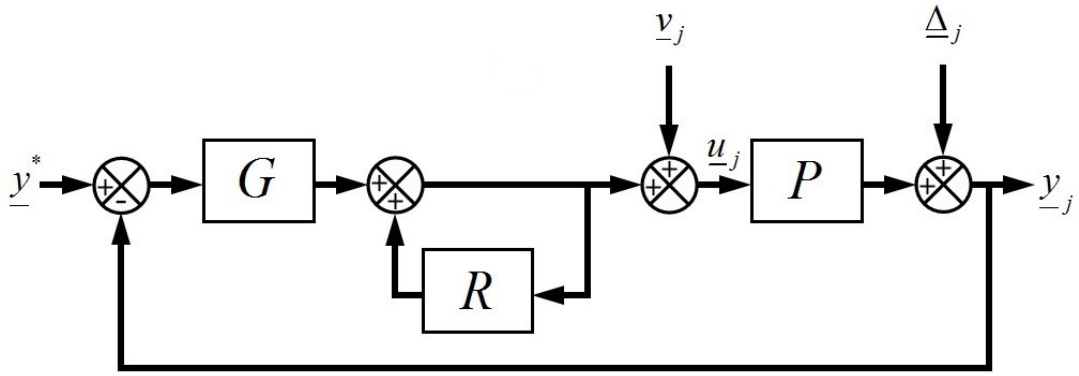



**Figure 5-2. Closed-Loop FBC and ILC Configuration One**

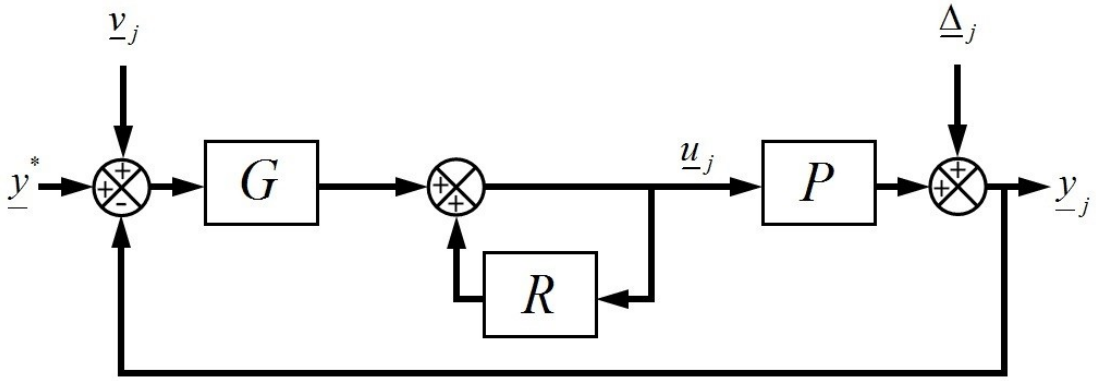
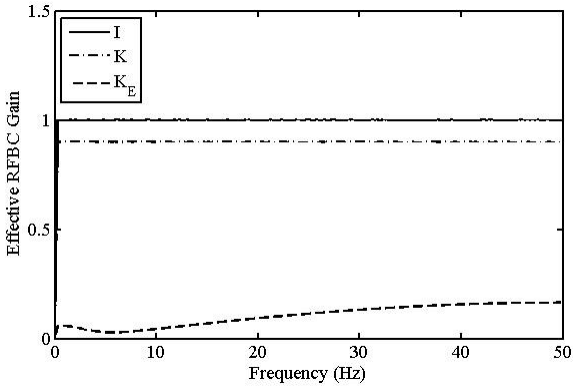
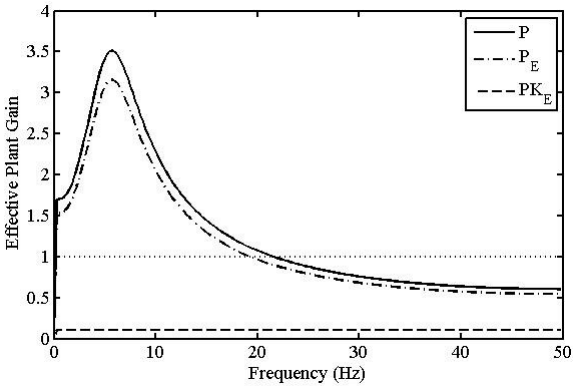



**Figure 5-3. Closed-Loop FBC and ILC Configuration Two**

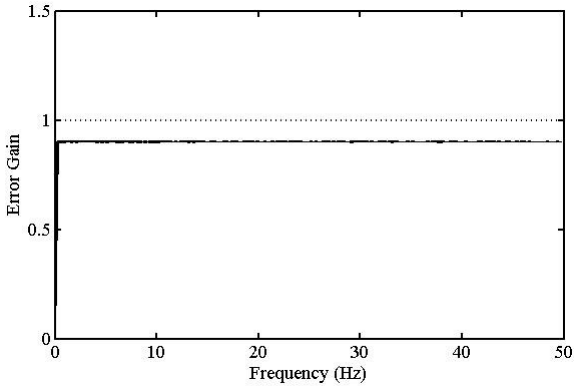
## Performance Comparison Using Sensitivity Transfer Matrix



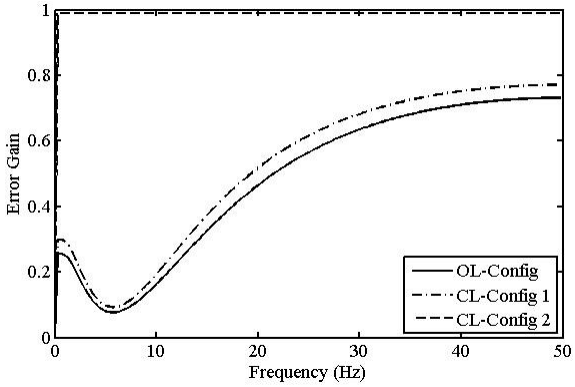

a. Effective RFBC Sensitivity TM



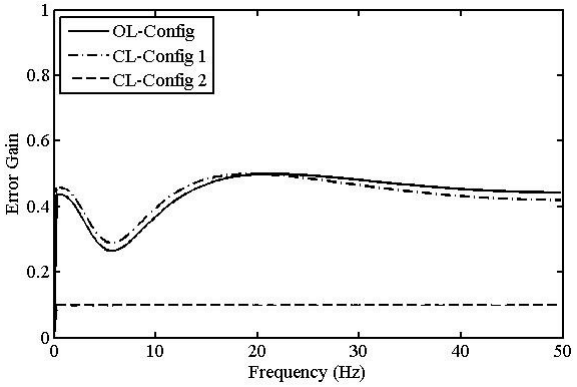

b. Effective Plant Sensitivity TM




c. Disturbance Difference Sensitivity TM

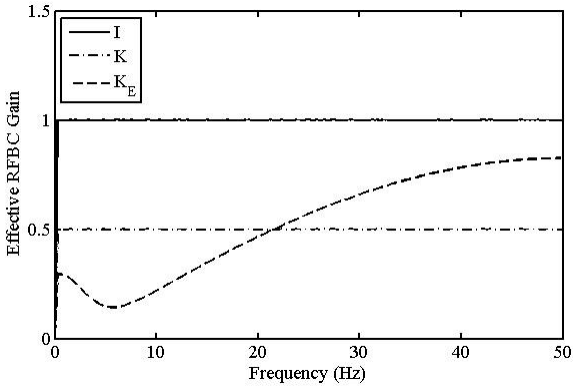



d. Previous Cycle Error Sensitivity TM

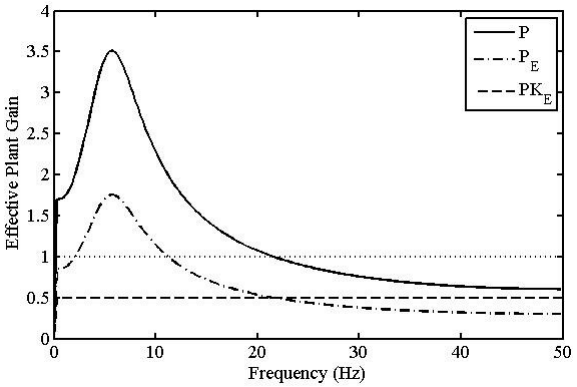



e. Previous Cycle Input Sensitivity TM

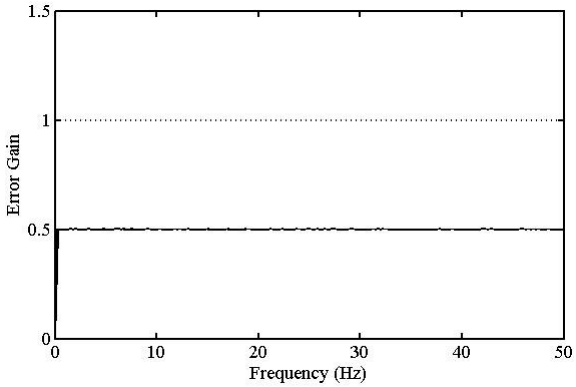
**Figure 5-4. Inverse Time-Domain FBC with $\propto = 0.1$ Transfer Matrices**



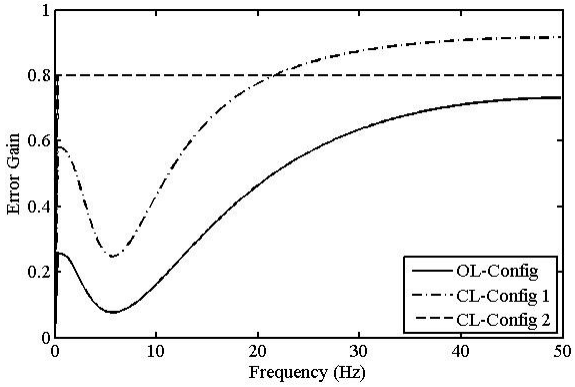

**a. Effective RFBC Sensitivity TM**



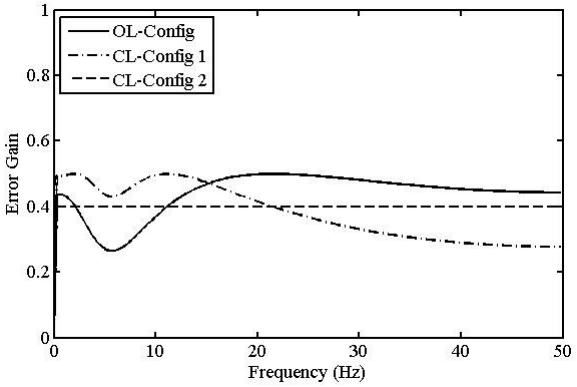

**b. Effective Plant Sensitivity TM**




**c. Disturbance Difference Sensitivity TM**

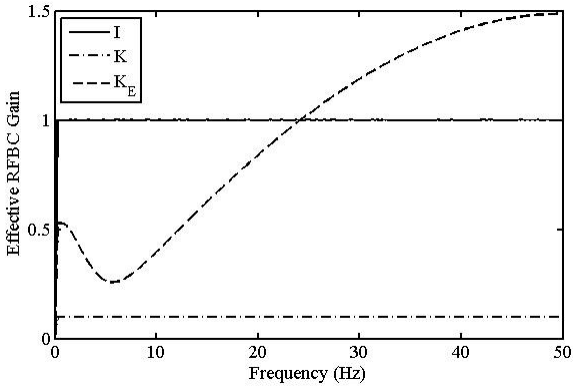



**d. Previous Cycle Error Sensitivity TM**

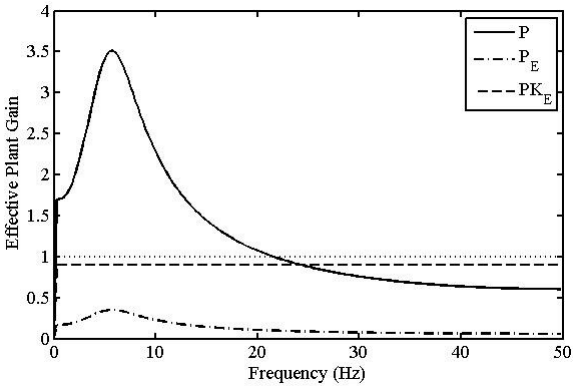



**e. Previous Cycle Input Sensitivity TM**

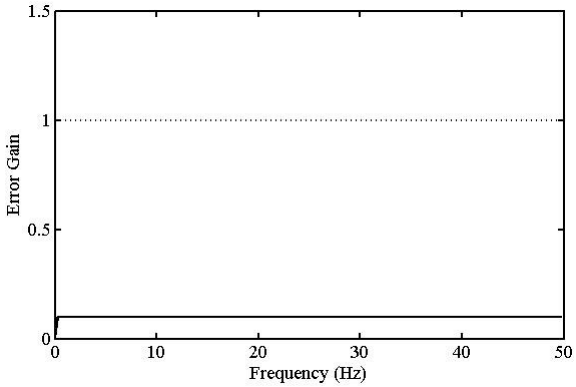
**Figure 5-5. Inverse Time-Domain FBC with $\propto = 0.5$ Transfer Matrices**



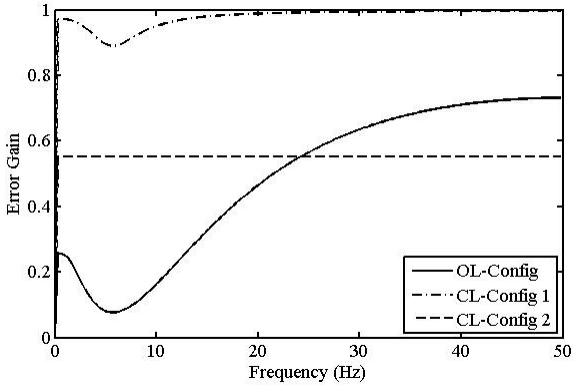

**a. Effective RFBC Sensitivity TM**



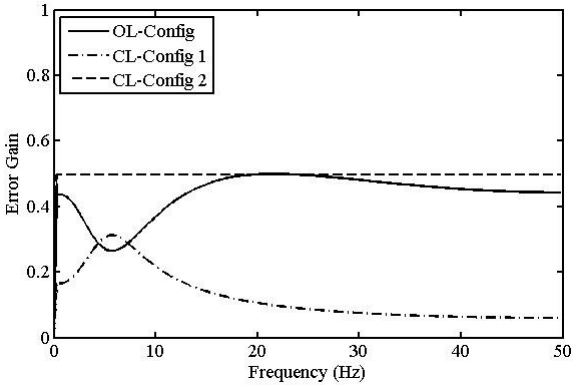

**b. Effective Plant Sensitivity TM**




**c. Disturbance Difference Sensitivity TM**

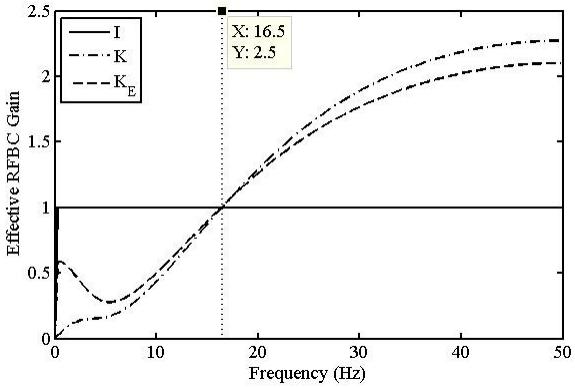



**d. Previous Cycle Error Sensitivity TM**

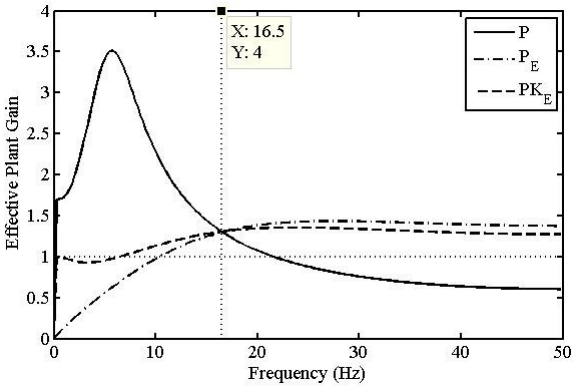



**e. Previous Cycle Input Sensitivity TM**

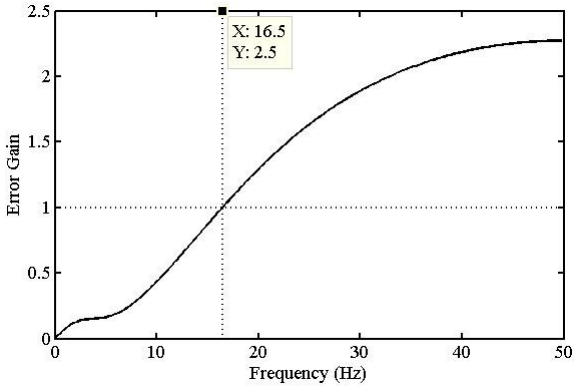
**Figure 5-6. Inverse Time-Domain FBC with $\propto = 0.9$ Transfer Matrices**



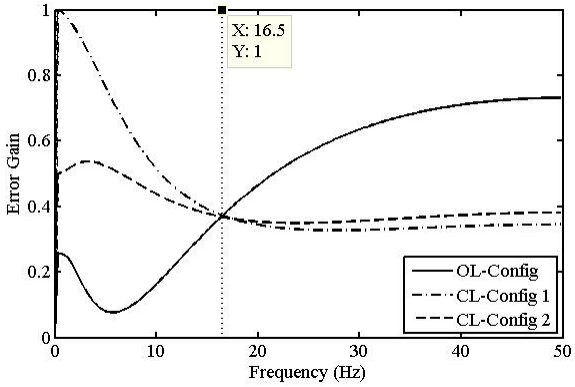

a. Effective RFBC Sensitivity TM



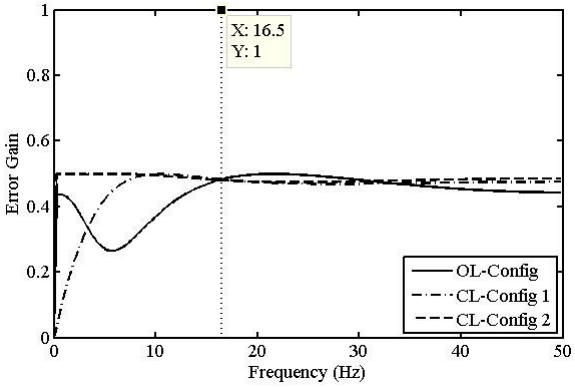

b. Effective Plant Sensitivity TM




c. Disturbance Difference Sensitivity TM



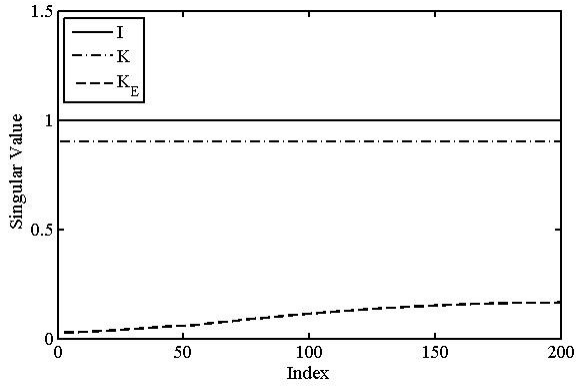

d. Previous Cycle Error Sensitivity TM



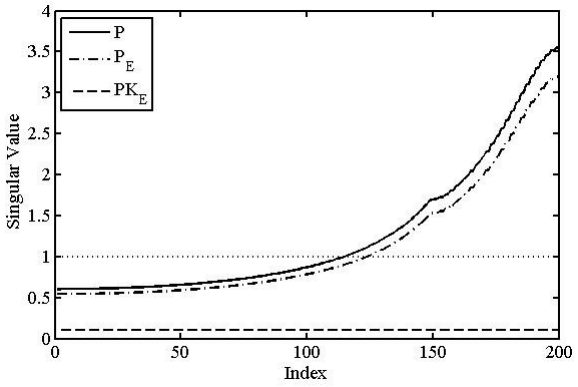

e. Previous Cycle Input Sensitivity TM

**Figure 5-7. Classic PI FBC Transfer Matrices**

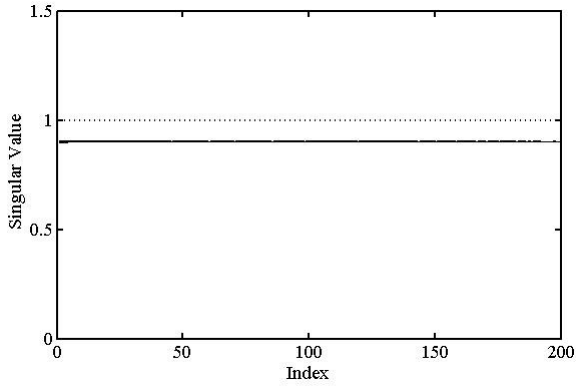
## Performance Comparison Using Singular Values



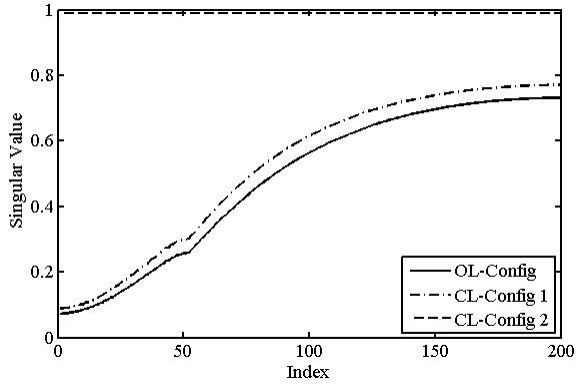

a. Effective RFBC Singular Values



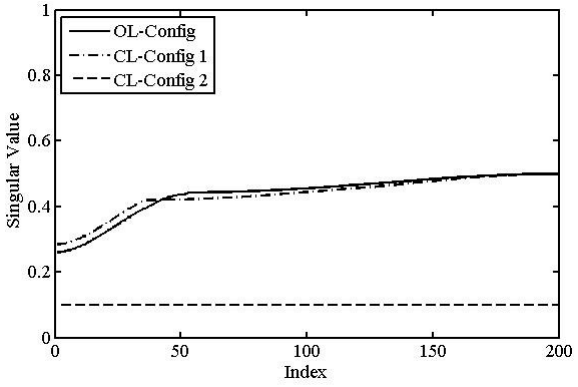

b. Effective Plant Singular Values




c. Disturbance Difference Singular Values

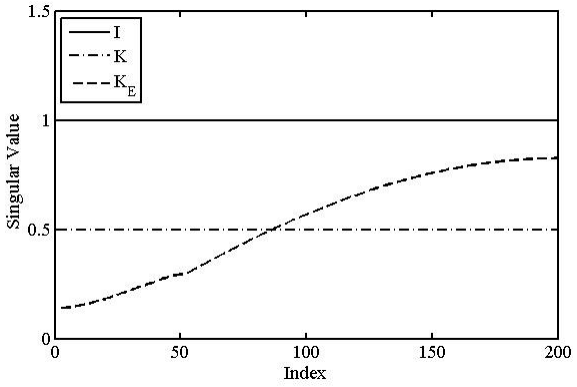



d. Previous Cycle Error Singular Values

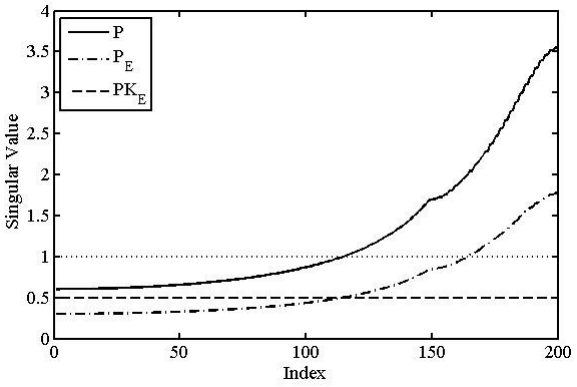



e. Previous Cycle Input Singular Values

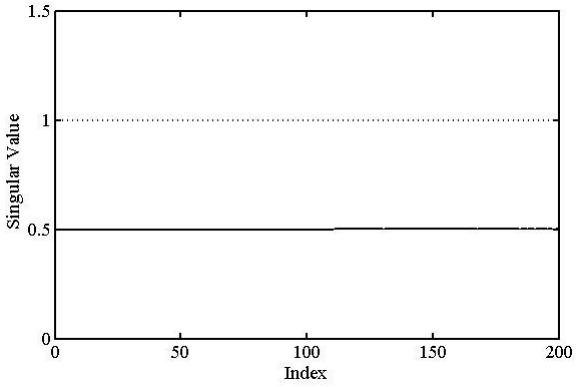
**Figure 5-8. Inverse Time-Domain FBC with $\propto = 0.1$ Singular Values**



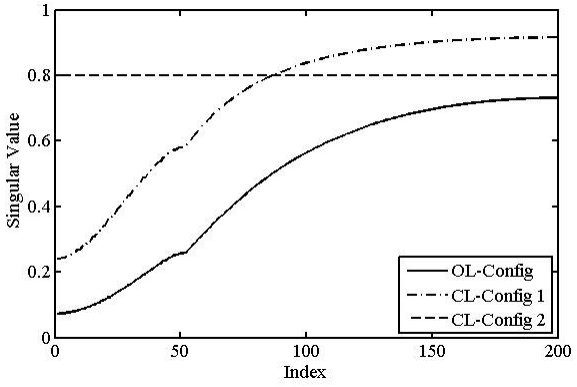

a. Effective RFBC Singular Values



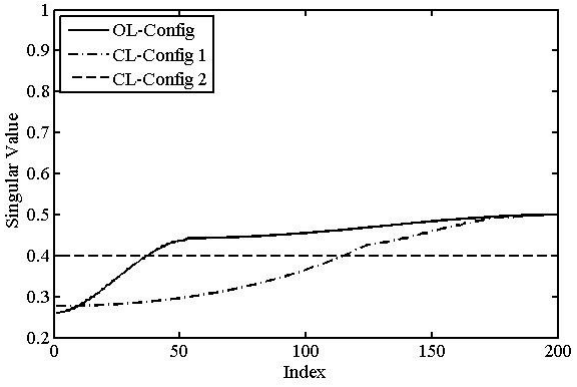

b. Effective Plant Singular Values




c. Disturbance Difference Singular Values

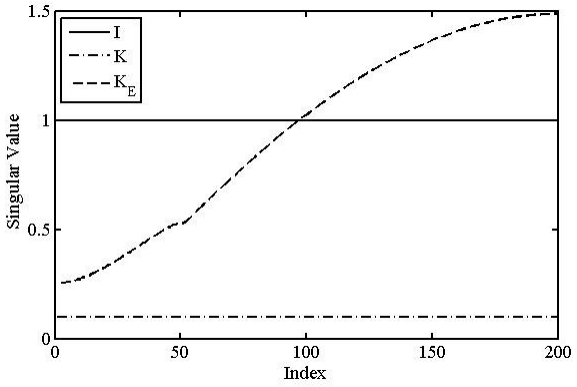



d. Previous Cycle Error Singular Values

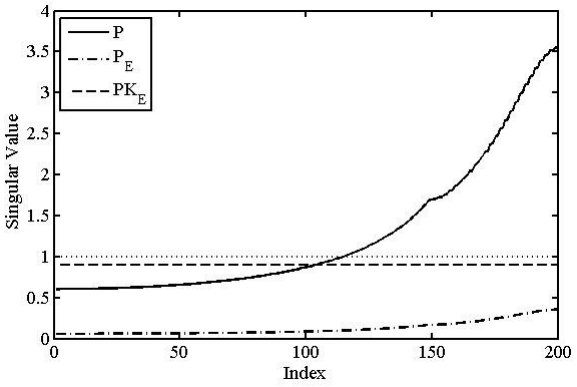



e. Previous Cycle Input Singular Values

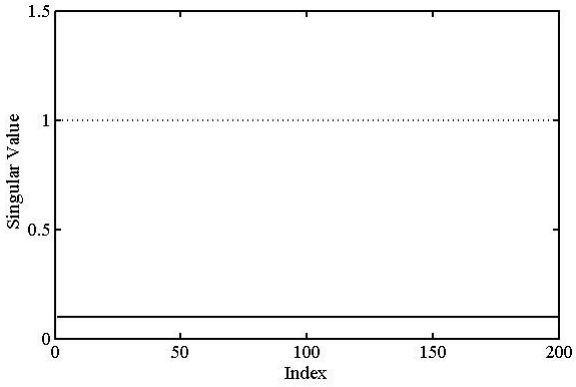
**Figure 5-9. Inverse Time-Domain FBC with $\propto = 0.5$ Singular Values**



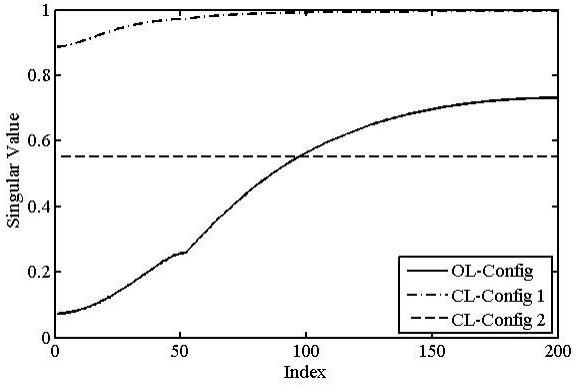

a. Effective RFBC Singular Values



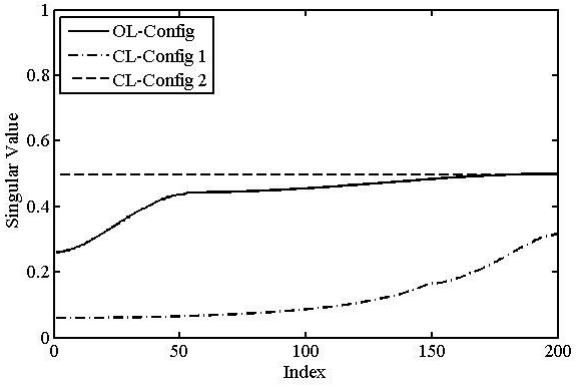

b. Effective Plant Singular Values




c. Disturbance Difference Singular Values

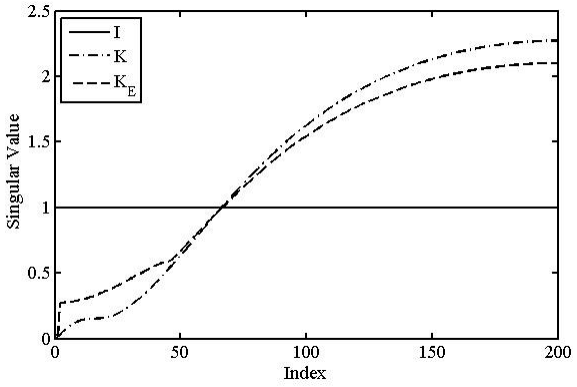



d. Previous Cycle Error Singular Values

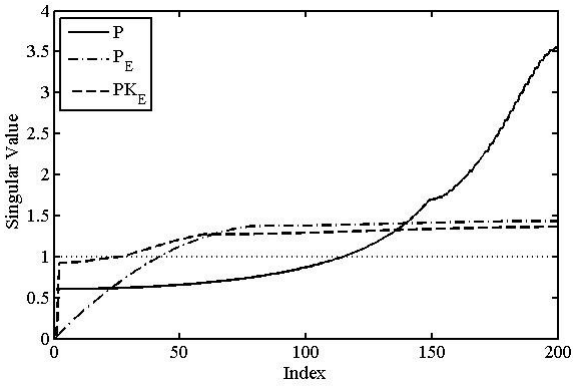



e. Previous Cycle Input Singular Values

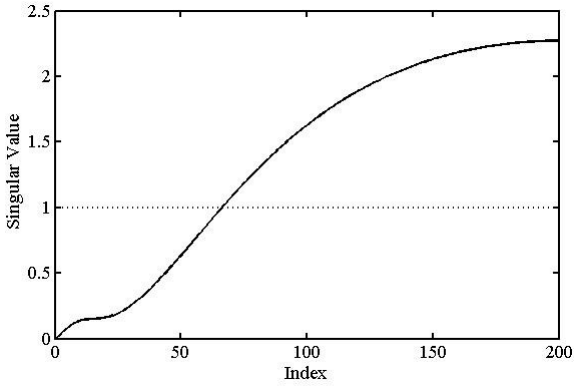
**Figure 5-10. Inverse Time-Domain FBC with $\propto = 0.9$ Singular Values**



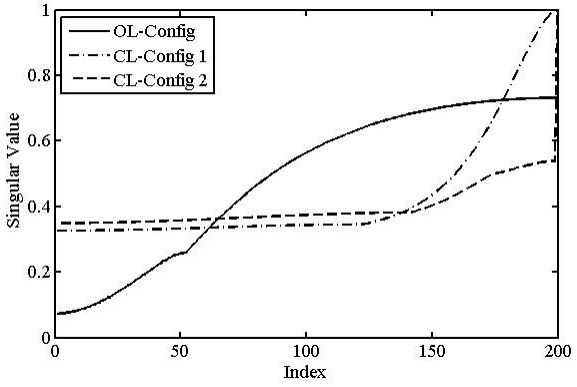

a. Effective RFBC Singular Values



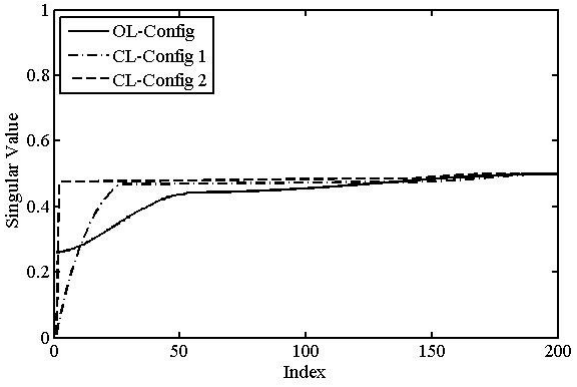

b. Effective Plant Singular Values




c. Disturbance Difference Singular Values




d. Previous Cycle Error Singular Values




e. Previous Cycle Input Singular Values

**Figure 5-11. Classic PI FBC Singular Values**

# CHAPTER SIX: Conclusion

Various effective methods have been developed to design iterative learning controllers for linear systems. These control laws make use of a Toeplitz matrix containing a number of system Markov parameters, or the unit pulse samples associated with the duration of the desired trajectory. For a long trajectory with a large number of time steps, it can be difficult to accurately determine these Markov parameters. In Chapter 2, two practical methods were investigated, OKID and DFT, to determine the pulse response of any linear system using input-output data. Methods were developed to address the issue when the number of accurately obtainable Markov parameters is limited. The effects of inaccurate or noisy Markov parameters were also examined and evaluated in use with the control laws. In this manner, some simple expansion methods were established that aim to produce stable and well-behaved control laws which can be used to control a large number of time steps even when only a small number of Markov parameters of limited accuracy is available.

Improved combination controllers, incorporating both FBC and ILC, have been developed for learning control applications in linear systems. These controllers improve previously developed combination controllers, by allowing FBC designs that are non-causal by one time step. The controllers are then further improved by the use of disturbance estimators both in the time-domain and in the repetition-domain. As described in Chapter 3, numerical simulations confirmed that these combination controllers, with disturbance estimation, significantly improve both tracking performance and learning behavior.

Optimal cost function based combination controllers, incorporating both finite-time FBC and ILC, have been developed in linear systems. These controllers are dependent on the $Q$ matrix to implement the finite-time feedback controller. Two separate designs, based on a static or dynamic $Q$ matrix, were provided for implementation in Chapter 4. The $Q$ matrix was found simultaneously with the learning controller by specifying a few gains in the most basic case. Specific design considerations take full advantage of the inherent flexibility of the $Q$ matrix based design for learning control applications. These considerations covered the fundamental aspects of this design approach including appropriate generation of the primary design agent, the training matrix, as well as the use of finite-time sensitivity transfer matrices as an evaluation tool. It has been demonstrated that finite-time sensitivity transfer matrices can be used to reliably bridge the gap between finite-time control design approaches and more conventional, steady-state, control design approaches. Numerical simulations confirmed that these combination controllers significantly improve both the tracking performance and the learning behavior when compared to conventional feedback only control. It was further demonstrated that the most optimal design, which guarantees optimal performance in all situations, is the dynamic $Q$ matrix based finite-time FBC and ILC controller.

Three controller configurations were thoroughly investigated in Chapter 5, specifically to facilitate implementation in any iterative learning control application. The first configuration considered was the basic open-loop ILC scheme, which is open-loop in time, but closed-loop in repetition. This type of open-loop approach can be considered straightforward to setup in practice and relatively easy to optimize through the use of a simple cost function since there is no distinction between the plant input and learning signal. A configuration of this type is ill-equipped to handle stochastic, non-repeating

disturbances. To address this limitation, two configurations are considered: one with the learning signal incorporated inside the feedback loop and one with the learning signal incorporated outside the feedback loop.

Closed-loop FBC and ILC configuration one was used to explore combination controllers that integrate the learning signal inside the feedback loop. Closed-loop FBC and ILC configuration two was used to explore combination controllers that integrate the learning signal outside the feedback loop. For both of these configurations, the learning signal and plant input were distinct. When optimizing these types of controllers using cost functions, the variable used to minimize the costs was the update to the learning signal. There were four cost functions available for consideration when attempting to optimize the two closed-loop configurations. For the basic open-loop configuration, only two cost functions were considered due to the redundancy that exists when the plant input and learning signal are the same. It turns out that the optimal controllers for these different configurations are closely related and similar in structure, with the differences being attributed to a few key parameters, such as the effective repetition-domain plant and effective repetition-domain feedback control matrix. The error performance of these controllers is also similar, and therefore design decisions regarding which overall control strategy to implement should be carefully considered for each individual application. As stated in the introduction, the main goal of this research is to establish an end-to-end simultaneous design approach that is adaptable for use in all practical ILC applications. This is precisely what has been developed in this thesis and the results provided in the previous chapters can be used to facilitate a thorough application-specific assessment on a case by case basis.

# REFERENCES


[1] M. UCHIYAMA, "Formulation of High-Speed Motion Pattern of a Mechanical Arm by Trial," *Transactions of the Society for Instrumentation and Control Engineers*, Vol. 14, 1978, pp. 706-712.

[2] S. ARIMOTO, S. KAWAMURA, AND F. MIYAZAKI, "Bettering Operation of Robots by Learning," *Journal of Robotic Systems*, Vol. 1, No. 2, 1984, pp. 123-140.

[3] G. CASALINO AND G. BARTOLINI, "A Learning Procedure for the Control of Movements of Robotic Manipulators," *Proceedings of the IASTED Symposium on Robotics and Automation*, May 1984, pp. 108-111.

[4] J.J. CRAIG, "Adaptive Control of Manipulators Through Repeated Trials," *Proceedings of the American Control Conference*, June 1984, pp. 1566-1573.

[5] K.L. MOORE, *Iterative Learning Control for Deterministic Systems*, Springer-Verlag, London, U.K., Advances in Industrial Control, 1993.

[6] Z. BIEN AND J.-X. XU, *Iterative Learning Control: Analysis, Design, Integration and Applications*, Kluwer, Boston, 1998.

[7] Y. CHEN AND C. WEN, *Iterative Learning Control: Convergence, Robustness, and Applications*, Springer, London, 1999.

[8] J.-X. XU AND Y. TAN, *Linear and Nonlinear Iterative Learning Control*, Springer, Berlin, 2003.

[9] K.L. MOORE AND J.-X. XU, "Editorial: Iterative Learning Control," *International Journal of Control*, Vol. 73, No. 10, 2000.

[10] "Iterative Learning Control," *Asian Journal of Control*, Vol. 4, No. 1, 2002.

[11] H.-S. AHN AND D.A. BRISTOW, "Special Issue on Iterative Learning Control," *Asian Journal of Control*, Vol. 13, No. 1, 2011.

[12] K.L. MOORE, M. DAHLEH, AND S.P. BHATTACHARYYA, "Iterative learning control: a survey and new results," *Journal of Robotic Systems*, Vol. 9, No. 5, 1992, pp. 563-594.

[13] R. HOROWITZ, “Learning control of robot manipulators,” *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 115, No. 2B, 1993, pp. 402-411.

[14] K.L. MOORE, “Iterative learning control - an expository overview,” *Applied and Computational Controls, Signal Processing, and Circuits*, Vol. 1, No. 1, 1999, pp. 151-241.

[15] J.-X. XU, “The frontiers of iterative learning control - part I,” *Journal of Systems, Control and Information*, Vol. 46, No. 2, 2002, pp. 63-73.

[16] J.-X. XU, “The frontiers of iterative learning control - part II,” *Journal of Systems, Control and Information*, Vol. 46, No. 5, 2002, pp. 233-243.

[17] D.A. BRISTOW, M. THARAYIL AND A.G. ALLEYNE, “A survey of iterative learning control,” *IEEE Control Systems Magazine*, Vol. 26, No. 3, 2006, pp. 96-114.

[18] H.-S. AHN, Y.Q. CHEN, AND K.L. MOORE, “Iterative learning control: brief survey and categorization,” *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, Vol. 37, No. 6, 2007, pp. 1099-1122.

[19] Y. WANG, F. GAO, F.J. DOYLE III, “Survey on iterative learning control, repetitive control, and run-to-run control,” *Journal of Process Control*, Vol. 19, No. 10, 2009, pp. 1589-1600.

[20] M. YAMAKITA AND K. FURUTA, “Iterative Generation of Virtual Reference for a Manipulator,” *Robotica*, Vol. 9, No. 1, 1991, pp. 71-80.

[21] R.W. LONGMAN, “Iterative Learning Control and Repetitive Control for Engineering Practice,” *International Journal of Control*, Vol. 73, No. 10, 2000, pp. 930-954.

[22] W. HUANG, L. CAI, AND X. TANG, “Adaptive repetitive output feedback control for friction and backlash compensation of a positioning table,” *Proceedings of the 37th IEEE Conference on Decision and Control*, December 1998, pp. 1250-1251.

[23] D.H. OWENS AND G. MUNDE, “Error convergence in an adaptive iterative learning controller,” *International Journal of Control*, Vol. 73, No. 10, 2000, pp. 851-857.

[24] H. CHEN AND P. JIANG, "Adaptive iterative feedback control for nonlinear system with unknown high-frequency gain," *Proceedings of the 4th World Congress on Intelligent Control and Automation*, June 2002, pp. 847-851.

[25] H. CHEN AND P. JIANG, "Adaptive iterative feedback control for nonlinear system with unknown control gain," *Control Theory Application*, Vol. 20, No. 5, 2003, pp. 691-694.

[26] A. TAYEBI, "Adaptive iterative learning control for robot manipulators," *Proceedings of the American Control Conference*, June 2003, pp. 4518-4523.

[27] S. ISLAM AND A. TAYEBI, "New adaptive iterative learning control (AILC) for uncertain robot manipulators," *Proceedings of the Canadian Conference on Electrical and Computer Engineering*, May 2004, pp. 1645-1651.

[28] A. TAYEBI, "Adaptive iterative learning control for robot manipulators," *Automatica*, Vol. 40, No. 7, 2004, pp. 1195-1203.

[29] F. PADIEU AND R. SU, "$\mathcal{H}\infty$ approach to learning control systems," *International Journal of Adaptive Control Signal Processing*, Vol. 4, No. 6, 1990, pp. 465-474.

[30] N. AMANN, D.H. OWENS, E. ROGERS, AND A. WAHL, "An $\mathcal{H}\infty$ approach to linear iterative learning control design," *International Journal of Adaptive Control Signal Processing*, Vol. 10, No. 6, 1996, pp. 767-781.

[31] C.J. GOH AND W.Y. YAN, "An $\mathcal{H}\infty$ synthesis of robust current error feedback learning control," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 118, No. 2, 1996, pp. 341-346.

[32] T.Y. DOH, J.H. MOON, K.B. JIN, AND M.J. CHUNG, "Robust iterative learning control with current feedback for uncertain linear systems," *International Journal of Systems Science*, Vol. 30, No. 1, 1999, pp. 39-47.

[33] W. PASZKE, K. GALKOWSKI, E. ROGERS, AND D.H. OWENS, "$\mathcal{H}\infty$ control of discrete linear repetitive processes," *Proceedings of the 42nd IEEE Conference on Decision and Control*, December 2003, pp. 628-633.

[34] I. CHIN, S.J. QIN, K.S. LEE, AND M. CHO, "A two-stage iterative learning control technique combined with real-time feedback for independent disturbance rejection," *Automatica*, Vol. 40, No. 11, 2004, pp. 1913-1920.

[35] D. H. OWENS AND N. AMANN, *Norm-Optimal Iterative Learning Control*, Internal Report Series of the Centre for Systems and Control Engineering, University of Exeter, 1994.

[36] N. AMANN, D.H. OWENS, AND E. ROGERS, "Iterative learning control for discrete-time systems using optimal feedback and feedforward actions," *Proceedings of the 34th IEEE Conference on Decision and Control*, Vol. 2, December 1995, pp. 1696-1701.

[37] N. AMANN, D.H. OWENS, AND E. ROGERS, "Predictive optimal iterative learning control," *International Journal of Control*, Vol. 69, No. 2, 1998, pp. 203-226.

[38] J.A. FRUEH AND M.Q. PHAN, "Linear Quadratic Optimal Learning Control (LQL)," *International Journal of Control*, Vol. 73, No. 10, 2000, pp. 832-839.

[39] C.-H. CHOI AND T.-J. JANG, "Iterative learning control in feedback systems based on an objective function," *Asian Journal of Control*, Vol. 2, No. 2, 2000, pp. 101-110.

[40] D.H. OWENS AND J.J. HÄTÖNEN, "A new optimality based adaptive ILC-algorithm," *Proceedings of the 7th International Conference on Control, Automation, Robotics and Vision*, December 2002, pp. 1496-1501.

[41] K. TAKANISHI, M.Q. PHAN, AND R.W. LONGMAN, "Design of an Optimal Combination of Feedback and Iterative Learning Controllers," *Advances in the Astronautical Sciences*, Vol. 119, 2005, pp. 789-800.

[42] P.B. GOLDSMITH, "The equivalence of LTI iterative learning control and feedback control," *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, October 2000, pp. 3443-3448.

[43] P.B. GOLDSMITH, "The fallacy of causal iterative learning control," *Proceedings of the American Control Conference*, June 2001, pp. 4475-4480.

[44] P.B. GOLDSMITH, "On the equivalence of causal LTI iterative learning control and feedback control," *Automatica*, Vol. 38, No. 4, 2002, pp. 703-708.

[45] D.H. OWENS AND E. ROGERS, "Comments on 'On the equivalence of causal LTI iterative learning control and feedback control'," *Automatica*, Vol. 40, No. 5, 2004, pp. 895-898.

[46] P.B. GOLDSMITH, "Author's reply to "Comments on 'On the equivalence of causal LTI iterative learning control and feedback control'"," *Automatica*, Vol. 40, No. 5, 2004, pp. 899-900.

[47] P.B. GOLDSMITH, "Stability, convergence, and feedback equivalence of LTI iterative learning control," *Proceedings of the IFAC 15th World Congress*, July 2002.

[48] K.S. LEE, S.H. BANG, AND K.S. CHANG, "Feedback-assisted iterative learning control based on an inverse process model," *Journal of Process Control*, Vol. 4, No. 2, 1994, pp. 77-89.

[49] H.F. DOU, K.K. TAN, T.H. LEE, AND Z.Y. ZHOU, "Iterative learning feedback control of human limbs via functional electrical stimulation," *Control Engineering Practice*, Vol. 7, No. 3, 1999, pp. 315-325.

[50] S.C. LI, X.H. XU, AND L. PING, "Feedback-assisted iterative learning control for batch polymerization reactor," *Advances in Neural Networks-ISSN 2004*, *Part 2 Lecture Notes in Computer Science*, Vol. 3174, 2004, Springer-Verlag, pp. 181-187.

[51] T.-Y. DOH AND J.-H. MOON, "Feedback-based iterative learning control for uncertain linear MIMO systems," *Proceedings of the 5th Asian Control Conference*, July 2004, pp. 198-203.

[52] S. LI, X. XU, AND P. LI, "Frequency domain analysis for feedback-assisted iterative learning control," *Proceedings of the International Conference on Information Acquisition*, June 2004, pp. 1-4.

[53] J.-X. XU, X.-W. WANG, AND L.T. HENG, "Analysis of continuous iterative learning control systems using current cycle feedback," *Proceedings of the American Control Conference*, June 1995, pp. 4221-4225.

[54] N. AMANN, D.H. OWENS, AND E. ROGERS, "Iterative learning control for discrete-time systems with exponential rate of convergence," *IEE Proceedings on Control Theory and Applications*, Vol. 143, No. 2, March 1996, pp. 217-224.

[55] K.L. MOORE, Y.Q. CHEN, AND V. BAHL, "Feedback controller design to ensure monotonic convergence in discrete-time, P-type iterative learning control," *Proceedings of the 4th Asian Control Conference*, September 2002, pp. 440-445.

[56] E. ROGERS AND D.H. OWENS, *Stability Analysis for Linear Repetitive Processes*, Springer-Verlag, Berlin, 1992.

[57] C.J. GOH, "A frequency domain analysis of learning control," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 116, No. 4, 1994, pp. 781-786.

[58] M. NORRLÖF AND S. GUNNARSSON, "Time and frequency domain convergence properties in iterative learning control," *International Journal of Control*, Vol. 75, No. 14, 2002, pp. 1114-1126.

[59] D.H. OWENS, E. ROGERS, AND K.L. MOORE, "Analysis of linear iterative learning control schemes using repetitive process theory," *Asian Journal of Control*, Vol. 4, No. 1, 2002, pp. 68-89.

[60] M. NORRLÖF, "Comparative study on first and second order ILC-frequency domain analysis and experiments," *Proceedings of the 39th IEEE Conference on Decision and Control*, Vol. 4, December 2000, pp. 3415–3420.

[61] J.-X. XU, T.H. LEE, AND H.W. ZHANG, "Analysis and comparison of iterative learning control schemes," *Engineering Applications of Artificial Intelligence*, Vol. 17, No. 6, 2004, pp. 675-686.

[62] U. GRENANDER AND G. SZEGÕ, *Toeplitz Forms and Their Applications*, University of California Press, Berkeley, CA, 1958.

[63] J.H. LEE, K.S. LEE, AND W.C. KIM, "Model-based iterative learning control with a quadratic criterion for time-varying linear systems," *Automatica*, Vol. 36, No. 5, 2000, pp. 641-657.

[64] J.-N. JUANG, M. PHAN, L. G. HORTA, AND R. W. LONGMAN, "Identification of Observer/Kalman Filter Markov Parameters: Theory and Experiments," *Journal of Guidance, Control, and Dynamics*, Vol. 16, No. 2, March-April 1993, pp. 320-329.

[65] M.Q. PHAN, R.W. LONGMAN, AND K.L. MOORE, "A Unified Formulation of Linear Iterative Learning Control," *Advances in the Astronautical Sciences*, Vol. 105, 2000, pp. 93-111.

[66] S. PANZIERI AND G. ULIVI, "Disturbance rejection of iterative learning control applied to trajectory tracking for a flexible manipulator," *Proceedings of the European Control Conference*, September 1995, pp. 2374-2379.

[67] M. NORRLÖF AND S. GUNNARSSON, “Disturbance aspects of iterative learning control,” *Engineering Applications of Artificial Intelligence*, Vol. 14, No. 1, 2001, pp. 87-94.

[68] R. MERRY, R. VAN DE MOLENGRAFT, AND M. STEINBUCH, “The influence of disturbances in iterative learning control,” *IEEE Conference on Control Applications*, August 2005, pp. 974-979.

[69] N. CABER, “Finite-Time Feedback Control Design,” *Bachelor Thesis*, Helmut Schmidt University, Hamburg, Germany and Dartmouth College, Hanover, NH, USA, 2014.

[70] S.J. OH, R.W. LONGMAN, AND Y.-P. HSIN, “The Possible Block Diagram Configurations for Repetitive Control to Cancel Periodic Plant and Measurement Disturbances,” *Proceedings of the AIAA/AAS Astrodynamics Specialist Conference*, Paper No. AIAA 2004-5297, August 2004.

[71] R.W. LONGMAN, J.W. YEOL, AND Y.S. RYU, “Placing the Repetitive Controller Inside or Outside the Feedback Loop: Simultaneously Achieving the Feedback and Repetitive Control Objectives,” *Advances in the Astronautical Sciences*, Vol. 127, 2007, pp. 1703-1722.

[72] R.W. LONGMAN, “On the Theory and Design of Linear Repetitive Control Systems,” *European Journal of Control*, Vol. 16, No. 5, 2010, pp. 447-496.

[73] M.Q. PHAN AND R.W. LONGMAN, “A Mathematical Theory of Learning Control for Linear Discrete Multivariable Systems,” *Proceedings of the AIAA/AAS Astrodynamics Conference*, August 1988, pp. 740-746.

[74] T. SUGIE AND T. ONO, “An iterative learning control law for dynamical systems,” *Automatica*, Vol. 27, No. 4, 1991, pp. 729-732.

[75] R. TOUSAIN AND E. VAN DER MECHÉ, “Design strategies for iterative learning control based on optimal control,” *Proceedings of the 40th IEEE Conference on Decision and Control*, December 2001, pp. 4463-4468.

[76] B.G. DIJKSTRA AND O.H. BOSGRA, “Extrapolation of optimal lifted system ILC solution, with application to a waferstage,” *Proceedings of the American Control Conference*, May 2002, pp. 2595-2600.

[77] M. CHO, Y. LEE, S. JOO, AND K.S. LEE, “Semi-empirical model-based multivariable iterative learning control of an RTP system,” *IEEE*

*Transactions on Semiconductor Manufacturing*, Vol. 18, No. 3, 2005, pp. 430-439.

[78] W.B.J. HAKVOORT, R.G.K.M. AARTS, J. VAN DIJK, AND J.B. JONKER, "Lifted system iterative learning control applied to an industrial robot," *Control Engineering Practice*, Vol. 16, No. 4, 2008, pp. 377-391.

[79] A. SCHÖLLIG AND R. D'ANDREA, "Optimization-based iterative learning control for trajectory tracking," *Proceedings of the European Control Conference*, August 2009, pp. 1505-1510.

[80] J.K. RICE AND M. VERHAEGEN, "A structured matrix approach to efficient calculation of LQG repetitive learning controllers in the lifted setting," *International Journal of Control*, Vol. 83, No. 6, 2010, pp. 1265-1276.

[81] Z. GENG, J.D. LEE, R.L. CARROLL, AND L.H. HAYNES, "Learning control system design based on 2-D theory – an application to parallel link manipulator," *Proceedings of the IEEE International Conference on Robotics and Automation*, May 1990, pp. 1510-1515.

[82] J.E. KUREK AND M.B. ZAREMBA, "Iterative learning control synthesis based on 2-D system theory," *IEEE Transactions on Automatic Control*, Vol. 38, No. 1, 1993, pp. 121-125.

[83] N. AMANN, D.H. OWENS, AND E. ROGERS, "2D systems theory applied to learning control systems," *Proceedings of the 33rd IEEE Conference on Decision and Control*, Vol. 2, December 1994, pp. 985-986.

[84] D.H. OWENS, N. AMANN, E. ROGERS, AND M. FRENCH, "Analysis of linear iterative learning control schemes – a 2D systems/repetitive processes approach," *Multidimensional Systems and Signal Processing*, Vol. 11, No. 1-2, 2000, pp. 125-177.

[85] S.SH. ALAVIANI, "Iterative learning control based on 2D system theory for singular linear continuous-time systems," *Proceedings of the IEEE International Conference on Control Applications*, August 2013, pp. 855-858.

[86] E.V. IFEACHOR AND B.W. JERVIS, *Digital Signal Processing: A Practical Approach*, Second Edition, Pearson Education Limited, 2002.

[87] A. CHINNAN AND R.W. LONGMAN, "Designing Iterative Learning Controllers from Limited Pulse Response Data," *Advances in the Astronautical Sciences*, Vol. 130, 2008, pp. 283-302.

[88] P.A. LEVOCI AND R.W. LONGMAN, “Intersample Error in Discrete Time Learning and Repetitive Control,” *Proceedings of the 2004 AIAA/AAS Astrodynamics Specialist Conference*, August 2004.

[89] R.W. LONGMAN, T.J. KWON, AND P.A. LEVOCI, “Making the Learning Control Problem Well Posed – Stabilizing Intersample Error,” *Advances in the Astronautical Sciences*, Vol. 123, 2006, pp. 1143-1162.

[90] Y. LI AND R.W. LONGMAN, “Addressing Problems of Instability in Intersample Error in Iterative Learning Control,” *Advances in the Astronautical Sciences*, Vol. 129, 2008, pp. 1571-1591.

[91] H.S. JANG AND R.W. LONGMAN, “A New Learning Control Law with Monotonic Decay of the Tracking Error Norm,” *Proceedings of the 32nd Annual Allerton Conference on Communication, Control, and Computing*, September 1994, pp. 314-323.

[92] J.S. JANG AND R.W. LONGMAN, "An Update on a Monotonic Learning Control Law and Some Fuzzy Logic Learning Gain Adjustment Techniques," *Advances in the Astronautical Sciences*, Vol. 90, 1996, pp. 301-318.

[93] G. LEE-GLAUSER, J.-N. JUANG, AND R.W. LONGMAN, “Comparison and Combination of Learning Controllers: Computational Enhancement and Experiments," *Journal of Guidance, Control and Dynamics*, Vol. 19, No. 5, 1996, pp. 1116-1123.

[94] K. KINOSHITA, T. SOGO, AND N. ADACHI, “Adjoint-Type Iterative Learning Control for a Single Link Flexible Arm,” *Proceedings of the 15th IFAC World Congress*, July 2002.

[95] H.S. JANG AND R.W. LONGMAN, "Design of Digital Learning Controllers using a Partial Isometry," *Advances in the Astronautical Sciences*, Vol. 93, 1996, pp. 137-152.

[96] K.CHEN AND R.W. LONGMAN, “Stability Issues using FIR Filtering in Repetitive Control,” *Advances in the Astronautical Sciences*, Vol. 112, 2002, pp. 1321-1339.

[97] K. ÅSTRÖM, P. HAGANDER, AND J. STRENBY, “Zeros of Sampled Systems,” *Proceedings of the 19th IEEE Conference on Decision and Control*, 1980, pp. 1077-1081.

[98] A. CHINNAN, M.Q. PHAN, AND R.W. LONGMAN, “Feedback and Iterative Learning Control with Disturbance Estimators,” *Advances in the Astronautical Sciences*, Vol. 152, 2014, pp. 747-766.

[99] C. COCHIOR, P.P.J. VAN DEN BOSCH, R. WAARSING, AND J. VERRIET, “Cold start control of industrial printers,” *Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, July 2012, pp. 123-128.

[100] J.G. ZIEGLER AND N.B. NICHOLS, “Optimum Settings for Automatic Controllers,” *Transactions of ASME,* Vol. 64, 1942, pp. 759-768.

[101] G.F. FRANKLIN, J.D. POWELL, AND A. EMAMI-NAEINI, *Feedback Control of Dynamic Systems*, Prentice Hall, New Jersey, 2002.

# APPENDIX

In this appendix detailed derivations intentionally omitted from the main body of the thesis are presented. These derivations were not included in the appropriate chapters due to their lengthy nature, but are provided here for those that may be interested.

## A: Mathematical Proof of Combination Controller Equivalence

This section mathematically proves the combination controller presented in Eq. (3.13) can be equivalently designed to learn through only current cycle error feedback or only previous cycle error feedback, in learning control applications where Eq. (2.3) can be assumed.

$$\textit{start with } (3.14) \quad \underline{u}_j = \underline{u}_{j-1} + \frac{\gamma}{1-\gamma} P^{-1} \underline{e}_j$$

$$\textit{rewrite } (2.3) \textit{ as} \quad \underline{e}_j = \underline{e}_{j-1} - P(\underline{u}_j - \underline{u}_{j-1})$$

$$\underline{u}_j = \underline{u}_{j-1} + \frac{\gamma}{1-\gamma} P^{-1}[\underline{e}_{j-1} - P(\underline{u}_j - \underline{u}_{j-1})]$$

$$(1-\gamma)\underline{u}_j = (1-\gamma)\underline{u}_{j-1} + \gamma P^{-1}\underline{e}_{j-1} - \gamma P^{-1} P(\underline{u}_j - \underline{u}_{j-1})$$

$$(1-\gamma)\underline{u}_j = (1-\gamma)\underline{u}_{j-1} + \gamma P^{-1}\underline{e}_{j-1} - \gamma \underline{u}_j + \gamma \underline{u}_{j-1}$$

$$\underline{u}_j = \underline{u}_{j-1} + \gamma P^{-1}\underline{e}_{j-1}$$

## B: Derivations of Causal Controllers

This section presents step-by-step derivations for the various causal controllers proposed in Chapter 3. Controllers without disturbance estimation, with one-repetition behind disturbance estimation, and with full disturbance estimation are derived.

### B.1: Non-Causal to Causal Controller Without Disturbance Estimation

$$
\begin{aligned}
\underline{u}_j &= R\underline{u}_j + S\underline{u}_{j-1} + G\underline{e}_j + L\underline{e}_{j-1} \\
&= (R_S + R_D)\underline{u}_j + S\underline{u}_{j-1} + (G_S + G_D)\underline{e}_j + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + G_D(\underline{y}^* - \underline{y}_j) + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + G_D(\underline{y}^* - (P_S + P_D)\underline{u}_j) + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j - G_DP_S\underline{u}_j - G_DP_D\underline{u}_j + L\underline{e}_{j-1} + G_D\underline{y}^* \\
&= (R_S - G_DP_S)\underline{u}_j + (R_D - G_DP_D)\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + L\underline{e}_{j-1} + G_D\underline{y}^*
\end{aligned}
$$

$$
(I - R_D + G_DP_D)\underline{u}_j = (R_S - G_DP_S)\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + L\underline{e}_{j-1} + G_D\underline{y}^*
$$

$$
\underline{u}_j = (I - R_D + G_DP_D)^{-1}[(R_S - G_DP_S)\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + L\underline{e}_{j-1} + G_D\underline{y}^*]
$$

$$
\underline{u}_j = \overline{R}\underline{u}_j + \overline{S}\underline{u}_{j-1} + \overline{G}\underline{e}_j + \overline{L}\underline{e}_{j-1} + G^*\underline{y}^*
$$

$$
\begin{aligned}
\overline{R} &= (I - R_D + G_DP_D)^{-1}(R_S - G_DP_S) \\
\overline{S} &= (I - R_D + G_DP_D)^{-1}S \\
\overline{G} &= (I - R_D + G_DP_D)^{-1}G_S \\
\overline{L} &= (I - R_D + G_DP_D)^{-1}L \\
G^* &= (I - R_D + G_DP_D)^{-1}G_D
\end{aligned}
$$

### B.2: Controller With One-Repetition Behind Disturbance Estimation

$$
\begin{aligned}
\underline{u}_j &= R\underline{u}_j + S\underline{u}_{j-1} + G\underline{e}_j + L\underline{e}_{j-1} \\
&= (R_S + R_D)\underline{u}_j + S\underline{u}_{j-1} + (G_S + G_D)\underline{e}_j + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + G_D(\underline{y}^* - \underline{y}_j) + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + G_D(\underline{y}^* - \left[(P_S + P_D)\underline{u}_j + \underline{\Delta}_j\right]) + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j - G_DP_S\underline{u}_j - G_DP_D\underline{u}_j - G_D\underline{\Delta}_j + L\underline{e}_{j-1} + G_D\underline{y}^* \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j - G_DP_S\underline{u}_j - G_DP_D\underline{u}_j - G_D(\underline{y}_{j-1} - P\underline{u}_{j-1}) + L\underline{e}_{j-1} + G_D\underline{y}^* \\
&= R_S\underline{u}_j + R_D\underline{u}_j + (S + G_DP)\underline{u}_{j-1} + G_S\underline{e}_j - G_DP_S\underline{u}_j - G_DP_D\underline{u}_j + L\underline{e}_{j-1} + G_D(\underline{y}^* - \underline{y}_{j-1}) \\
&= (R_S - G_DP_S)\underline{u}_j + (R_D - G_DP_D)\underline{u}_j + (S + G_DP)\underline{u}_{j-1} + G_S\underline{e}_j + L\underline{e}_{j-1} + G_D(\underline{y}^* - \underline{y}_{j-1})
\end{aligned}
$$

$$(I - R_D + G_D P_D)\underline{u}_j = (R_S - G_D P_S)\underline{u}_j + (S + G_D P)\underline{u}_{j-1} + G_S \underline{e}_j + L\underline{e}_{j-1} + G_D(\underline{e}_{j-1})$$
$$\underline{u}_j = (I - R_D + G_D P_D)^{-1}[(R_S - G_D P_S)\underline{u}_j + (S + G_D P)\underline{u}_{j-1} + G_S \underline{e}_j + (L + G_D)\underline{e}_{j-1}]$$

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{S}\underline{u}_{j-1} + \overline{G}\underline{e}_j + \overline{L}\underline{e}_{j-1}$$

$$\overline{R} = (I - R_D + G_D P_D)^{-1}(R_S - G_D P_S)$$
$$\overline{S} = (I - R_D + G_D P_D)^{-1}(S + G_D P)$$
$$\overline{G} = (I - R_D + G_D P_D)^{-1} G_S$$
$$\overline{L} = (I - R_D + G_D P_D)^{-1}(L + G_D)$$

### B.3: Controller With Full Disturbance Estimation

$$\begin{aligned}
\underline{u}_j &= R\underline{u}_j + S\underline{u}_{j-1} + G\underline{e}_j + L\underline{e}_{j-1} \\
&= (R_S + R_D)\underline{u}_j + S\underline{u}_{j-1} + (G_S + G_D)\underline{e}_j + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + G_D(\underline{y}^* - \underline{y}_j) + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j + G_D(\underline{y}^* - \left[(P_S + P_D)\underline{u}_j + \underline{\Delta}_j\right]) + L\underline{e}_{j-1} \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j - G_D(P_S + P_D)\underline{u}_j - G_D(D\underline{\Delta}_j + \underline{\varepsilon}_j) + L\underline{e}_{j-1} + G_D\underline{y}^* \\
&= R_S\underline{u}_j + R_D\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j - G_D P_S\underline{u}_j - G_D P_D\underline{u}_j - G_D D(\underline{y}_j - P\underline{u}_j) - G_D\underline{\varepsilon}_j + L\underline{e}_{j-1} + G_D\underline{y}^*
\end{aligned}$$

$$\begin{aligned}
(I - R_D + G_D P_D)\underline{u}_j &= \begin{array}{c}(R_S - G_D P_S + G_D DP)\underline{u}_j + S\underline{u}_{j-1} + G_S\underline{e}_j - G_D D(\underline{y}^* - \underline{e}_j) \\ -G_D\underline{\varepsilon}_j + L\underline{e}_{j-1} + G_D\underline{y}^*\end{array} \\
&= \begin{array}{c}(R_S - G_D P_S + G_D DP)\underline{u}_j + S\underline{u}_{j-1} + (G_S + G_D D)\underline{e}_j \\ -G_D(\underline{\Delta}_{j-1} - D\underline{\Delta}_{j-1}) + L\underline{e}_{j-1} + (G_D - G_D D)\underline{y}^*\end{array} \\
&= \begin{array}{c}(R_S - G_D P_S + G_D DP)\underline{u}_j + S\underline{u}_{j-1} + (G_S + G_D D)\underline{e}_j \\ -G_D(\underline{y}_{j-1} - P\underline{u}_{j-1}) + G_D D(\underline{y}_{j-1} - P\underline{u}_{j-1}) + L\underline{e}_{j-1} + (G_D - G_D D)\underline{y}^*\end{array} \\
&= \begin{array}{c}(R_S - G_D P_S + G_D DP)\underline{u}_j + (S + G_D P - G_D DP)\underline{u}_{j-1} + (G_S + G_D D)\underline{e}_j \\ -G_D(\underline{y}^* - \underline{e}_{j-1}) + G_D D(\underline{y}^* - \underline{e}_{j-1}) + L\underline{e}_{j-1} + (G_D - G_D D)\underline{y}^*\end{array} \\
&= \begin{array}{c}(R_S - G_D P_S + G_D DP)\underline{u}_j + (S + G_D P - G_D DP)\underline{u}_{j-1} + (G_S + G_D D)\underline{e}_j \\ +(L + G_D - G_D D)\underline{e}_{j-1} - (G_D - G_D D)\underline{y}^* + (G_D - G_D D)\underline{y}^*\end{array}
\end{aligned}$$

$$\underline{u}_j = \begin{array}{c}(I - R_D + G_D P_D)^{-1}[(R_S - G_D P_S + G_D DP)\underline{u}_j + (S + G_D P - G_D DP)\underline{u}_{j-1} \\ +(G_S + G_D D)\underline{e}_j + (L + G_D - G_D D)\underline{e}_{j-1}]\end{array}$$

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{S}\underline{u}_{j-1} + \overline{G}\underline{e}_j + \overline{L}\underline{e}_{j-1}$$

$$\overline{R} = (I - R_D + G_D P_D)^{-1}(R_S - G_D P_S + G_D DP)$$
$$\overline{S} = (I - R_D + G_D P_D)^{-1}(S + G_D P - G_D DP)$$
$$\overline{G} = (I - R_D + G_D P_D)^{-1}(G_S + G_D D)$$
$$\overline{L} = (I - R_D + G_D P_D)^{-1}(L + G_D - G_D D)$$

## C: Derivations of Tracking Error Propagation for Causal Controllers

This section provides step-by-step derivations of the tracking error propagation, in the repetition-domain, for the various causal controllers proposed in Chapter 3. The tracking error propagation is derived for the basic ILC law and combination controllers without disturbance estimation, with one-repetition behind disturbance estimation, and with full disturbance estimation.

### C.1: Basic ILC Law Error Propagation

$$\underline{u}_j = \underline{u}_{j-1} + L^*\underline{e}_{j-1}$$
$$\delta_j\underline{u} = L^*\underline{e}_{j-1}$$
$$P\delta_j\underline{u} = PL^*\underline{e}_{j-1}$$
$$\delta_j\underline{y} - \delta_j\underline{d} = PL^*\underline{e}_{j-1}$$
$$\delta_j\underline{y} = PL^*\underline{e}_{j-1} + \delta_j\underline{d}$$
$$-\delta_j\underline{e} = PL^*\underline{e}_{j-1} + \delta_j\underline{d}$$
$$\delta_j\underline{e} = -PL^*\underline{e}_{j-1} - \delta_j\underline{d}$$

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - \delta_j\underline{d}$$

### C.2: Error Propagation Without Disturbance Estimation

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + \overline{S}\underline{u}_{j-1} + \overline{L}\underline{e}_{j-1} + G^*\underline{y}^*$$
$$(I + G_D P_D - R_D)\underline{u}_j = (R_S - G_D P_S)\underline{u}_j + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\underline{y}^*$$
$$\underline{u}_j = (R_S - G_D P_S)\underline{u}_j + (R_D - G_D P_D)\underline{u}_j + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\underline{y}^*$$
$$\underline{u}_j = \big(R_S + R_D - G_D(P_S + P_D)\big)\underline{u}_j + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\underline{y}^*$$
$$\underline{u}_j = R\underline{u}_j - G_D P\underline{u}_j + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\underline{y}^*$$
$$\underline{u}_j = R\underline{u}_j - G_D\left(\underline{y}_j - \underline{d}\right) + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\underline{y}^*$$
$$\underline{u}_j = R\underline{u}_j + G_D\left(\underline{y}^* - \underline{y}_j\right) + G_S\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\underline{d}$$

$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\underline{d}$$
$$(I-R)\underline{u}_j = G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\underline{d}$$
$$S\underline{u}_j = G\underline{e}_j + S\underline{u}_{j-1} + \big((GP+I-R)L^* - G\big)\underline{e}_{j-1} + G_D\underline{d}$$
$$S\delta_j\underline{u} = G\underline{e}_j - G\underline{e}_{j-1} + (GP+S)L^*\underline{e}_{j-1} + G_D\underline{d}$$
$$PS\delta_j\underline{u} = PG\underline{e}_j - PG\underline{e}_{j-1} + P(S+GP)L^*\underline{e}_{j-1} + PG_D\underline{d}$$
$$S\delta_j\underline{y} = PG\underline{e}_j - PG\underline{e}_{j-1} + (S+PG)PL^*\underline{e}_{j-1} + PG_D\underline{d}$$
$$-S\delta_j\underline{e} = PG\delta_j\underline{e} + (S+PG)PL^*\underline{e}_{j-1} + PG_D\underline{d}$$
$$-(S+PG)\delta_j\underline{e} = (S+PG)PL^*\underline{e}_{j-1} + PG_D\underline{d}$$
$$\delta_j\underline{e} = -PL^*\underline{e}_{j-1} - (S+PG)^{-1}PG_D\underline{d}$$

$$\underline{e}_j = (I-PL^*)\underline{e}_{j-1} - (S+PG)^{-1}PG_D\underline{d}$$

### C.3: Error Propagation With One-Repetition Behind Disturbance Estimation

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + \overline{S}\underline{u}_{j-1} + \overline{L}\underline{e}_{j-1}$$
$$(I + G_DP_D - R_D)\underline{u}_j = (R_S - G_DP_S)\underline{u}_j + G_S\underline{e}_j + (S+G_DP)\underline{u}_{j-1} + (L+G_D)\underline{e}_{j-1}$$
$$\underline{u}_j = (R_S - G_DP_S)\underline{u}_j + (R_D - G_DP_D)\underline{u}_j + G_S\underline{e}_j + (S+G_DP)\underline{u}_{j-1} + (L+G_D)\underline{e}_{j-1}$$
$$\underline{u}_j = R\underline{u}_j - G_DP\underline{u}_j + G_S\underline{e}_j + (S+G_DP)\underline{u}_{j-1} + (L+G_D)\underline{e}_{j-1}$$
$$\underline{u}_j = R\underline{u}_j - G_D\left(\underline{y}_j - \underline{\Delta}_j\right) + G_S\underline{e}_j + (S+G_DP)\underline{u}_{j-1} + (L+G_D)\underline{e}_{j-1} + G_D\underline{y}^* - G_D\underline{y}^*$$
$$\underline{u}_j = R\underline{u}_j - G_D\left(\underline{y}_j - \big(\underline{d} + \underline{d}_j\big)\right) + G_S\underline{e}_j + (S+G_DP)\underline{u}_{j-1} + (L+G_D)\underline{e}_{j-1} + G_D\underline{y}^* - G_D\underline{y}^*$$
$$\underline{u}_j = R\underline{u}_j + G_D\left(\underline{y}^* - \underline{y}_j\right) + G_S\underline{e}_j + (S+G_DP)\underline{u}_{j-1} + (L+G_D)\underline{e}_{j-1} + G_D\big(\underline{d} + \underline{d}_j\big) - G_D\underline{y}^*$$
$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_DP\underline{u}_{j-1} + G_D\left(\underline{y}^* - \underline{y}_{j-1}\right) + G_D\big(\underline{d} + \underline{d}_j\big) - G_D\underline{y}^*$$
$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\left(\underline{y}_{j-1} - \underline{\Delta}_{j-1}\right) - G_D\underline{y}_{j-1} + G_D\big(\underline{d} + \underline{d}_j\big)$$
$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\left(\underline{y}_{j-1} - \big(\underline{d} + \underline{d}_{j-1}\big)\right) - G_D\underline{y}_{j-1} + G_D\big(\underline{d} + \underline{d}_j\big)$$
$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} - G_D\big(\underline{d} + \underline{d}_{j-1}\big) + G_D\big(\underline{d} + \underline{d}_j\big)$$
$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\delta_j\underline{d}$$
$$(I-R)\underline{u}_j = G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D\delta_j\underline{d}$$
$$S\underline{u}_j = G\underline{e}_j + S\underline{u}_{j-1} + \big((GP+I-R)L^* - G\big)\underline{e}_{j-1} + G_D\delta_j\underline{d}$$
$$S\left(\delta_j\underline{y} - \delta_j\underline{d}\right) = PG\underline{e}_j - PG\underline{e}_{j-1} + (S+PG)PL^*\underline{e}_{j-1} + PG_D\delta_j\underline{d}$$
$$-(S+PG)\delta_j\underline{e} = (S+PG)PL^*\underline{e}_{j-1} + (S+PG_D)\delta_j\underline{d}$$

$$\underline{e}_j = (I-PL^*)\underline{e}_{j-1} - (S+PG)^{-1}(S+PG_D)\delta_j\underline{d}$$

### C.4: Error Propagation With Full Disturbance Estimation

$$\underline{u}_j = \overline{R}\underline{u}_j + \overline{G}\underline{e}_j + \overline{S}\underline{u}_{j-1} + \overline{L}\underline{e}_{j-1}$$
$$(I + G_DP_D - R_D)\underline{u}_j = (R_S - G_DP_S + G_DDP)\underline{u}_j + (G_S + G_DD)\underline{e}_j +$$
$$(S + G_DP - G_DDP)\underline{u}_{j-1} + (L + G_D - G_DD)\underline{e}_{j-1}$$
$$\underline{u}_j = R\underline{u}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} - G_DP\underline{u}_j + G_DDP\underline{u}_j + (G_S + G_DD)\underline{e}_j +$$

$$(G_D P - G_D DP)\underline{u}_{j-1} + (G_D - G_D D)\underline{e}_{j-1}$$

$$\underline{u}_j = R\underline{u}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} - G_D(I-D)P\underline{u}_j + (G_S + G_D D)\underline{e}_j + G_D(I-D)P\underline{u}_{j-1} + (G_D - G_D D)\underline{e}_{j-1}$$

$$\underline{u}_j = R\underline{u}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} - G_D(I-D)\left(\underline{y}_j - \underline{\Delta}_j\right) + (G_S + G_D D)\underline{e}_j + G_D(I-D)\left(\underline{y}_{j-1} - \underline{\Delta}_{j-1}\right) + (G_D - G_D D)\underline{e}_{j-1}$$

$$\underline{u}_j = R\underline{u}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} - G_D(I-D)\left(\underline{y}_j - \left(\underline{d} + \underline{d}_j\right)\right) + (G_S + G_D D)\underline{e}_j + G_D(I-D)\left(\underline{y}_{j-1} - \left(\underline{d} + \underline{d}_{j-1}\right)\right) + (G_D - G_D D)\underline{e}_{j-1}$$

$$\underline{u}_j = R\underline{u}_j + G_S\underline{e}_j + S\underline{u}_{j-1} + (L + G_D)\underline{e}_{j-1} - G_D(I-D)\left(\delta_j\underline{y} - \delta_j d\right) + G_D D\delta_j\underline{e}$$

$$\underline{u}_j = R\underline{u}_j + G_S\underline{e}_j + S\underline{u}_{j-1} + (L + G_D)\underline{e}_{j-1} + G_D(I-D)\left(\delta_j\underline{e} + \delta_j d\right) + G_D D\delta_j\underline{e}$$

$$\underline{u}_j = R\underline{u}_j + G_S\underline{e}_j + S\underline{u}_{j-1} + (L + G_D)\underline{e}_{j-1} + G_D(I-D)\delta_j\underline{e} + G_D D\delta_j\underline{e} + G_D(I-D)\delta_j d$$

$$\underline{u}_j = R\underline{u}_j + G_S\underline{e}_j + S\underline{u}_{j-1} + (L + G_D)\underline{e}_{j-1} + G_D\delta_j\underline{e} + G_D(I-D)\delta_j d$$

$$\underline{u}_j = R\underline{u}_j + G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D(I-D)\delta_j d$$

$$(I-R)\underline{u}_j = G\underline{e}_j + S\underline{u}_{j-1} + L\underline{e}_{j-1} + G_D(I-D)\delta_j d$$

$$S\underline{u}_j = G\underline{e}_j + S\underline{u}_{j-1} + \left((GP + I - R)L^* - G\right)\underline{e}_{j-1} + G_D(I-D)\delta_j d$$

$$S\left(\delta_j\underline{y} - \delta_j\underline{d}\right) = PG\underline{e}_j - PG\underline{e}_{j-1} + (S + PG)PL^*\underline{e}_{j-1} + PG_D(I-D)\delta_j d$$

$$-(S + PG)\delta_j\underline{e} = (S + PG)PL^*\underline{e}_{j-1} + \left(S + PG_D(I-D)\right)\delta_j\underline{d}$$

$$\underline{e}_j = (I - PL^*)\underline{e}_{j-1} - (S + PG)^{-1}\left(S + PG_D(I-D)\right)\delta_j\underline{d}$$