Examination of Bandwidth Enhancement and Circulant Filter Frequency Cutoff Robustification in Iterative Learning Control

Tianyi Zhang

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
under the Executive Committee
of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2021

Abstract

Examination of Bandwidth Enhancement and Circulant Filter Frequency Cutoff Robustification in Iterative Learning Control

Tianyi Zhang

The iterative learning control (ILC) problem considers control tasks that perform a specific tracking command, and the command is to be performed is many times. The system returns to the same initial conditions on the desired trajectory for each repetition, also called run, or iteration. The learning law adjusts the command to a feedback system based on the error observed in the previous run, and aims to converge to zero-tracking error at sampled times as the iterations progress. The ILC problem is an inverse problem: it seeks to converge to that command that produces the desired output. Mathematically that command is given by the inverse of the transfer function of the feedback system, times the desired output. However, in many applications that unique command is often an unstable function of time. A discrete-time system, converted from a continuous-time system fed by a zero-order hold, often has non-minimum phase zeros which become unstable poles in the inverse problem. An inverse discrete-time system will have at least one unstable pole, if the pole-zero excess of the original continuous-time counterpart is equal to or larger than three, and the sample rate is fast enough. The corresponding difference equation has roots larger than one, and the homogeneous solution has components that are the values of these poles to the power of k, with k being the time step. This creates an unstable command growing in magnitude with time step. If the ILC law aims at zero-tracking error for such systems, the command produced by the ILC iterations will ask for a

command input that grows exponentially in magnitude with each time step. This thesis examines several ways to circumvent this difficulty, designing filters that prevent the growth in ILC.

The sister field of ILC, repetitive control (RC), aims at zero-error at sample times when tracking a periodic command or eliminating a periodic disturbance of known period, or both. Instead of learning from a previous run always starting from the same initial condition, RC learns from the error in the previous period of the periodic command or disturbance. Unlike ILC, the system in RC eventually enters into steady state as time progresses. As a result, one can use frequency response thinking. In ILC, the frequency thinking is not applicable since the output of the system has transients for every run. RC is also an inverse problem and the periodic command to the system converges to the inverse of the system times the desired output. Because what RC needs is zero error after reaching steady state, one can aim to invert the steady state frequency response of the system instead of the system transfer function in order to have a stable solution to the inverse problem. This can be accomplished by designing a Finite Impulse Response (FIR) filter that mimics the steady state frequency response, and which can be used in real time.

This dissertation discusses how the digital feedback control system configuration affects the locations of sampling zeros and discusses the effectiveness of RC design methods for these possible sampling zeros. The sampling zeros are zeros introduced by the discretization process from continuous-time system to the discrete-time system. In the RC problem, the feedback control system can have sampling zeros outside the unit circle, and they are challenges for the RC law design. Previous research concentrated on the situation where the sampling zeros of the feedback control system come from a zero-order hold on the input of a continuous-time feedback system, and studied the influence of these zeros including the influence of these sampling zeros as the sampling rate is changed from the asymptotic value of sample time interval approaching

zero. Effective RC design methods are developed and tested based for this configuration. In the real world, the feedback control system may not be the continuous-time system. Here we investigate the possible sampling zero locations that can be encountered in digital control systems where the zero-order hold can be in various possible places in the control loop. We show that various new situations can occur. We discuss the sampling zeros location with different feedback system structures, and show that the RC design methods still work. Moreover, we compare the learning rates of different RC design methods and show that the RC design method based on a quadratic fit of the reciprocal of the steady state frequency response will have the desired learning rate features that balance the robustness with efficiency.

This dissertation discusses the steady-state response filter of the finite-time signal used in ILC. The ILC problem is sensitive to model errors and unmodelled high frequency dynamics, thus it needs a zero-phase low-pass filter to cutoff learning for frequencies where there is too much model inaccuracy for convergence. But typical zero-phase low-pass filters, like Filtfilt used by MATLAB, gives the filtered results with transients that can destabilize ILC. The associated issues are examined from several points of view. First, the dissertation discusses use of a partial inverse of the feedback system as both learning gain matrix and a low-pass filter to address this problem The approach is used to make a partial system inverse for frequencies where the model is accurate, eliminating the robustness issue. The concept is used as a way to improve a feedback control system performance whose bandwidth is not as high as desired. When the feedback control system design is unable to achieve the desired bandwidth, the partial system inverse for frequency in a range above the bandwidth can boost the bandwidth. If needed ILC can be used to further correct response up to the new bandwidth.

The dissertation then discusses Discrete Fourier Transform (DFT) based filters to cut off the learning at high frequencies where model uncertainty is too large for convergence. The concept of a low pass filter is based on steady state frequency response, but ILC is always a finite time problem. This forms a mismatch in the design process, and we seek to address this. A math proof is given showing the DFT based filters directly give the steady-state response of the filter for the finite-time signal which can eliminate the possibility of instability of ILC. However, such filters have problems of frequency leakage and Gibbs phenomenon in applications, produced by the difference between the signal being filtered at the start time and at the final time, This difference applies to the signal filtered for nearly all iterations in ILC. This dissertation discusses the use of single reflection that produced a signal that has the start time and end times matching and then using the original signal portion of the result. In addition, a double reflection of the signal is studied that aims not only to eliminate the discontinuity that produces Gibbs, but also aims to have continuity of the first derivative. It applies a specific kind of double reflection. It is shown mathematically that the two reflection methods reduce the Gibbs phenomenon. A criterion is given to determine when one should consider using such reflection methods on any signal. The numerical simulations demonstrate the benefits of these reflection methods in reducing the tracking error of the system.

Table of Contents

List of Charts and Graphs

Acknowledgments

The research and the dissertation could not be done without the help from my academic advisor Prof. Richard W. Longman.

Prof. Longman is not only an advisor, but also a tutor and mentor. I still recalled the first course that Prof. Longman taught me, which is the introduction to modern control. He used the clear, concise, yet accurate words to efficiently and elegantly illustrate complex concepts that puzzle me for a while. The dead words on the book become as vivid explanations and life experiences that help me get the essence of the concepts instead of being overwhelmed by definitions and equations. It is a magic. As his PhD student, he taught me more than the academic part. I am always astonished by his curiosity and love for the research he is working on at his age, and he always want to move the research edge outwards for one more millimeter. Ideas burst out when we had weekly meeting, and many of his questions motivate me to think a little deeper and look a little wider. Prof. Longman has a notepad of the time table of weekly meeting he holds, and it is often full. But he is energetic in every meeting and keep contributing to the topics. Prof. Longman is generous, patient, and tolerant. I never recalled a moment that he was disappointed with me for my weekly research, and when I felt overwhelmed and disappointed with myself for the work I did, he would be the first one to comfort and encourage me to move one. He is a role model to me not only in the attitudes towards the research but also attitudes toward life and people.

I would like to extend thanks to the other members of my dissertation committee, Prof. Raimondo Betti, Prof. Homayoon Beigi, Prof. Nicolas W. Chbat and Prof. Minh Phan. I thank all my lab members Dr. Zhu, Dr. Song, and Dr. Ji for their companionship in these years. I would also thank the Department of Mechanical Engineering at Columbia supporting my studies.

Dedication

The story starts from two families.

My father grew up in a big family. My grandparents have seven children, and my father has two elder sisters, and four little brothers. It was hard to feed a large family in the famine era from 1959 to 1961, and my father was born in 1959. My grandparents did not read or write in their early 30s, and they learned to read and write later on, but they know knowledge made a difference. My grandparents worked hard to support the family so that every child can attend school at that time. However, the social turmoil and financial burden hovered above the family. My two aunts started to work after they graduated from high school to provide financial support to other family members. My father, too graduated from high school, became a primary school teacher at that time since all college admission stopped during 1960s and 1970s in China. Not until 1977, the college restarted the admission, and my father seized that opportunity to become one of 270 thousand freshmen admitted that year after the 10-year pause in college admission in a nation of one billion citizens. My father got his admission letter while he was digging a canal for the local farm. The college was free at that time with financial support to every admitted student in China, but still many admitted students like my father were worrying about the transportation fee since their family could not afford. The government already had a solution for that: each admission letter can be redeemed for a free train ticket at the railway stations to ensure that every freshman can arrive at college even from the most remote area. My father got onto his train and started his college.

My father was 19 at that time. His oldest classmate was already 35 and was a father of three children, and his youngest classmate was just above 15. The restart of college admission ignited the passion for the knowledge again after 10-year pause of admission, and everyone was

taking their chances, and all admitted students are among top in their generations. All the top students gathered around the college, and my father was still among the best of the best. When I did the research for my family tree, I accidentally found a piece of thesis with grades for my father's sophomore. I quickly found the name of my father: he was the 1st among more than 200 students in his major. I also found another piece of thesis: his admission letter of the master program. But I knew my father started to work right after his undergraduate. I never asked my father about it, but I learned from his classmates that at that time they knew my father were admitted to the master program but gave up later on even if he was offered the scholarship. The financial burden from the family, I guess, made my father start to work just like what my two aunts did for him. I think this is the biggest regret of my father to stop his academia.

My mother grew up in a smaller family. I only have two uncles on my mother's side. But my grandparents, on my mother's side, were indeed from the bottom of the society. My grandfather told me that his family never had a piece of land as peasants and they lived on selling hand-pulled noodle in the local market. My grandmother's family was even financially worse. My grandmother could only read and write her name even for now. My grandfather, in his early day, did every short-term job as a low-wage laborer I could ever imagined: painting walls, digging canals, and laying bricks. My grandfather first long-term job was a security guard and secretary at the same time for delivering important thesis work for the new Chinese government in early 1950s. My grandfather was trusted because he could NOT read, and it is the most characteristic one wants for a delivery man of classified files. Later on, his boss told him knowledge made a difference, my grandfather started to learn read and write and even got a high school diploma in his forties. He became one of guys that changed his fate, far better than his co-workers in his early days. I asked him what made that. My grandfather said he had two things to

thank for. The first one is his courage to leave his rural village in his hometown. The second one is his persistence in learning. “Live and learn”, he said to me.

I think, my grandparents on my father side never ever thought that their grandson could be a PhD candidate in Columbia University in a nation they never heard of when they decided to send every child to school with huge financial burden. Neither did my grandparents on my mother side when they left their village to do every short-term job they could find in his early days. So did my father when he decided to refuse the offer and started to work to provide financial support for his family right after his undergraduate. When it comes to me who is the one I want to dedicate to, all these pieces of stories and people come to my mind. These are only a small fraction of stories of two families I decided to write here in my dissertation. I want to end it with one quote from a hospital in Queens where I was on a bus. It wrote, “amazing things happen here”.

Preface

The dissertation has six chapters. The first chapter introduces the basics of iterative learning control (ILC) and repetitive control (RC), and lays the foundation for the readers for the next chapters.

The second chapter discusses the sampling zero locations introduced from conversion of the continuous-time transfer functions to the discrete-time transfer functions using zero-order hold in different feedback systems, and the effectiveness of RC design methods to compensate for these sampling zeros in RC. The third chapter discusses a partial inverse of the system based on the singular value decomposition, and its potential to use as both the learning gain matrix and cutoff filter in ILC, and its potential to use as a prefilter to increase the bandwidth of a feedback control system. The fourth chapter introduces two DFT-based filters, Circulant Filter and Cliff Filter, and gives the math proof that both filters give the steady-state response of a filter for a finite-time input. The fifth chapter discusses Gibbs phenomenon in Discrete Fourier Transfrom (DFT) based filters used in ILC, and proves that single reflection and double reflection of the input signal can reduce the influence of Gibbs phenomenon, and shows that both single/double reflection methods can reduce the tracking error of ILC when filtering is needed for robustness. The sixth chapter is a summary of discussions in the dissertation.

Chapter 1: Introduction

Iterative Learning Control (ILC) considers control systems that perform a specific tracking command repeatedly aiming at zero-tracking error at sampled times. The control system returns to the same initial condition before the start of each run (iteration, or repetition). A sister field of ILC is repetitive control (RC). Repetitive control aims to track periodic commands, and uses the error in the last period to make adjustments to the command in the present period in order to converge to zero-tracking error at the sample times. ILC has transients in every run, and design methods in ILC are based on time domain models. But frequency design methods rigorously apply in RC since the transients become negligible as time progresses in RC applications. Both are in the discrete time domain. Early literature about ILC is motivated by robots doing repetitive tasks, Arimoto *el al*., Casalino and Bartolini, and Craig, are all independent contributors to the early development of ILC [1-3]. The origins of repetitive control had different motivation, and early works include Inoue *et al*., Omada *et al*., and Hara *et al* [4-6].

1.1 Iterative Control Basics

Iterative Learning Control (ILC) adjusts the command to a feedback control system based on the errors in the previous iteration aiming at zero tracking error at sampled times. A Single Input Single Output (SISO) linear discrete-time feedback control system is written in the following state-space equation,

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \quad k = 0, 1, 2, \dots, N-1 \\ y(k) &= Cx(k) \qquad\qquad\quad k = 1, 2, \dots, N \end{aligned} \tag{1-1}$$

We use underbar to denote the history of a signal, and all signals are expressed as column vectors. The lower-case letter subscript is used to denote its iteration number in ILC. In this

dissertation, iteration is an interchangeable word with run or repetition to describe one-time start and end of an operation, and in this dissertation, we use iteration for ILC, and use repetition for RC. The column vector $\underline{y}^*$ is the desired output of length N, $\underline{y}_j$ is the output history at iteration j, $\underline{e}_j$ is the error history at iteration j given by $\underline{e}_j = \underline{y}^* - \underline{y}_j$, and $\underline{u}_j$ is the input history at iteration j. There is a one time-step delay from input to output because of the zero-order hold assumption for the feedback system.

$$\begin{aligned} \underline{y}^* &= [y^*(1), y^*(2), \dots, y^*(N)]^T \\ \underline{y}_j &= \left[y_j(1), y_j(2), \dots, y_j(N)\right]^T \\ \underline{e}_j &= \left[e_j(1), e_j(2), \dots, e_j(N)\right]^T \\ \underline{u}_j &= \left[u_j(0), u_j(1), \dots u_j(N-1)\right]^T \end{aligned} \tag{1-2}$$

The input/output relationship of the feedback system in Equation (1-1) for any iteration j is expressed in its convolution sum solution form as

$$y_j(k) = CA^k x(0) + \sum_{i=0}^{N-1} CA^{k-i-1} B u_j(i) \tag{1-3}$$

or

$$\underline{y}_j = P\underline{u}_j + Ox(0) \tag{1-4}$$

where matrix P is a lower triangular Toeplitz matrix whose non-zero entries are unit pulse responses of the system, and O is the observability matrix

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

$$O = [CA, CA^2, CA^3, \dots, CA^N]^T$$

A general linear iterative learning control law takes the form below

$$\underline{u}_{j+1} = \underline{u}_j + L\underline{e}_j \tag{1-6}$$

where L is the learning gain matrix chosen by the designer. Previous literature has discussed several designs of L. For example, a pure integral control based ILC design is given by $L = diag(\phi, \phi, ...)$[7]. The contraction mapping law or P transpose law uses the learning gain matrix in the form $L = P^T$, where P is the P matrix of feedback system as shown in Equation (1-5), and this L produces the monotonic decay of the tracking error in terms of the Euclidean norm [8]. The partial isometry law uses $L = VU^T$, where U and V are singular vectors from the singular value decomposition (SVD) of $P = USV^T$, and it too produces the monotonic decay of the tracking error [9]. These methods discussed above are all special cases of the learning gain matrices from the general ILC law based on the quadratic cost of errors and commands with tuning parameters [10]. One can add a gain in front of L to turn down the learning rate based on robustness considerations.

From Equations (1-3), one computes for $\underline{y}_j - \underline{y}_{j-1}$. The ILC problem assumes that each iteration starts from the same initial conditions. Then its $\underline{y}_j - \underline{y}_{j-1} = P(\underline{u}_j - \underline{u}_{j-1})$, and the left side of this equation, rewrite it as $\underline{y}_j - \underline{y}_{j-1} = \underline{y}_j - \underline{y}^* + \underline{y}^* - \underline{y}_{j-1} = \underline{-e}_j + \underline{e}_{j-1}$; on the right side of this equation, use Equation (1-6) to produce the relationship of the error history from iteration to iteration as,

$$\underline{e}_j = (I - PL)\underline{e}_{j-1} = (I - PL)^j \underline{e}_0 \tag{1-7}$$

The learning gain matrix L makes the tracking error converge to zero for all possible initial conditions, if and only if the absolute value of all eigenvalues λ_i of the following matrix are less than one [11].

$$|\lambda_i(I - PL)| < 1 \quad i = 1,2, \dots, N \tag{1-8}$$

A sufficient condition that guarantees monotonic decay of the error in the sense of the Euclidean norm asks that the singular values σ_i satisfy [11],

$$|\sigma_i(I - PL)| < 1 \quad i = 1,2, \dots, N \tag{1-9}$$

Typical ILC applications need a zero-phase low-pass filter to cut off the learning at high frequency to increase the robustness of the system. Consider that the control action computed in Equation (1-6) goes through a zero-phase low-pass filter written as an *N* by *N* matrix *F*

$$\underline{u}_{j+1} = F(\underline{u}_j + L\underline{e}_j) \tag{1-10}$$

Recalling Equation(1-4) and the definition of $\underline{e}_j = \underline{y}^* - \underline{y}_j$, one can write,

$$\begin{gathered} \underline{e}_j = y^* - y_j = -Pu_j + y^* - Ox(0) = -Pu_j + f \\ f = y^* - Ox(0) \end{gathered} \tag{1-11}$$

Plug Equation (1-11) to Equation (1-10),

$$\underline{u}_{j+1} = F\underline{u}_j + FL\left(-P\underline{u}_j + f\right) = F(I - LP)\underline{u}_j + FLf \tag{1-12}$$

Assuming that $\underline{u}_{j+1} = \underline{u}_j = u_\infty$ after reaching steady state, then,

$$\underline{u}_\infty = [I - F(I - LP)]^{-1}FLf \tag{1-13}$$

The output and tracking error of the system at steady state can be written as [12],

$$\begin{gathered} \underline{y}_\infty = P[I - F(I - LP)]^{-1}FLf + Ox(0) \\ \underline{e}_\infty = \{I - P[I - F(I - LP)]^{-1}FL\}f \end{gathered} \tag{1-14}$$

1.2 Repetitive Control Basics

Repetitive control (RC) seeks to make a feedback control system converge to zero tracking error at each sample time for a periodic command and/or periodic disturbance. Figure 1-1 shows the typical structure of a repetitive control system. Repetitive controller $R(z)$ modifies

the command to the existing feedback system $G(z)$ in the current period based on the error in the previous period.

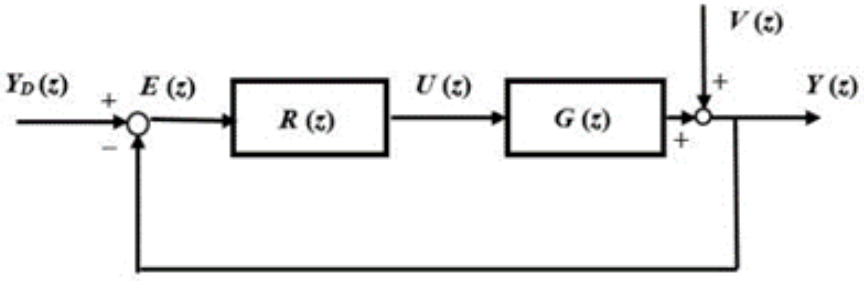

Figure 1- 1: A typical repetitive control system block diagram.

The simplest form of repetitive controller makes the equivalent of integral control action in classical control but for t makes an integral controller for each frequency component. It adjusts the command $u(k)$ based on the command $u(k-p)$ and the error $e(k-p+1)$ in the previous period, p is the period of the command as shown in Equation (1-15). It only uses one repetitive control gain ϕ to adjust the command at each time step

$$u(k) = u(k-p) + \phi e(k-p+1) \tag{1-15}$$

The $u(k)$ is the command to the feedback system at time step k, e is the error of the desired output minus the output, and p is the period of the disturbance or command. The error e is the error shifted forward one time step in the previous period to reflect the fact that one first observed a change of the output one step after the zero-order hold input is changed. Equation (1-15) creates a sum of errors in previous periods, a discrete form of integral, applied to each time step in the current time step k.

A more general form of repetitive control law in the z-domain is a compensator $R(z)$

$$U(z) = R(z)E(z) = \left[\frac{F(z)}{z^p - 1}\right] E(z) \tag{1-16}$$

Equation (1-15) is a special case of Equation (1-16), where $F(z) = \phi z$. One might think to use Nyquist stability criteria to determine the stability of the system in Figure 1-1, but it has computational difficulty because of the p poles on the unit circle. With poles on the stability boundary, the Nyquist contour must go around each pole. Huang and Longman give a method to determine the stability of RC [13].

The output of the system in Figure 1-1 is $Y(z) = G(z)U(z) + V(z)$, where $V(z)$ is the constant or periodic disturbance. One uses Equation (1-6) and has $Y(z) = \frac{FG}{z^p - 1}E(z) + V(z)$. Use $Y^*(z)$, the desired output, to deduct both sides of above equation, and recall $E(z) = Y^*(z) - Y(z)$. Rearrange the equation and one has the following result

$$\{1 - z^{-p}[1 - G(z)F(z)]\}E(z) = (1 - z^{-p})[Y_D(z) - V(z)] \qquad (1\text{-}17)$$

The term z^{-p} is a backward shift of one period. Since both $Y_D(z)$ and $V(z)$ are periodic functions of p time step, $Y_D(z)$ and $z^{-p}Y_D(z)$ are the same and so are $V(z)$ and $z^{-p}V(z)$. The right-hand side of Equation (1-17) is zero. Equation (1-17) then becomes a homogeneous equation for tracking error $E(z)$, and if one ensures all the roots of the characteristic polynomial, which is the numerator polynomial in the curly bracket, have magnitude less than 1, then the solution of $E(z)$ goes to zero for all possible initial conditions, as the time steps go to infinity. Hence, it achieves the zero-tracking error of the periodic command and a perfect cancellation of the influence of the periodic disturbance.

One can rewrite Equation (1-17) as

$$z^p E(z) = [1 - G(z)F(z)]E(z) \qquad (1\text{-}18)$$

The term z^p shifts $E(z)$ forward by one period. The left-hand side of Equation (1-18) is the error in the next period, and the right-hand side of Equation (1-18) can be interpreted as the transfer function $[1 - G(z)F(z)]$ times the current error $E(z)$. One might consider the term in the square

bracket in Equation (1-18) as a transfer function from one period to the next, and then substitute $z = e^{i\omega T}$ to form the frequency transfer function. This suggests that if one can ensure the magnitude of the frequency transfer function satisfies,

$$\left|1 - G\left(e^{i\omega T}\right)F(e^{i\omega T})\right| < 1 \quad \forall\omega \tag{1-19}$$

for all frequencies up to Nyquist frequency, then the magnitude of all frequency components of the error will decay from one period to the next indicating asymptotic stability. However, this thinking is flawed because the frequency response is a steady state response, and it is being used to model the change of error with time. One can only say that the magnitude on the left-hand side of Equation (1-19) for a given frequency ω is an estimate of the decay in amplitude from one period to the next under a quasi-steady-state assumption, i.e. that the decay rate is so slow that we can approximate the response in each period as if it were in steady state.

Equation (1-19) gives a good estimation of the decay rate of the error, also called the learning rate, from one period to the next period, and that the quasi-static assumption here is not a serious issue for a reasonable size of p. Equation (1-19) is both necessary and sufficient for the asymptotic stability for all possible p [13]. Thus Equation (1-19) not only determines the stability of the system, but also quantifies the error decay for each frequency component per period, indicating the learning rate (convergence rate) to zero tracking error.

One repetitive controller design in Equation (1-16) uses an FIR filter simply making a linear combination of errors observed before and after the time step k in the previous period [14]. The corresponding *z*-transformation has the form

$$F(z) = a_1 z^{m-1} + a_2 z^{m-2} + \cdots a_m z^0 + \cdots a_{n-1} z^{m-(n-1)} + a_n z^{m-n} \tag{1-20}$$

The z^0 term corresponds to the error one period back at time step k, $e(k-p)$. The rest of the terms are for errors before and after $e(k-p)$. The FIR design parameters are the coefficients in front of each term and the values of *n* and *m*. If one rewrites Equation (1-20) as

$$F(z) = (a_1 z^{n-1} + a_2 z^{n-2} + \cdots a_m z^{n-m} + \cdots a_{n-1} z^1 + a_n z^0)/z^{n-m} \qquad (1\text{-}21)$$

The value *n* determines the number of zeros in the FIR filter, and it is equal to $n-1$. The number of poles at the origin is equal to $n-m$. Then, $R(z)$ has the form,

$$R(z) = \frac{F(z)}{z^p - 1} = \frac{a_1 z^{n-1} + a_2 z^{n-2} + \cdots + a_m z^{n-m} + \cdots + a_{n-1} z^1 + a_n z^0}{z^{n-m}(z^p - 1)} \qquad (1\text{-}22)$$

Chapter 2: Influence of Discrete Control System Structure on Closed Loop Location of Sampling Zeros

Discrete-time equivalents of continuous-time models using zero-order hold usually have zeros introduced outside the unit circle, making the inverse model unstable. Such zeros introduced during the discretization are often called sampling, and sampling zero asymptotic locations are known in general. The sampling zero outside the unit circle is a challenge in RC design since it makes the inverse system model unstable. Previous literature has developed several RC design methods designing an FIR filter that compensates these sampling zeros outside the unit circle by introducing extra zeros outside the unit circle with unique patterns [14-16]. The RC design methods above are discussed and tested considering the sampling zeros are from a conversion of a continuous-time feedback system to a discrete-time feedback system using zero-order hold on the input. In the real world, RC will often be needed on digital control systems. The conversion to a discrete-time model maybe applied to a bock of blocks inside the digital feedback control loop. A digital feedback control system may have multiple components including the discrete controller, a continuous-time plant, a possible anti-aliasing filter, and the possible sensor noise filter, etc. The sampling zero locations are affected by the system components and its structure. It is the purpose of this chapter to discuss the influence of discrete-time system structure and its components on the location of sampling zeros, and to discuss the effectiveness of RC design for such cases.

2.1 Sampling Zero Location

There are two types of zeros when one converts a continuous-time transfer function $G(s)$ fed by zero order hold to the corresponding discrete-time transfer function $G(z)$. The zeros of $G(z)$ in the z-plane which are the mappings of zeros of $G(s)$ in the s-plane are called intrinsic

zeros. The extra zeros of $G(z)$ which are introduced during this conversion are called sampling zeros. In this section, we focus on sampling zeros and do not deal with intrinsic zeros. Åström, Hagander, and Strenby discuss the asymptotic location of the sampling zeros as the sampling time interval goes to zero [17]. Consider a discrete-time system created from a strictly proper linear continuous-time system with *n* poles and *m* zeros, fed by a zero-order hold with a synchronized sampler on the output. As the sampling time interval goes to 0, the *m* intrinsic zeros of $G(z)$ converge to 1, and the remaining $n - m - 1$ sampling zeros of $G(z)$ will have the asymptotic locations indicated in Table 2-1.

Table 2-1: Asymptotic Locations of Sampling Zeros as Sampling Period Goes to Zero

No. of Sampling zeros	Sampling Zero Locations
1	-1
2	-3.732, -0.268
3	-9.899, -1, -0.101
4	-23.2, -2.32, -0.432, -0.0431

They also conclude that all continuous-time systems with pole-zero excess larger than 2 will have sampling zeros outside the unit circle, provided that the sampling period is sufficiently small. Such sampling zeros outside the unit circle are called unstable sampling zeros. In fact, unstable sampling zeros occur for quite reasonable sample time intervals. For example, if $G(s)$ having 3 poles at -1 and no zeros, as long as the sampling period is smaller than 1.8399 seconds, it will have one sampling zero outside the unit circle [17].

2.2 Closed Loop Location of Sampling Zeros Influenced by Different Discrete Control System Configurations

Consider the case that the control system is a continuous-time system $G(s)$ fed by a zero-order hold, and the corresponding discrete-time model $G_1(z)$ is an exact conversion of $G(s)$. From Åström, Hagander, and Strenby's discussion, we know how many sampling zeros there will be and their asymptotic locations. However, in the real world, system $G(z)$ could be a

discrete control system, and it could be composed of a discrete controller $C(z)$, the plant $P(s)$, and a transfer function $H(s)$ or $H(z)$ in the feedback loop with various block structures. Since $G(z)$ has continuous components, it will have sampling zeros too. But the sampling zero asymptotic locations are different between $G_1(z)$ and $G(z)$ even if they both have the same plant $P(s)$. It is the purpose of this section to discuss the difference of asymptotic location of sampling zeros between $G_1(z)$ and $G(z)$. In our comparison, we assume the continuous-time control system $G(s)$ fed by a zero-order hold and the discrete-time control system $G(z)$ have one of the following structures.

System 1. Continuous-time control system $G(s)$ composed of the controller $C(s)$ and the plant $P(s)$ with unity feedback, as shown in Figure 2-1.

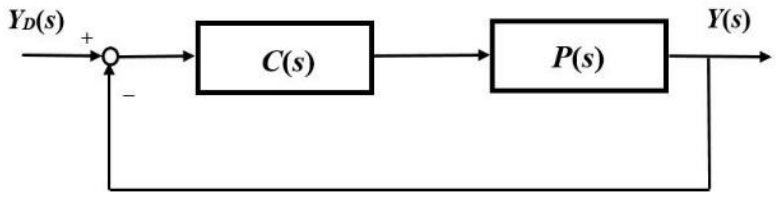

Figure 2- 1: Block diagram of $G(s)$ with unity feedback

System 2. Discrete-time control system $G(z)$ composed of the discrete controller $C(z)$ and the plant $P(s)$ with unity feedback, as shown in Figure 2-2.

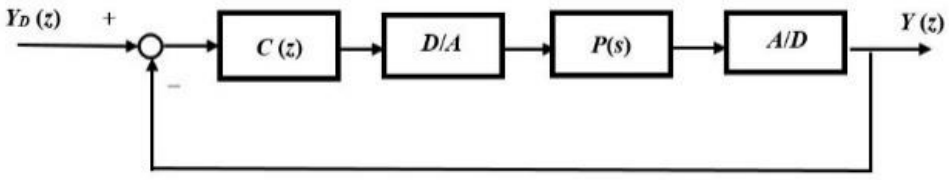

Figure 2- 2: Block diagram of $G(z)$ with unity feedback

System 3. Continuous-time control system $G(s)$ composed of controller $C(s)$ and the plant $P(s)$ with $H(s)$ in the feedback loop, as shown in Figure 2-3.

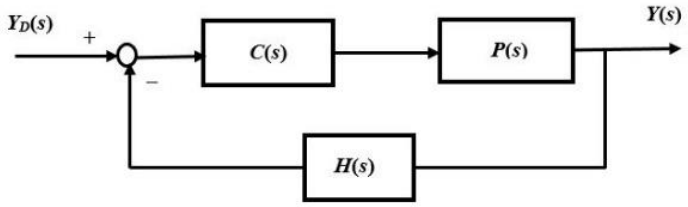

Figure 2- 3: Block diagram of $G(s)$ with $H(s)$ in the feedback loop

System 4. Discrete-time control system $G(z)$ composed of the discrete controller $C(z)$ and plant $P(s)$ with $H(s)$ in the feedback loop, as shown in Figure 2-4.

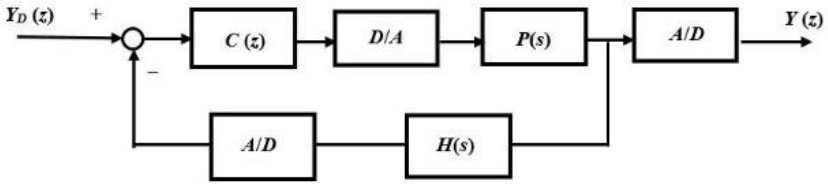

Figure 2- 4: Block diagram of $G(z)$ with $H(s)$ in the feedback loop

System 5. Continuous-time control system $G(s)$ composed of the low-pass filter $L(s)$, the controller $C(s)$ and the plant $P(s)$ with unity feedback as shown in Figure 2-5.

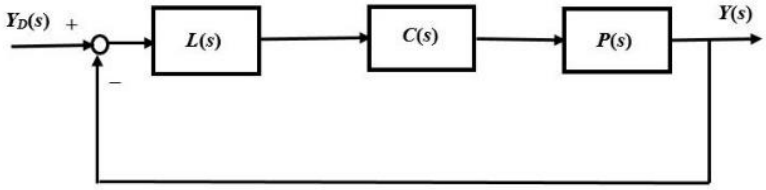

Figure 2- 5: Block diagram of $G(s)$ with the low-pass filter $L(s)$ and the unit feedback

System 6. Discrete-time control system $G(z)$ composed of the anti-aliasing filter $L(z)$, the discrete controller $C(z)$ and the plant $P(s)$ with unity feedback as shown in Figure 2-6.

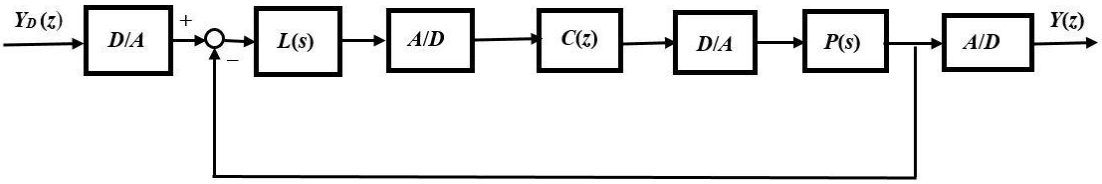

Figure 2- 6: Block diagram of $G(z)$ with an anti-aliasing filter $L(z)$ and unity feedback

System 1 and System 2 are a pair to compare the difference of the asymptotic location of the sampling zero between a continuous-time control system $G(s)$ and a discrete-control system $G(z)$ with unity feedback. Systems 3 and 4 are a pair to compare the difference of the asymptotic location of the sampling zeros between two systems with non-unity feedback. Systems 5 and 6 are a pair to compare the difference of the asymptotic location of the sampling zero between two system with unity feedback and a filter structure.

There are three observations about the relationship between zeros of a single block and the zeros of the system transfer function.

Observation 1. Zeros of a block on the feed-forward path are still zeros of the closed-loop system. When one closes the loop, the transfer function of the block on the feed forward path will be on the numerator of the closed loop transfer function. For example, in System 1, the transfer function on the feed forward path is $C(s)P(s)$, and the closed loop transfer function is $C(s)P(s)/(1 + C(s)P(s))$. Thus, the zeros of $C(s)P(s)$ will also be zeros of the closed loop transfer function. The same applies to discrete time systems like System 2, 4, and 6, but one needs to perform conversion first and then do the block manipulation.

Observation 2. Poles of a block on the feedback path become zeros of the closed-loop transfer function. For example, in System 3, one has a transfer function $H(s)$ in the feedback loop, which could be rate feedback or a low pass filter. The system transfer function $G(s)$ of

System 3 is $C(s)P(s)/(1 + C(s)P(s)H(s))$. The denominator of $1 + C(s)P(s)H(s)$, which is same as the denominator of $C(s)P(s)H(s)$, will be flipped to the numerator of $G(s)$ to cancel the poles of $C(s)P(s)$, leaving the poles of $H(s)$ as the numerator of $G(s)$. Thus, the poles of *H(s)* become the zeros of $G(s)$ and the zeros of $H(s)$ becomes part of the denominator of $G(s)$. The same applies to System 4, if one does the conversion and block manipulation.

Observation 3. The pole-zero excess of the closed-loop transfer function $G(s)$ is equal to the pole-zero excess of the transfer functions on the forward path. This rule applies to continuous control Systems 1, 3, and 5. For example, in System 1, according to Observation 1, the zeros of $G(s)$ are zeros of $C(s)P(s)$, thus the order of the numerator of $G(s)$ is equal to the order of numerator of $C(s)P(s)$. On the other hand, the order of the denominator of $G(s)$, which is equal to the order of the numerator of $1 + C(s)P(s)$, is the same as the order of the denominator of $C(s)P(s)$ if $C(s)P(s)$ has more poles than zeros. A system with more poles than zeros is said to be a strict proper system, and it is said to be proper if the system has the number of poles greater than or equal to the number of zeros. For a typical control system, it is strict proper due to causality. Therefore, the order of the numerator of $G(s)$ equals the order of the numerator of $C(s)P(s)$, and the order of the denominator of $G(s)$ equals the order of the denominator of $C(s)P(s)$. Thus, the pole-zero excess of $G(s)$ is equal to the pole-zero excess of $C(s)P(s)$.

For System 3, the order of the numerator of $G(s)$ is equal to the order of the numerator of $C(s)P(s)$ plus the order of the denominator of $H(s)$. On the other hand, the denominator of $G(s)$ is the numerator of $1 + C(s)P(s)H(s)$, and the order of the numerator $1 + C(s)P(s)H(s)$ is the same as the order of the denominator of $C(s)P(s)H(s)$ if the control system is proper. For the typical control system, due to causality, it is a proper system. Therefore, the order of the numerator of $G(s)$ equals the order of the numerator of $C(s)P(s)$ plus the order of denominator

of $H(s)$, and the order of denominator of $G(s)$ is equal to the order of the denominator of $C(s)P(s)H(s)$. Since both numerator and denominator order of $G(s)$ has the components of the order of denominator of $H(s)$ and it cancels, the pole-zero excess of $G(s)$ is still equal to the pole-zero excess of transfer function $C(s)P(s)$. System 5 is a special case of System 1, thus this rule still applies.

Based on the three observations above, there are three conclusions about the difference of the asymptotic locations of sampling zeros for each pair of continuous-time system $G(s)$ and discrete-time system $G(z)$ with the same plant.

Conclusion 1. For Systems 1, 3, and 5, sampling zeros come from the discretization of the system transfer function $G(s)$; however, for System 2 and 4, sampling zeros come from the discretization of the plant transfer function $P(s)$ alone; for System 6, sampling zeros come from the discretization of plant $P(s)$ and the anti-aliasing filter $L(s)$ respectively. For example, in System 1, sampling zeros come from the discretization of feedback control system transfer function $G(s)$ which is $C(s)P(s)/(1 + C(s)P(s))$, but in System 2, the sampling zeros comes from the discretization of plant $P(s)$ alone. Assuming both system 1 and system 2 have the same plant transfer function, if the pole-zero excesses of the system transfer function $G(s)$ and the plant transfer function $P(s)$ are different, than the asymptotic location of sampling zeros are different for system 1 and system 2; if the pole-zero excesses of $G(s)$ and $P(s)$ are the same, the asymptotic location of sampling zeros are different for system 1 and system 2.

Conclusion 2. For each pair of systems, the difference of the number of sampling zeros between the two depends on the controller transfer function $C(s)$. For example, in System 1 and System 2, Conclusion 1 concludes the sampling zeros of System 1 come from conversion of $G(s)$ and the sampling zeros of System 2 come from conversion of $P(s)$. Observation 3

concludes that pole-zero excess of $G(s)$, which is equal to $C(s)P(s)/(1+C(s)P(s))$, is the same as the pole-zero excess of $C(s)P(s)$. Thus, the number of sampling zeros in System 1 depends on the pole-zero excess of $C(s)P(s)$, and the number of sampling zeros in System 2 depends on the pole-zero excess of $P(s)$. There are three cases to determine the possible number of sampling zeros between $P(s)$ and $C(s)P(s)$.

A. If $C(s)$ is an integral controller or PI controller, $C(s)P(s)$ will have one more pole-zero excess than $P(s)$, thus System 1 will have one more sampling zero than System 2.
B. If $C(s)$ is a PD controller or PID controller, $C(s)P(s)$ will have one less pole-zero excess than $P(s)$, thus System 1 will have one less sampling zero than System 2.
C. If $C(s)$ is P controller, Lead controller, Lag controller, or Lead-Lag controller, since the pole zero excess of $C(s)$ is zero, $C(s)P(s)$ and $P(s)$ have the same pole-zero excess, thus both systems have the same number of sampling zeros.

The asymptotic sampling zero locations only depends on the pole-zero excess of the continuous-time system. Therefore, for Case A and Case B, System 1 and System 2 will have different asymptotic sampling zero locations, and for Case C, System 1 and System 2 will have the same asymptotic sampling zero locations. The same rule applies to System 3 and System 4. The sampling zeros of System 4 come from sampling of $P(s)$ alone, but the sampling zeros of System 3 come from discretization of $C(s)P(s)/(1+C(s)P(s)H(s))$. Recall Observation 3, pole-zero excess of System 3 is still the same as the pole-zero excess of $C(s)P(s)$, thus the result discussed in the previous paragraph again applies.

Conclusion 3. With a prefilter, e.g. an anti-aliasing filter, in the discrete-time system like System 6, sampling zeros are the union of sampling zeros from the prefilter and the plant respectively, thus, the asymptotic sampling zero location is different from the result in Reference

17 which is based on the pole-zero excess of the transfer function alone. Suppose in System 6, $L(s)$ has pole-zero excess of 3, and $P(s)$ has pole zeros excess of 2, then one expects the asymptotic location of sampling zeros is at -3.732, -0.268, and -1, which is different from a typical system with pole-zero excess of four where the asymptotic sampling zero locations are at -9.899, -1, and -0.101 suggested by Reference 17. For a special case, when $L(s)$ and $P(s)$ are identical, one can expect repeated sampling zeros patterns; if $L(s)$ and $P(s)$ have the same pole-zero excess but are not identical, for a reasonable sampling rate, one might have adjacent pairs of sampling zeros; if $L(s)$ and $P(s)$ have different pole-zero excesses as we assumed previously, the asymptotic location of sampling zeros in system 6 are a combination of the asymptotic location of $L(s)$ and $P(s)$ respectively.

With established the possible changes in the pattern of sampling zeros that can occur with different discrete control configurations, we now address the question of how these differences influence the design of repetitive controllers.

2.3 FIR Compensator Design Using Optimization of Learning Rate in the Frequency Domain

The repetitive controller will have a general form as suggested by Equation (1-16). To design a repetitive controller is equivalent to design $F(z)$. Looking back at Equation (1-18), if one can make $G(z)F(z)$ equal to one, then one can have $E(z) = 0$, suggesting one has zero tracking error after one period of command. An ideal repetitive controller uses $F(z) = G^{-1}(z)$ and has zero tracking-error after one period of the command. However, as discussed above this approach usually fails, $G(z)$ will have sampling zeros outside the unit circle, and its inverse is not stable. Thus, $R(z)$ in Equation (1-16) is not stable leading to an unstable command to the feedback system. This is the reason why the sampling zero location of the feedback system is a

key factor in our RC design. The previous literature discusses several design approaches to solve this issue.

Panomruttanarug and Longman use a quadratic cost function to optimize the FIR compensator $F(z)$, making it close to the inverse of the frequency response of the system transfer function $G(z)$ [14]. Since the FIR compensator is only approximately the inverse of the frequency response of $G(z)$, it bypasses the instability issue of the compensator $F(z)$ chosen as the inverse of $G(z)$.

The cost function has the form

$$J = \sum_{j=0}^{179} [1 - G(e^{i\omega_j T})F(e^{i\omega_j T})]W_j[1 - G(e^{i\omega_j T})F(e^{i\omega_j T})]^* \qquad (2\text{-}1)$$

where W_j is a weighing factor, ω_j is a chosen set of frequencies from zero up to Nyquist frequency, and the asterisk represents the complex conjugate. In the above equation, the number of frequencies has been chosen as 180 corresponding to every one degree going around the unit circle, which is usually sufficient. One only needs the frequency response of the system to minimize this cost function, and one does not necessarily need the system model to do the computation. Therefore, experimental data of the magnitude response $M(\omega)$ and the phase response $\varphi(\omega)$ of the system can be used to find the corresponding compensator. This is one important benefit of using this design approach.

The minimum of *J* is achieved when one differentiates with respect to the filter coefficients a_i and sets it to zero. It produces the following *n* linear equations to solve for the coefficients a_i

$$Ax = b \qquad (2\text{-}2)$$

A

$$= \sum_{j=0}^{179} M^2(\omega_j) \begin{bmatrix} 1 & cos\,(\omega_j T) & \cdots & cos\,((n-1)\omega_j T) \\ cos\,(\omega_j T) & 1 & \cdots & cos\,((n-2)\omega_j T) \\ \vdots & \vdots & \ddots & \vdots \\ cos\,((n-1)\omega_j T) & cos\,((n-2)\omega_j T) & \cdots & 1 \end{bmatrix} \quad (2\text{-}3)$$

$$x = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \end{bmatrix} \quad (2\text{-}4)$$

$$b = \sum_{j=0}^{179} M(\omega_j) \begin{bmatrix} cos\,(\varphi(\omega_j) + (m-1)\omega_j T) \\ cos\,(\varphi(\omega_j) + (m-2)\omega_j T) \\ \vdots \\ cos\,(\varphi(\omega_j) + (m-n)\omega_j T) \end{bmatrix} \quad (2\text{-}5)$$

2.4 FIR Compensator Design Based on Individual Taylor Series Expansion Approximations of Reciprocal of Each Transfer Function Zero

Xu and Longman present another approach of designing compensator *F*(*z*). This approach handles the reciprocal of sampling zeros outside the unit circle by using a Taylor expansion of the reciprocal of such sampling zeros, term by term [15]. This produces a power series and overcomes the instability issue. For example, consider a third order system $G(z)$ having one sampling zero z_I inside the unit circle and one sampling zero z_O outside the unit circle and all the poles p_i inside the unit circle

$$G(z) = \frac{K(z - z_I)(z - z_o)}{(z - p_1)(z - p_2)(z - p_3)} \quad (2\text{-}6)$$

The inverse of this system which is unstable can be written as

$$G^{-1}(z) = \left[\frac{(z - p_1)(z - p_2)(z - p_3)}{K}\right]\left[\frac{1}{z - z_I}\right]\left[\frac{1}{z - z_o}\right] \quad (2\text{-}7)$$

The corresponding FIR filter $F(z)$ uses the terms in the first square bracket and uses the Taylor expansion of the next two terms. Recall the Taylor expansion of the following equation

$$\frac{1}{1+\hat{z}} = 1 - \hat{z} + \hat{z}^2 - \hat{z}^3 + \cdots = \sum_{k=0}^{\infty} (-\hat{z})^k \qquad (2\text{-}8)$$

This series has a convergence range of $|\hat{z}| < 1$. The $1/(z - z_I)$ term and $1/(z - z_o)$ can be manipulated to have the same form as the above expansion

$$\frac{1}{z - z_I} = \frac{1}{z}\left(\frac{1}{1 - z_I/z}\right) = \frac{1}{z}\left(1 + \frac{z_I}{z} + \left(\frac{z_I}{z}\right)^2 + \left(\frac{z_I}{z}\right)^3 + \cdots\right) = \frac{1}{z}\sum_{k=0}^{\infty}\left(\frac{z_I}{z}\right)^k \qquad (2\text{-}9)$$

$$\frac{1}{z - z_o} = \frac{1}{z_o}\left(\frac{1}{1 - \frac{z}{z_o}}\right) = \frac{1}{z_o}\left(1 + \frac{z}{z_o} + \left(\frac{z}{z_o}\right)^2 + \cdots\right) = \frac{1}{z_o}\sum_{k=0}^{\infty}\left(\frac{z}{z_o}\right)^k \qquad (2\text{-}10)$$

Equation (2-9) has a convergence range of $|z_I/z| < 1$. Since $|z_I| < 1$ and the z value needed in the filter is on the unit circle, the Taylor expansion of Equation (2-9) will be convergent. The same applies to Equation (2-10). Therefore, the unstable poles of the inverse system transfer function are converted to a series of zeros. When one designs $F(z)$, one only picks the number of terms needed in the Taylor expansions in Equations (2-9) and (2-10), and the FIR filter $F(z)$ will have the form of

$$F(z) = \left[\frac{(z - p_1)(z - p_2)(z - p_3)}{K}\right]\left[\frac{1}{z}\sum_{j=0}^{n}\left(\frac{z_I}{z}\right)^j\right]\left[\frac{1}{z_o}\sum_{k=0}^{m}\left(\frac{z}{z_o}\right)^k\right] \qquad (2\text{-}11)$$

2.5 FIR Compensator Designs Based on an Improved Taylor Series Expansion of the Reciprocal of All Zeros Simultaneously

Prasitmeeboon and Longman created a modification of the Taylor expansion method above [16]. The method above from Xu and Longman make Taylor series expansions of each factor separately. The true Taylor expansion of the zeros terms expands the product and consider some cross multiplications terms in the Taylor expansion. This method uses the mathematically correct Taylor expansion for the multiplication terms. The results presented below indicate that

for many systems the expectation is correct, but for some systems the previous Taylor expansion works better. For an example of the difference between the two expansions, expand the following expression where both a_1 and a_2 are larger than 1

$$G^{-1}(z) = \frac{1}{(z+a_1)(z+a_2)} \tag{2-12}$$

The method of Reference 15 first expands the individual terms to a chosen number of terms and then multiplies them together, which in the case of using only two terms produces FIR filter $F_1(z)$ as

$$F_1(z) = \frac{1}{a_1 a_2}\left(1-\frac{z}{a_1}\right)\left(1-\frac{z}{a_2}\right) = \frac{1}{a_1 a_2}(1-\left(\frac{1}{a_1}+\frac{1}{a_2}\right)z+\frac{z^2}{a_1 a_2}) \tag{2-13}$$

The Taylor expansion of the product here suggests that mathematically one needs to consider the z^2 term in both brackets because it can multiply with the constant term in the other bracket. Therefore, the FIR filter $F(z)$, whose highest power is 2, should be

$$\begin{aligned} F_2(z) &= \frac{1}{a_1 a_2}\left(1-\frac{z}{a_1}+\frac{z^2}{a_1^2}+\cdots\right)\left(1-\frac{z}{a_2}+\frac{z^2}{a_2^2}+\cdots\right) \\ &= \frac{1}{a_1 a_2}\left(1-\left(\frac{1}{a_1}+\frac{1}{a_2}\right)z+\left(\frac{1}{a_1 a_2}+\frac{1}{a_1^2}+\frac{1}{a_2^2}\right)z^2+\cdots\right) \end{aligned} \tag{2-14}$$

The two Taylor expansion approaches help one interpret the circle-like zero patterns of the FIR filter zeros designed by the Learning Rate Optimization method. For the Optimization method in section 2.3, one needs to pick the value of n and m for the compensator. The value $n-1$ decides how many zeros will be in your FIR compensator $F(z)$, and the value $n-m$ decides the number of poles at the origin in the FIR compensator $F(z)$. Generally speaking, a larger value of n means a better FIR filter performance: the learning rate is faster which one has the same level of tracking errors in fewer periods. This result is quite natural since the more terms in the FIR filter, the more its frequency response resembles the frequency response of

$G^{-1}(z)$. Given a value of n, one must select the value of m so that the number of poles of $G(z)F(z)$ inside the unit circle equals to the number of zeros of $G(z)F(z)$ insides the unit circle. Otherwise, stability condition Equation (1-19) is violated. Adjusting the value of m adjusts the number of poles at the origin addressing this need. The criteria as Equation (2-1) will also put zeros more or less on top of the system poles inside the unit circle, once the *m* and *n* values are picked based on the rule to equalize the number of poles and zeros of $G(z)F(z)$ inside the unit circle.

For the Taylor Expansion Method in section 2.4, one can actually decide the number of terms one needs to compensate each zero inside or outside the unit circle without worrying about balancing the number of poles and the number of zeros within the unit circle, the math of the Taylor Expansion for each term of $G^{-1}(z)$ automatically satisfies this requirement. However, for the Improved Taylor Expansion Method in section 2.5, the pick is subtle because one has a Taylor expansion of $G^{-1}(z)$ aligned in the increasing order of z, one can use the FIR filter designed by Learning Rate Optimization or Taylor Expansion Method to assist in determining in picking which section of expansion designed by Improved Taylor Expansion Method to use.

2.6 Assumptions for Comparing Performance of FIR Design Approaches for Different Control System Structures

We now compare the performance of different FIR filter design approaches for the possible sampling zero locations. In order to simplify the discussion, we make the following three assumptions. $G(z)$ is the discrete-time system transfer function of system 1 to 6, assuming system 1, 3, 5 with zero-order hold.

Assumption 1. All the poles of $G(z)$ inside the unit circle are canceled by putting zeros on top of them in $F(z)$.

Assumption 2. There are no intrinsic zeros of $G(z)$. On one hand, this assumption simplifies our discussion so that we can solely focus on the compensator performance for the sampling zero patterns without the interference from the intrinsic zeros; on the other hand, even if the system transfer function has intrinsic zeros, as it suggests, all these zeros will approach to $+1$ as the sampling time interval decreases. Thus, the compensator zero patterns for those intrinsic zeros should be more or less like the compensator zero patterns for sampling zeros inside the unit circle.

Assumption 3. There are no zeros from the discrete controller in system 2, 4, and 6. This assumption limits out discussion to the proportional controller for discrete control system. However, since the zeros from the discrete controller will all be inside the unit circle, the compensator zero patterns for those zeros from the discrete controller would be more or less like the compensator zero patterns for sampling zeros inside the unit circle. Thus, this assumption won't affect our result, but will significantly simplify our discussion.

Assumption 4. Each sampling zero is compensated by the same number of zeros of FIR compensator $F(z)$. On one hand, this assumption helps compare three methods on the same baseline. On the other hand, the allocation of FIR filter zeros inside or outside the unit circle depends on the location of the sampling zeros as well. For example, for fixed number $n = 15$, for sample zeros at -3.54 and -0.8, the best result comes when $m = 2$; but, for sampling zeros at -3.54 and -0.5, the best result comes when $m = 6$. Therefore, we pick the value of m so that each sampling zero is compensated by the same number of FIR filter zeros, and this helps us compare the three methods under the same standard.

2.7 FIR Filter Zero Patterns from the Exact Discretization of the Continuous Time Transfer Function

In our numerical study, we will compare how three methods will compensate for the sampling zeros only based on our assumption. Several examples show performance of the previous three repetitive controller designs for continuous-time systems fed by zero-order hold. Those examples use $G_2(s)$ and $G_1(s)G_2(s)$ as the closed loop system transfer function $G(s)$. Little previous research investigated the FIR filter design if the system has pole-zero excess larger than 3, with no thorough and detailed comments were made about the FIR filter's zero pattern. This section will examine these situations. The sampling zeros are from the exact conversion of system transfer function $G(s)$, and $G(s)$ is some combination of $G_1(s)$ and $G_2(s)$ where $a = 8.8, \omega_0 = 37, \zeta_0 = 0.5$. For a single sampling zero, the exact conversion of $G_2(s)$ is considered; for two sampling zeros, the exact conversion of $G_1(s)G_2(s)$; for three sampling zeros, the exact conversion of $G_2(s)G_2(s)$; and for four sampling zeros, the exact conversion of $G_1(s)G_2(s)G_2(s)$. Unless otherwise stated, the sample rate is 100 Hz. In the numerical study, we assume that our $G(s)$ will be the combination of $G_1(s), G_2(s)$

$$G_1(s) = \frac{a}{s+a} \qquad G_2(s) = \frac{\omega_0^2}{s^2 + 2\zeta_0\omega_0 + \omega_0^2} \tag{2-15}$$

For System 1, System 3, and System 5, based on the assumptions and Reference 17, its asymptotic sampling zero locations are determined by its pole-zero excess of $G(s)$ only. For the ease of format, the figures in the following sections are put in the end of this chapter. Assume that Systems 1, 3, 5, will have the $G(s)$ as below, then,

1. When the pole-zero excess of $G(s)$ is two, meaning only one sampling zero is introduced, there is no difference between the Taylor expansion approximation method and

Improved Taylor expansion method, as shown in Figure 2-7. In Figure 2-7, the sampling zero at -0.8836, and the resulting FIR filter of the above two methods have the zeros coinciding with each other; the two methods are different only when there are two or more sampling zeros, as shown in Figure 2-8 and Figure 2-16, with two sampling zeros coming from $G_1(s)G_2(s)$ and three sampling zeros coming from $G_2(s)G_2(s)$ fed by zero order hold.

2. FIR filter by the Taylor Expansion approximation method has a pattern with zeros evenly distributed exactly on circles centered at the origin with radii equal to the distance between the sampling zero and the origin as shown in Figure 2-7, 2-8, and 2-9; the number of circles is equal to the number of sampling zeros, as shown in Figure 2-8.

3. FIR filter designed by Improved Taylor expansion approximation method has a zero pattern of a distorted circle if it has more than one sampling zero, as shown in Figure 2-8 and Figure 2-9. Moreover, if there is more than one sampling zero, the number of distorted circles is always two: one is inside the unit circle and the other is outside the unit circle as shown in Figure 2-8, Figure 2-10, and Figure 2-17. This is a new pattern not observed before.

4. FIR filter designed by Learning Rate Optimization has a zero pattern of distorted circle and the number of circles is always two if there are more than two sampling zeros: one circle is inside the unit circle, and the other is outside the unit circle, as shown in Figure 2-7, 2-8, 2-10, 2-13, and 2-16. Figures 2-10 and 2-13 both show the FIR filter for the same transfer function $G_2(s)G_2(s)$ while the first one is sampled at 100 Hz and the second at 20Hz. The two have different zero patterns.

5.The three methods all generate FIR filters whose zeros form circle-like patterns. The radius of these circle-like patterns expands and shrinks as the sampling zeros move close to or move away from the origins as shown in Figures 2-7, 2-20, and 2-21.

6. For a fixed number of terms in the FIR filter, the learning rate increases as the sampling zero is further from the unit circle; the learning rate decreases as the sampling zero is closer to the unit circle; the sampling zeros which are too close to the unit circle might result in unstable systems as shown in Figures 2-27 and 2-28.

7. For fixed locations of the sampling zeros, the learning rate increases as one includes more terms in FIR filter; the learning rate decreases as one reduces the number of terms in the FIR filter, as shown in Figure 2-23, 2-24, 2-25, and 2-26.

8.The Learning Rate Optimization method emphasizes low frequency performance, Taylor expansion method has no preference for frequency because of its symmetric zero pattern, and the improved Taylor expansion method emphasizes high frequency performance, as shown in Figures 2-12, 2-15, 2-19, 2-25, and 2-26.

For System 2 and System 4, its asymptoticly sampling zero location is determined by the pole-zero excess of $P(s)$ only, if $P(s)$ and $G(s)$ have the same pole-zero excess, then its asymptotic sampling zeros locations are same. Now the discussion above is applied to System 2 and 4.

2.8 Repeated Zeros Pattern Produced by Discretization of Two Identical Systems

In System 6, as stated previously, with a pre-filter in the discrete control system, one might have repeated sampling zeros. As an illustration, consider a discrete control system that has identical transfer functions for both pre-filter and plant, consisting of the same combination of $G_1(s)$ and $G_2(s)$ with 100Hz sampling rate. Observations 5, 6, 7, and 8 in the previous section are valid here. And some new patterns are discovered.

9. For FIR filter by the Taylor Expansion approximation method, the zeros are evenly distributed and repeated at each radius. However, the Improved Taylor method and Learning

Rate Optimization method have FIR filter zeros spread out, as shown in Figure 2-27 and Figure 2-30.

10. When repeated sampling zeros are close to the unit circle, FIR filter by Improved Taylor Expansion approximation method can lead to an unstable system when there are relatively few terms in the FIR filter, because, when sampling zeros are outside, its FIR filter zeros can be inside, violating the rule of equal number of zeros and poles inside the unit circle. For this case, Improved Taylor method is not improving the performance, as shown in Figure 2-28 and Figure 2-31.

11. One solution to the case of repeated sampling zeros close to the unit circle is to increase the number of terms in the FIR filter as shown in Figure 2-29. Figure 2-29 shows 33 terms are needed for the FIR filter designed by Improved Taylor method to have a better learning rate than a 13-term FIR filter designed by Taylor Method.

2.9 Adjacent Zero Pattern Produced by Discretization of Two Systems with the Same Pole-Zero Excess

In System 6, when this discrete control system has two transfer function with the same pole-zero excess for the pre-filter and the plant but different transfer functions for each, there will be adjacent sampling zeros after the exact conversions. Consider adjacent zeros inside the unit circle at -0.8836 and -0.8, and adjacent zeros outside the unit circle are at -1.2 and -1.3. Rules 5, 6, 7, 8, and 9 apply here, and there are some new phenomena.

12. For adjacent zeros inside the unit circle, FIR filter by Learning Rate Optimization method and Improved Taylor method both have a zero pattern with zeros on a deformed circle with a radius larger than the distance between the sampling zero and the origin, as shown in Figure 2-32.

13. For adjacent zeros outside the unit circle, FIR filter by Learning Rate Optimization method and Improved Taylor method have a zero pattern whose zeros tend to form a circle with a radius smaller than the distance between sampling zeros outside the unit circle and the origin as in Figure 2-35.

14. Like the repeated sampling zeros case, when sampling zeros are close to the unit circle, FIR filter designed by Improved Taylor Method does not improved performance compared to the one by Taylor Method as shown in Figures 2-33 and 2-36. Figure 2-34 shows that when the number of terms reaches 28, Improved Taylor method has better performance than an 11-term FIR filter by Taylor Expansion method for adjacent zeros inside the circle; Figure 2-37 shows that Improved Taylor method needs an 18-term filter to have better performance than an 11-term filter by Taylor expansion method for adjacent zeros outside the unit circle.

2.10 Sampling Zero Patterns Produced by Discretization of Two Systems with Different Pole-Zero Excesses

In System 6, when this discrete control system has two transfer functions with different pole-zero excess for the pre-filter and the plant, there will be a new sampling zero pattern after the exact conversion. As an illustration, we examine the case where the pre-filter has the form of $G_2(s)$ and the plant is $G_1(s)G_2(s)$, and we also examine the case when the pre-filter has the form of $G_2(s)$ and the plant is $G_2(s)G_2(s)$ so that one has four sampling zeros.

15. For the first case, since its sampling zeros are closer to the unit circle, the learning rate is slower than the typical system with three sampling zeros as shown in Figure 2-39.

16. For the second case, since the sampling zeros are closer to the unit circle, the learning rate is slower than the typical system with four sampling zeros, as shown in Figure 2-41. Also

since there are two adjacent zeros near the unit circle, its zero pattern is like what we describe in Observations 12 to 14.

2.11 Conclusion

Previous RC literature emphasized handling of the zeros introduced by discretization of the input/output of a continuous time feedback control system. These zeros have limiting patterns that are only a function of the zero-pole excess. The previous literature presents three methods to design FIR compensators to handle the sampling zeros outside the unit circle that prevent the use of an inverse model. Each introduces extra zeros around distorted circles about the origin. We show that the distortions make the Optimization method favor learning at low frequencies, Taylor expansion method shows no preference, and Improved Taylor Expansion method favors high frequencies. It is also shown that the Improved Taylor method has only an improvement if there are enough terms in the FIR filter. Previous works studied RC design for the limiting patterns for converting a continuous time feedback system, while here we study applications to discrete feedback control systems. This includes a discrete controller, continuous plant that is converted, possible feedback loop filter or rate term, and possible anti-aliasing filter. New possible patterns of zeros that the FIR must compensate are discovered with possibly repeating zeros, union of patterns from the same pole excess but different locations, and union of the patterns from different pole excesses. Each of the three methods still work for these new patterns.

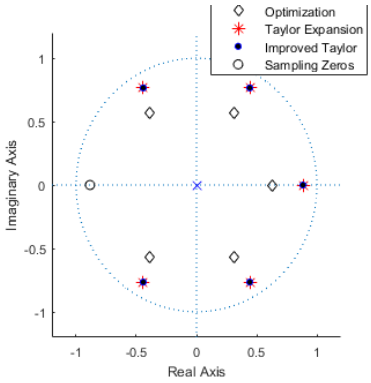

Figure 2-7. 6-term compensator design using three approaches to compensate sampling zero -0.8836 with $n = 6, m = 0$

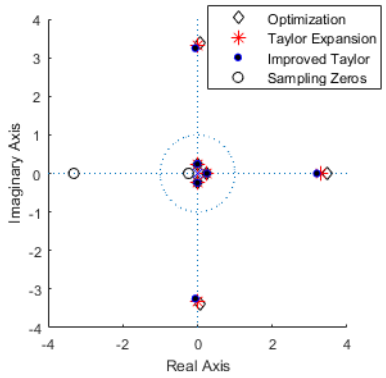

Figure 2-8. 7-term compensator design for two sampling zeros from $G_1(s)G_2(s)$ with $n = 7, m = 3$ (full image)

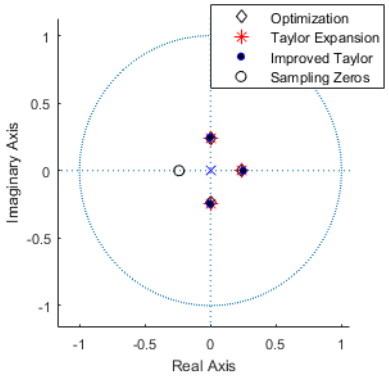

Figure 2- 9. 7-term compensator design for two sampling zeros from $G_1(s)G_2(s)$ (detail)

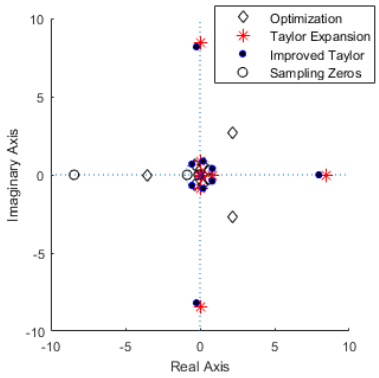

Figure 2- 10. 10-term compensator designs for three sampling zeros (full image) with $n = 10, m = 2$

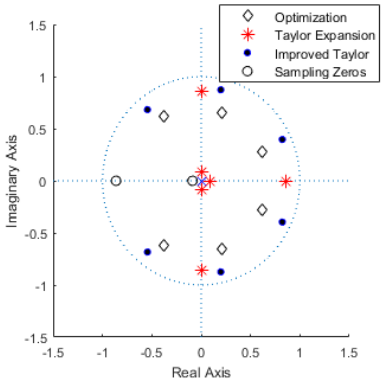

Figure 2-11. 10-term compensator designs for three sampling zeros (detail)

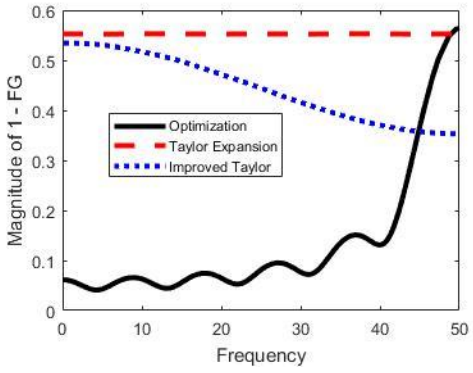

Figure 2-12. Learning rate of compensators in Figure 2-10

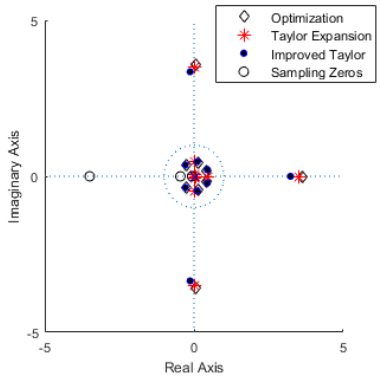

Figure 2-13. 10-term compensator designs for the same system as Figure 2-11, but sampling at 20 Hz (full view) with $n = 10, m = 2$

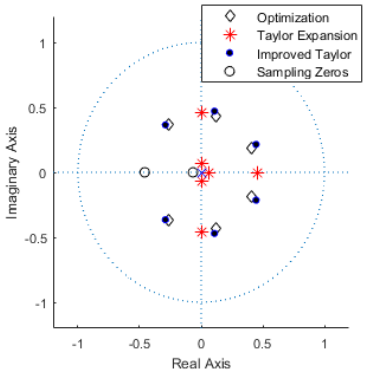

Figure 2-14. 10-term compensator designs for the same system in Figure 2-13 (detail)

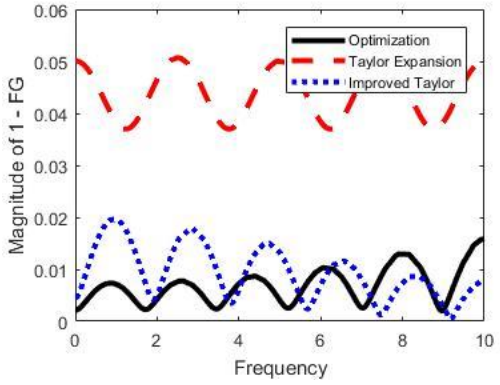

Figure 2-15. Learning rate vs Frequency for system in Figure 2-13

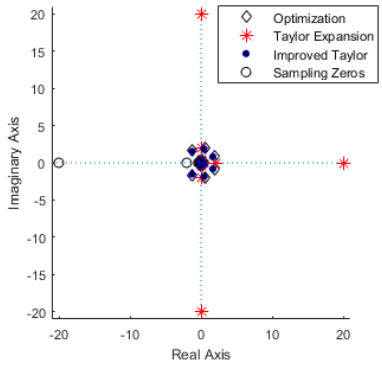

Figure 2-16. 13-term compensator for four sampling zeros (full view) with $n = 13, m = 5$

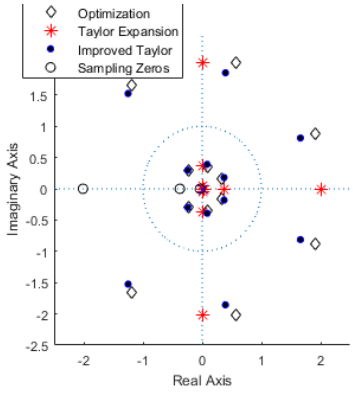

Figure 2-17. 13-term compensator for four sampling zeros (detail)

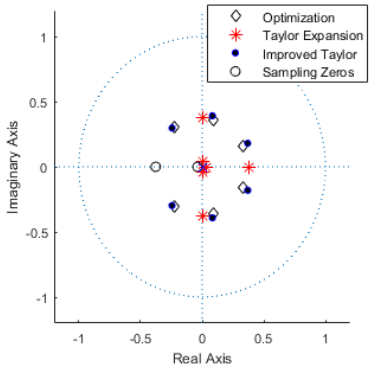

Figure 2-18. 13-term compensator for four sampling zeros (more detail)

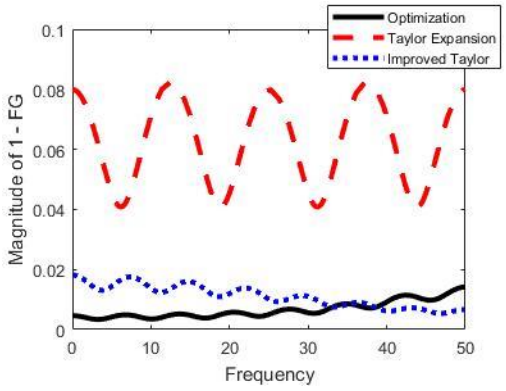

Figure 2-19. Learning rate vs Frequency for system in Figure 2-16

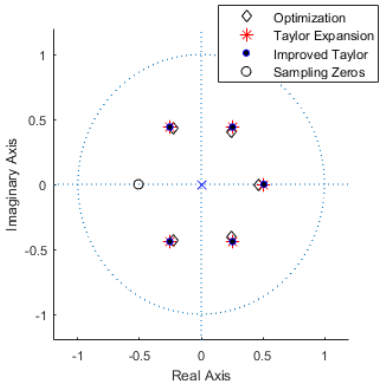

Figure 2-20. Sampling $G_1(s)$ at 20 Hz, sampling zero at -0.53, small radius circle with $n = 6, m = 0$

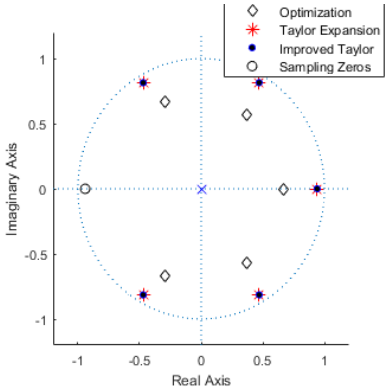

Figure 2-21. Sampling $G_1(s)$ at 200 Hz, sampling zero at -0.9401, small radius circle with $n = 6, m = 0$

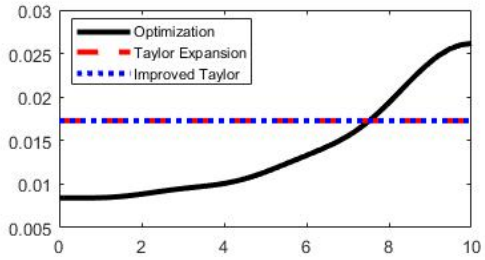

Figure 2-22. Learning rate vs. frequency for 6-term filters in Figure 2-20

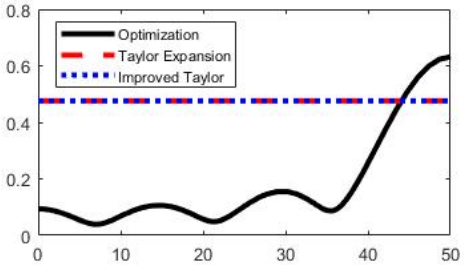

Figure 2-23. Learning rate vs. frequency of 6 term filters in Figure 2-21

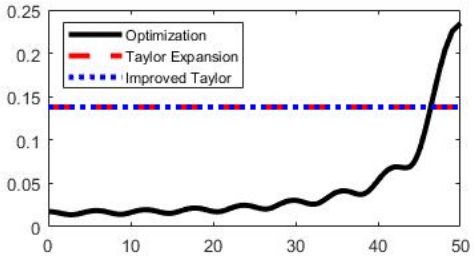

Figure 2-24. Learning rate vs. frequency for sampling zero -0.8836, but using 17 term FIR filter

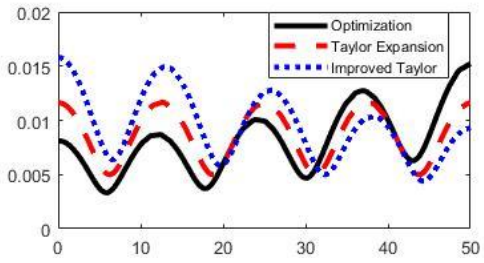

Figure 2-25. Learning rate vs. frequency for the 7-term filter in Figure 2-9

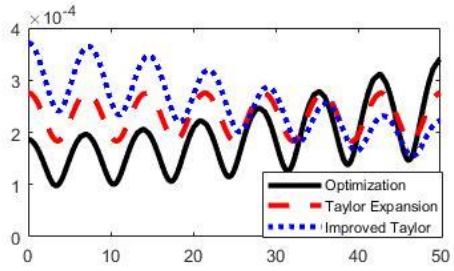

Figure 2-26. Learning rate vs. frequency for sampling zeros at -3.3104 and -0.2402 using 13-term FIR filter

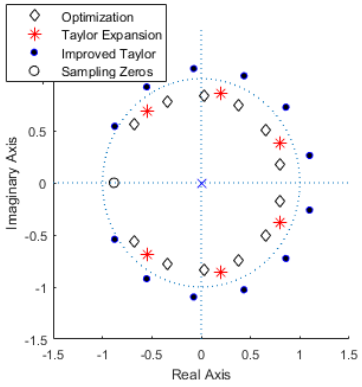

Figure 2-27. Three 13-term FIR filters for repeated zeros at -0.8836 with $n = 13, m = -1$

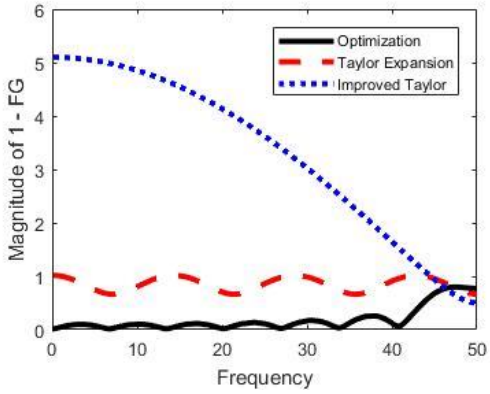

Figure 2-28. Learning rate graph for Figure 2-27

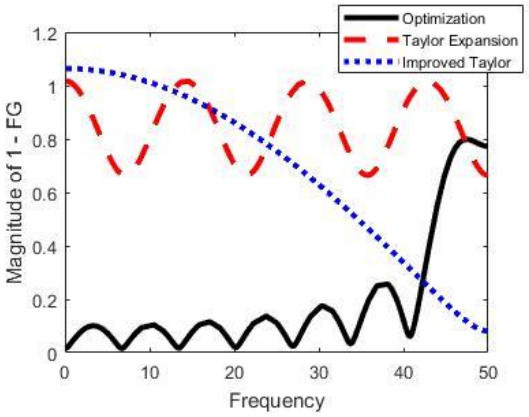

Figure 2-29. 33-term Improved Taylor filter vs. 13-term optimization filter and Taylor filter for repeated zeros at -0.8836

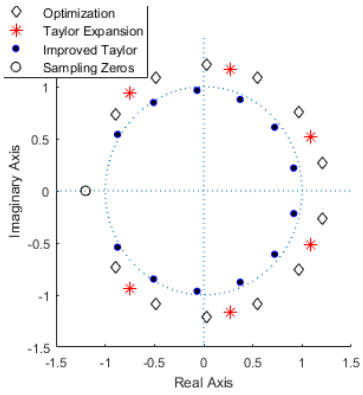

Figure 2-30. Three 13-term FIR filters for repeated zeros at -1.2 with $n = 13, m = 13$

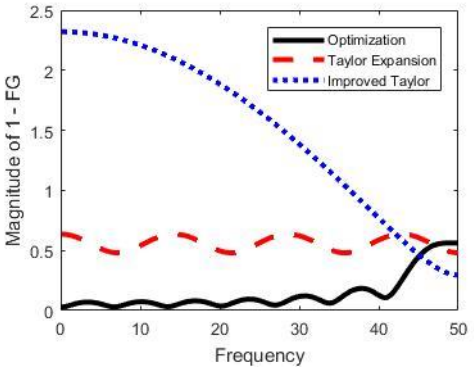

Figure 2-31. Learning rate graph for Figure 2-30

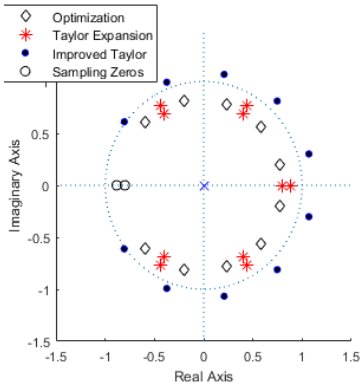

Figure 2-32. 11-term FIR compensator design using different approaches for adjacent zeros at –0.8836 and -0.8 with $n = 11, m = -1$

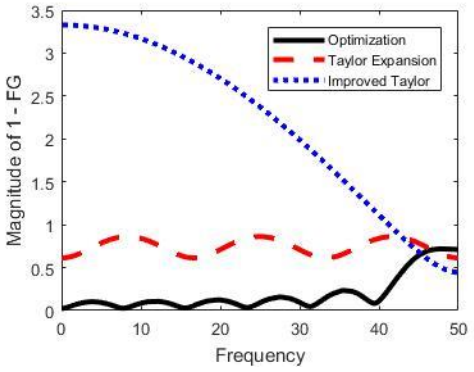

Figure 2-33. Learning rate figure for filters in Figure 2-33

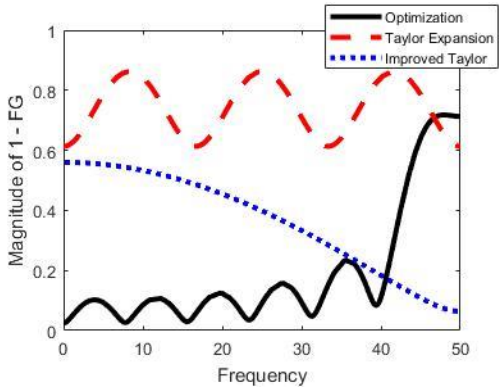

Figure 2-34. Learning rate of a 28-term Improved Taylor filter vs filter Figure 2-32

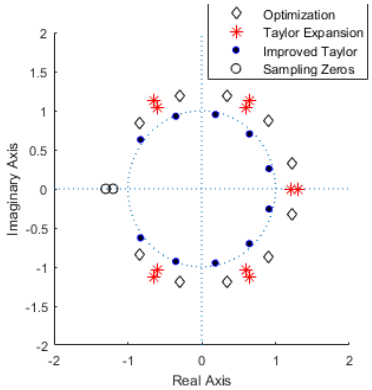

Figure 2-35. 11-term FIR compensator design using different approaches for adjacent zeros at –1.2 and -1.3 with $n = 11, m = 11$

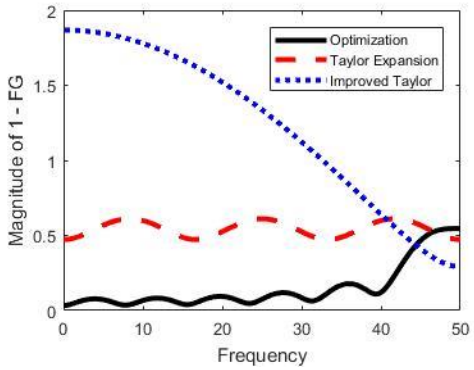

Figure 2-36. Learning rate figure for filters in Figure 2-35

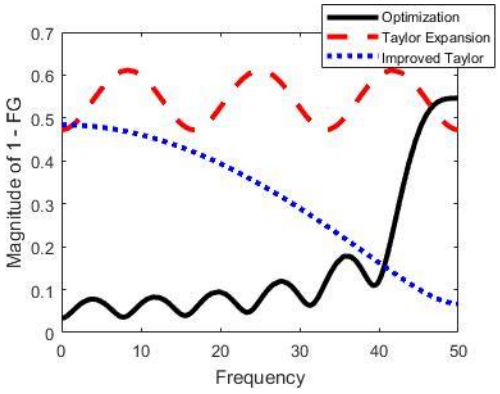

Figure 2-37. 18-term Improved Taylor filter vs 11-term Optimization and Taylor filters

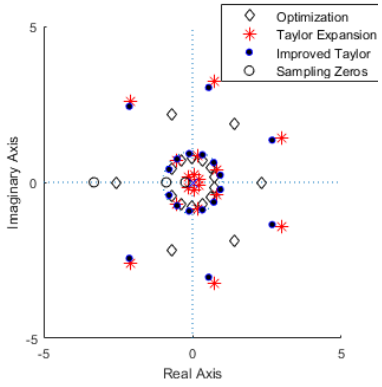

Figure 2-38. 19-gain filter for united sampling zeros pattern with $n = 19, m = 5$

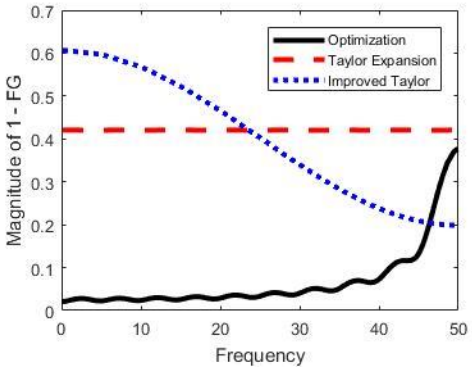

Figure 2-39. Learning rate for Figure 2-30

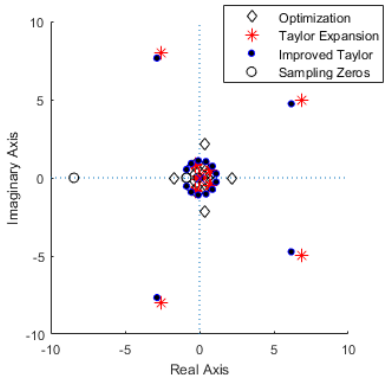

Figure 2-40. 17 gain filter design for union case (full view) with $n = 17, m = 2$

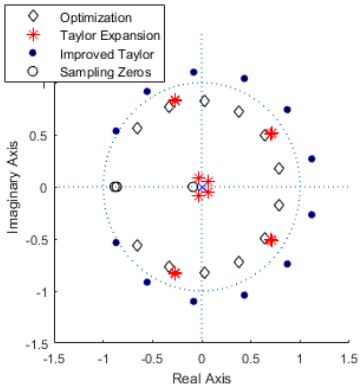

Figure 2-41. 17 gain filter for union case (detail)

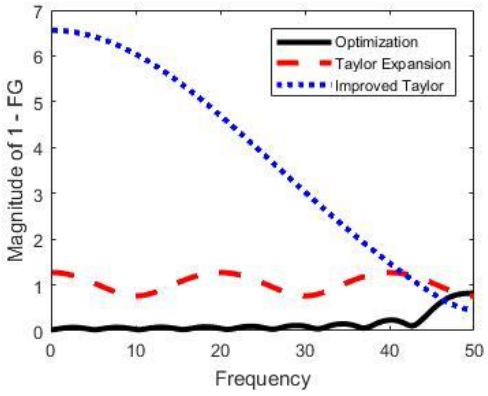

Figure 2-42. Learning rate for Figure 2-40

Chapter 3: Good Performance Above Feedback Control System Bandwidth Using Command Modified by Partial Inverse Model

A common requirement specified to a feedback control system designer is the needed bandwidth. When using typical feedback control laws, such as proportional, integral, proportional plus derivative, etc. it can easily happen that the desired bandwidth requirement is not achievable for any choice of control gains. Several ways of seeing the bandwidth limitation the designer faces are presented. To address this situation, a method is offered to effectively raise the bandwidth of a feedback system modifying its command based on the partial inverse of this feedback system: given a desired tracking maneuver, one first uses the partial inverse of the feedback system to compute the command and then applies it to the feedback system. This method is founded in the effort to combine the learning gain matrix and the low-pass filter in ILC. Bandwidth specification does not ask for high accuracy tracking within the bandwidth, but if it is needed, a method is also presented to use ILC concepts to improve the tracking accuracy.

3.1 Introduction

Any feedback control system has a bandwidth associated with the transfer function from command to response. Frequency components of command below this bandwidth frequency will produce outputs that are reasonably close to the commanded component, but frequency components above this bandwidth frequency have rapid amplitude attenuation as the frequency increases. Unfortunately, the feedback control system designer has serious limitations concerning how high a bandwidth can be produced by typical feedback control laws through optimization of controller gain or gains. The initial objective of ILC is to eliminate tracking error at all frequencies, at the expense of performing iterations to learn the command needed to produce the desired output. The feedback controller then has a command that is not what the user wants the

system to do, but the command has the effect of making the system do what you want it to do. ILC is iterating with the real world, it can also learn to counteract the influence of model error and repeating deterministic disturbances.

ILC can be very effective. Experiments performed on a commercial robot at NASA Langley Research Center decreased the tracking error of the robot following a high-speed trajectory by a factor of 1000 in about 12 iterations for learning [11]. However, there are various issues that appear when trying to use ILC to correct errors at all frequencies up to Nyquist. ILC, in fact, can be viewed as aiming to converge to the inverse transfer function of the system: the command to the system, modified by ILC, is equal to the desired output multiplied by the inverse of the feedback system transfer function if one asks for zero tracking error at sampled time. The discrete-time inverse transfer function for a majority of feedback control systems in the world is unstable. If a continuous-time system with a pole-zero excess of 3 or more is fed by a zero-order hold, and the discrete-time equivalent difference equation is computed using a reasonable sample rate, the inverse model is unstable [17]. Moreover, the stability of ILC needs a model that is accurate within plus or minus 90 degrees all the way to Nyquist frequency. Hence, ILC system is not robust to unmodeled high-frequency modes, parasitic poles, residual modes, etc. One can avoid potential instability by cutting off the learning using a zero-phase low-pass filter [18-19]. The cutoff frequency needs to be adjusted in hardware since one does not know what is wrong with the model.

This chapter starts with the ILC concept, but formulates a completely different objective, aiming to address the bandwidth limitation issues encountered by the feedback control system designer, rather than aiming to achieve zero tracking error. The approach simply creates a finite-time inverse model that is accurate up to the desired bandwidth, and assumes that one has a

reasonably good model up to this frequency. This approach simply adjusts the command based on the partial system model inverse and applies it to the system. Given the limited tracking accuracy requirements associated with the concept of the bandwidth in the original design, this approach could be sufficient to effectively raise the bandwidth to a higher level. This approach bypasses all of the issues described associated with ILC. But, in case, one wants to improve the performance, one can then apply ILC iterating with the real world, so that it corrects for model errors in the chosen frequency range.

3.2 Bandwidth Concept

The frequency response of feedback control systems plotted with logarithmic scales for both the magnitude response (log or dB) and frequency, can be approximated by straight lines with given slopes, which is routinely done when using Bode plots in classical design. For low frequencies, the plot is approximately horizontal, meaning that the amplitude of the response in this frequency range is close to a constant when the command is a sinusoid. At some frequencies, the amplitude response starts to decay, usually with a slope given by a factor of 10 reduction in output when the frequency is increased by a factor of 10. The aim of the definition of the bandwidth is to identify the frequency at which the system response starts such a decay for a sinusoidal command. Frequency components of the command below this frequency have the output reasonably close to the command amplitude, frequencies well above have the output much smaller than the command amplitude. The bandwidth indicates the frequency when the control system starts to ignore your command. This is straightforward for systems with dominant poles in the frequency plot being real. We will generalize the concept somewhat for complex conjugate dominant roots.

3.3 Bandwidth for Single Roots, Multiple Roots, and Dominant Roots

For a command of cos (ωt), the steady-state frequency response of transfer function $G(s)$ is $M(\omega)\cos(\omega t + \phi(\omega))$ where $M(\omega) = |G(i\omega)|$ and $\phi(\omega)$ is the phase angle of $G(i\omega)$. For a first-order differential equation $G(s) = b/(s + a)$

$$M(\omega) = \left(\frac{b}{a}\right)\left|\frac{1}{1 + i(\frac{\omega}{a})}\right| \tag{3-1}$$

For $\frac{\omega}{a} \ll 1$, then $M(\omega) \approx \left(\frac{b}{a}\right)$. For $\frac{\omega}{a} \gg 1$ then $M(\omega) \approx \left(\frac{b}{a}\right)(\frac{\omega}{a})^{-1}$, and the magnitude decays by a factor of 10 when ω increases by a factor of 10. The frequency dividing these two behaviors is $\frac{\omega}{a} = 1$ defining the bandwidth as a. At this frequency, the actual output amplitude has decreased by a factor of $\frac{1}{\sqrt{2}} = 0.707$ or -3.02 dB. As a result, the bandwidth is the frequency at which the amplitude of the output from a sinusoidal input has decreased by this factor 0.707 compared to that of a very low frequency or DC.

It is important to note that the bandwidth of the first-order system is associated with the root $(s + a) = 0$, which also determines the time constant of the first-order system as $1/a$. The time constant describes how fast the transients disappear with time. The time you have to wait for the initial condition influence on the output to be essentially gone is called the settling time, often defined as $4/a$. By waiting this much time the original initial value of the transient term Ce^{-at} has decayed to 1.8% of its original value, i.e. has negligible influence on the output. After this time the output is listening to the command, and not the initial conditions. A higher bandwidth indicates a shorter setting time.

In summary, the bandwidth is an indicator of what frequencies components of command are executed well in the response after transients are gone. It also indicates the settling time of

the transients, showing what part of the trajectory is listening to the command and not the initial conditions. And in addition, it can be shown that by indicating the settling time, it is telling the control system user to limit all commands to ones that do not change substantially within a settling time. Otherwise, the command will not be executed well. The discussion above suggests that the designer often may want a higher effective bandwidth for the feedback system.

Consider a higher order system with all real roots:

$$G(s) = \frac{b}{(s+a_1)(s+a_2)(s+a_3)}$$

$$M(\omega) = |G(i\omega)| = \frac{b}{a_1 a_2 a_3}\left|\frac{1}{1+i(\frac{\omega}{a_1})}\right|\left|\frac{1}{1+i(\frac{\omega}{a_2})}\right|\left|\frac{1}{1+i(\frac{\omega}{a_3})}\right|$$

$$logM(\omega) = log\left(\frac{b}{a_1 a_2 a_3}\right) + log\left|\frac{1}{1+i(\frac{\omega}{a_1})}\right| + log\left|\frac{1}{1+i(\frac{\omega}{a_2})}\right| \quad (3\text{-}2)$$

$$+ log\left|\frac{1}{1+i(\frac{\omega}{a_3})}\right|$$

The log-log plot is the sum of the first order plots like those in the previous section. Hence, if $a_1 < a_2 < a_3$ then the first term determines when the overall plot starts to decay, and a_1 is approximately the -3.02 dB down frequency, or the bandwidth. Note that if instead of having real roots a_2, a_3, they were a complex conjugate pair but the decay for a_1 still dominated the plot, then a_1 is still the bandwidth. Other comments above apply again.

Consider a transfer function with a complex conjugate pair of poles with damping ratio $0 < \zeta < 1$ and undamped natural frequency ω_n

$$G(s) = \frac{\omega_b^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \quad (3\text{-}3)$$

$$M(\omega) = |G(i\omega)| = \left(\frac{\omega_b^2}{\omega_n^2}\right)\left[\left(1 - \left(\frac{\omega}{\omega_n}\right)^2\right) + i2\zeta\left(\frac{\omega}{\omega_n}\right)\right]^{-1}$$

When $\left(\frac{\omega}{\omega_n}\right) \ll 1$ then $M(\omega) \approx \left(\frac{\omega_b^2}{\omega_n^2}\right)$. When $\left(\frac{\omega}{\omega_n}\right) \gg 1$ then $M(\omega) = \left(\frac{\omega_b^2}{\omega_n^2}\right)\left(\frac{\omega}{\omega_n}\right)^{-2}$, and in this region, when ω is increased by a factor of 10, the amplitude of the output is decreased by a factor of 100. These roots can have a resonant peak around $\left(\frac{\omega}{\omega_n}\right) = 1$ with output magnitude given by $M(\omega_n) = \left(\frac{\omega_b^2}{\omega_n^2}\right)\left(\frac{1}{2\zeta}\right)$. This is the point at which the asymptotic straight-line approximations start to decay. This is what we choose to call the bandwidth for such roots. The root locations are

$$s = -\zeta\omega_n \pm i\sqrt{1-\zeta^2}\omega_n \tag{3-3}$$

Note that the radial distance from the origin to the roots is equal to what we defined as the bandwidth for dominant complex roots, i.e. ω_n. Thus, when real roots determine the bandwidth, it is the distance from the origin to the root on the negative real axis, and when complex roots determine the bandwidth, it is again the radial distance from the origin to the roots. Dominant or slowest decaying roots on a circle centered at the origin all produce the same bandwidth according to our definition, whether the root or roots are real or complex.

3.4 Bandwidth Limitations in Feedback Control System Design

The classical feedback control system designer does not have the ability to pick whatever bandwidth he desires. Considering typical classical control laws, proportional (*P*), integral (*I*), proportional plus derivative (*PD*), and proportional-integral-derivative (*PID)*, adjusting the gain or gains can only reach certain limited bandwidths. There are three types of bandwidth limitation for the classical controller design.

First of all, the bandwidth is limited by the average root location constraint**.** We observe the relationship between the roots and the coefficients in the associated characteristic polynomial

$$s^3 + \alpha_2 s^2 + \alpha_1 s + \alpha_0 = (s - s_1)(s - s_2)(s - s_3)$$
$$= s^3 - (s_1 + s_2 + s_3)\, s^2 + (s_1 s_2 + s_2 s_3 + s_3 s_1) s - s_1 s_2 s_3 \quad (3\text{-}5)$$

Applying a *P* controller to such a third order system will introduce a gain in α_0, and leave all other coefficients unaltered. Applying an *I* controller will convert from 3rd degree polynomial to 4th degree, and again only influence the s^0 term. Examining *PD* and *PID* one concludes that none of the routine control laws can influence the coefficient of the next to the highest power. The second coefficient is the negative of the sum of all roots $s_1 + s_2 + s_3$. The average root location is,

$$\sigma = (s_1 + s_2 + s_3)/3 \quad (3\text{-}6)$$

If all roots are real, this condition says that no matter what controller one uses, one will not be able to influence the average position. Therefore, the maximum possible bandwidth that could possibly be achieved would occur when all roots were actually at the average position. For proportional control, the average position is already defined before you turn on the controller; for the integral controller alone, it raises the number of roots by one, and introduces a root at the origin. Both of which make the resulting average position have a smaller maximum possible bandwidth.

Consider if there were a complex conjugate pair. The real part of all the roots add up to this average position. The best possible situation is to have all real parts actually at the average position. According to the bandwidth defined above for complex conjugate roots, it is the radial distance to the roots on this vertical line through the average position that determines bandwidth. This suggests that one might be able to adjust the bandwidth substantially in this situation by making the imaginary parts become arbitrarily large. This comes at a cost of a reduced damping ratio and increasing the output overshoot. Design guidelines suggest that one would like to limit

the damping ratio ζ to be no less than 0.707, suggesting making the lines from the origin to the roots have an angle less than or equal to 45 degrees relative to the negative real axis. This rule limits that peak overshoot, in response to a step input, to a reasonable amount. And it prevents having undesirable high frequency oscillations in the transients. Using this rule, the bandwidth could be increased from the value associated with the average position on the real axis by only the square root of 2. Again, it is clear that there is a maximum possible bandwidth already determined before the control system designer even starts to adjust the gains available.

Second, the bandwidth is limited by the rule of root locus. Having all the roots line up in a manner so that one could reach the maximum possible values described in the previous section, is very unlikely. Consider all real roots and a proportional controller. The root locus plot produces asymptotic angles for the roots to approach as the gain increases. The centroid of the asymptotes is given by σ equal to the sum of the poles, divided by the number of poles (when there are no zeros). In other words, it is the average position of the roots that does not change with controller gain. The angles that the asymptotes made with the positive real axis are

$$\theta = \frac{180}{n} \pm k\frac{360}{n} \quad ; \quad k = 0,1,2,\ldots \tag{3-7}$$

This produces angles of asymptotes of the root locus going into the right half plane to be at ± 60 deg for 3rd order, ± 45 deg for 4th order, etc. Suppose the loci actually following these asymptotes, then the maximum distance from the origin to the complex roots would occur when the roots were crossing the imaginary axis, and the bandwidth would then by $\sqrt{3}\sigma/2$ for 60 deg, and σ for 45 deg.

Third, the bandwidth limitations are imposed by the feedback control system designer because of the hardware constraints**.** There are various control problems where the control system designer implements a low bandwidth, intentionally making the system have slow

performance like those in the controllers used on the Robotics Research Corporation robot [20]. Fitting frequency response data for each axis of this robot produces a third-order transfer function from command to the response as,

$$G(s) = \left(\frac{a}{s+a}\right)\left(\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}\right) \tag{3-4}$$

The robot uses harmonic drives, and although they are claimed to have no backlash, they are flexible, inserting a rotational spring constant between successive links. A string of rigid bodies connected by springs in this matter will have vibration modes. The complex conjugate roots in this transfer function represent the first vibration mode occurring at an undamped natural frequency $\omega_n = 37$ rad/s, or about 5.5Hz. The control system designer inserted the first factor into the closed loop behavior, producing a bandwidth of $a = 8.8$ or 1.4Hz. This starts the decay of output amplitude at 1.4 Hz, and by 5.6 Hz the output has been reduced from a DC gain of unity by 6 dB or a factor of 4. The intention is to detune the control system so that commands do not have much content at the resonant frequency, and hence, commands minimally excite the lowest frequency vibration frequency in the robot.

The same design process is regularly used in spacecraft attitude control. Spacecraft often have a relatively rigid hub, with various very flexible appendages attached, such as large unfolded solar panels, or antennae of various kinds. When one wants to rotate the spacecraft by rotating the hub, this excites vibrations of the flexible appendages. The approach to treating this problem is to impose a bandwidth that attenuates components of any command that causes vibrations of the appendages.

3.5 Singular Values of *P* Matrix of the System Gives the Magnitude Response of the System as the Matrix Size Goes to Infinity

Consider a closed-loop feedback control system represented in state variable form

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \qquad k = 0, 1, 2, \dots, p-1 \\ y(k) &= Cx(k) + v(k) \qquad\qquad k = 1, 2, 3, \dots, p \end{aligned} \tag{3-5}$$

where $x(k)$ is the state variable, $u(k)$ is the command, and $y(k)$ is the output. In some physical problems, there is a deterministic disturbance associated with the desired trajectory, and the $v(k)$ is included as the equivalent output disturbance in such cases. One wishes to have the output correspond to a desired trajectory $y^*(k)$ by proper choice of the command input.

The solution of the state-space equation is,

$$y(k) = CA^k x(0) + \sum_{i=0}^{k-1} CA^{k-i-1} Bu(i) + v(k) \tag{3-10}$$

Using the history vectors defined in Equation (1-2) produces,

$$\underline{y} = P\underline{u} + \overline{A}x(0) + \underline{v} \tag{3-11}$$

$$P = \begin{bmatrix} CB & 0 & 0 & \cdots & 0 \\ CAB & CB & 0 & \cdots & 0 \\ CA^2B & CAB & CB & \ddots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B & CA^{p-2}B & CA^{p-3}B\cdots & & CB \end{bmatrix} \qquad \overline{A} = \begin{bmatrix} CA \\ CA^2 \\ CA^3 \\ \vdots \\ CA^p \end{bmatrix} \tag{3-12}$$

Given the desired output $\underline{y}^*$, the input that produces it, $\underline{u}^*$, is given by

$$\underline{u}^* = P^{-1}[\underline{y}^* - \left(\overline{A}x(0) + \underline{v}\right)] \tag{6}$$

If one uses $\underline{u}^*$ as the command, one should expect the output $\underline{y}$ is exactly the desired trajectory $\underline{y}^*$, if one plugs Equation (3-13) back to Equation (3-11). Unfortunately, the matrix *P* is usually ill conditioned as a result of zeros that are introduced in the equivalent *z*-transfer function produced from a continuous time system fed by a zero-order hold. This is true whenever

the continuous-time transfer function has a pole excess of 3 or more and the sample rate is reasonable [20]. Usually, the command u uses the desired trajectory $\underline{y}^*$, and one can see from Equation (3-13) that for this case, the output will not be exactly as the desired trajectory.

In the math formation of ILC, it includes the disturbance and noise terms in Equation (3-11). For constant and repeated disturbance and noises, the formulation of error convergence equation will not change, thus the error will decrease to zero if the eigenvalues of $I - PL$ are less than 1 as Equation (1-7). ILC can cancel the tracking error if there is constant or repeated disturbance existing for the system. Phan shows that if there is non-repeated disturbance or white noise excitation, larger eigenvalues of $I - PL$ can lead to larger tracking error for the system, and one could add a small gain ϕ in front of L to turn down the learning speed to reduce the steady-state tracking error for the non-repeated disturbance and noise at a cost of a slower decreasing of tracking errors from iteration to iteration [21].

In fact, the P matrix of the system gives us the information about the steady-state response of the system. The following material is a replica of the proof by Chen and Longman to show this result. One can take the singular value decomposition of matrix P, $P = USV^T$, it presents a relationship between this decomposition and frequency response, and shows that as the size of the matrix gets large, the singular values in S approach the magnitude response of the system at the frequencies that can be seen in the p time steps of data. The left and right singular vectors in U and V respectively, look like sinusoids, and the relationship between each input and the corresponding output singular vector, contains the phase change associated with the frequency represented by the associated singular value [22].

The equation $U(z) = \sum_{k=0}^{\infty} u(k)z^{-k}$ is the z-transformation for an infinite sequence $u(k)$. Substituting $z = e^{i\omega T}$, one gets the frequency component of the infinite sequence $u(k)$.

Suppose $u(k)$ is p steps long, then one can only see the discrete frequency components of ω, where $\omega = (2\pi/p)n = \omega_o n$ for $n = 0, 1, \ldots p-1$. The discrete Fourier transform is

$$U\left(e^{i\omega_o n}\right) = \sum_{k=0}^{p-1} u(k)\left(e^{i\omega_o n}\right)^{-k} \tag{3-14}$$

Define $z_o = e^{i\omega_o}$, $U_n = U\left(e^{i\omega_o n}\right) = U(z_o^n)$, $\underline{U} = \left[U_0, U_1, \ldots U_{p-1}\right]^T$, one can write the relationship between $\underline{U}$ and $\underline{u}$ in matrix form, $\underline{U} = H\underline{u}$, where

$$H = \begin{bmatrix} (z_o^0)^0 & (z_o^0)^{-1} & \cdots & (z_o^0)^{-(p-1)} \\ (z_o^1)^0 & (z_o^1)^{-1} & & (z_o^1)^{-(p-1)} \\ \vdots & & \ddots & \vdots \\ \left(z_o^{p-1}\right)^0 & \left(z_o^{p-1}\right)^{-1} & \cdots & \left(z_o^{p-1}\right)^{-(p-1)} \end{bmatrix} \tag{3-15}$$

The corresponding inverse discrete Fourier transform is then $H^{-1} = (1/p)\left(H^*\right)^T$, where H^* is the complex conjugate matrix of H, thus

$$\underline{u} = \left(\frac{1}{p}\right)\left(H^*\right)^T \underline{U} \tag{3-16}$$

Note that for the above equation, one finds that $u(k)$ is just a linear combination of $e^{i(\omega_o n)k}$, which is also a linear combination of $\cos(\omega_o nk)$ and $\sin(\omega_o nk)$ for the discreted set of frequencies.

Refer to Equation (3-11) without considering the initial condition and disturbance, then it is $\underline{y} = P\underline{u}$. Multiplying on the left by H on both sides, and inserting $H^{-1}H$ in front of $\underline{u}$, one gets its discrete Fourier transform,

$$\begin{gathered} Y = EU \\ E = \left(\frac{1}{p}\right) HP(H^*)^T \end{gathered} \tag{3-17}$$

Let $\widehat{H} = \left(\frac{1}{\sqrt{p}}\right) H$, thus , $\widehat{H}^{-1} = \left(\widehat{H}^*\right)^T$, and one can rewrite

$$E = \widehat{H}P\left(\widehat{H}^*\right)^T$$
$$P = \left(\widehat{H}^*\right)^T E\widehat{H} \tag{3-18}$$

From Equation (3-16) we know that U_n is the coefficient of $e^{i\omega_o nk}$ in $u(k)$, and the system response is $Y_n = M_n e^{i\theta_n} U_n$, thus as the value of p is sufficient large, then matrix E must converge to a diagonal matrix

$$E = diag(M_0 e^{i\theta_0}, M_1 e^{i\theta_1}, \dots, M_{p-1} e^{i\theta_{p-1}}) \tag{3-19}$$

Recall Equation (3-18), We know that $P = \left(\widehat{H}^*\right)^T E\widehat{H}$. One can write $\widehat{H} = [h_0, h_1, \dots, h_{p-1}]^T$, and corresponding $\widehat{H}^* = [f_0, f_1, \dots f_{p-1}]$ where $f_n = e^{-i\theta_n} h_n$. The matrix P can also be decomposed using singular value decomposition so that $P = USV^T$, where the singular values on the diagonal of S are $[\sigma_1, \sigma_2, \dots, \sigma_p]$, and denote the associated singular vectors as the column partitions in matrices $U = [\, u_0, u_1, u_2, \dots u_p]$ and $V = [v_1, v_2, v_3, \dots v_p]$. Thus we have the following two equations

$$P = USV = \sigma_0 u_0 v_0^T + \sigma_1 u_1 v_1^T + \cdots + \sigma_{p-1} u_{p-1} v_{p-1}^T$$
$$P = \left(\widehat{H}^*\right)^T E\widehat{H} = M_0 f_0^* h_0^T + M_1 f_1^* h_1^T + \cdots + M_{p-1} f_{p-1}^* h_{p-1}^T \tag{3-20}$$

The difference between the above two is that h_n, f_n are complex but u_n, v_n are real. One can eliminate the complex parts by matching the conjugate pairs. Note h_1 and h_{p-1} are a complex conjugate pair, and both refer to the same frequency, thus $M_1 = M_{p-1}$

$$M_1 f_1^* h_1^T + M_{p-1} f_{p-1}^* h_{p-1}^T = M_1 f_1^* h_1^T + M_1 f_1 \left(h_1^*\right)^T = M_1 (2 f_{1R} h_{1R}^T + 2 f_{1I}^* h_{1I}^T) \tag{3-21}$$

Entries numbered k (starting from 0 at the top) in $h_{1R}, h_{1I}, f_{1R}, f_{1I}$ in the above are given by $\cos(\omega_o k), -\sin(\omega_o k), \cos(\omega_o k + \theta_n), -\sin(\omega_o k + \theta_n)$, all multiplied by the $1/\sqrt{p}$ original

normalization. We can renormalize then by setting $\widehat{h}_{1R} = \sqrt{2}h_{1R}$, and similarly for the other vectors, so that each is now of unit Euclidean norm.

Now we have a mapping between the SVD of P and the system frequency response. As the number of time steps p in the desired trajectory, and correspondingly the size of matrix P, tends to infinity, the following relationships to the steady state magnitude response M_i and phase change through the system θ_i are represented as seen in the matrix as follows. For $i = i, 2, \ldots, p/2$, where i is the index of the singular value and corresponding singular vectors, it has the following relationship

$$\begin{aligned} \sigma_i &= M_i \\ u_i(k) &= \sqrt{2/p}\, cos(\omega_0 ik) \\ v_i(k) &= \sqrt{2/p}\, cos(\omega_0 ik + \theta_i) \end{aligned} \tag{3-22}$$

For the rest of the singular values and singular vectors, the relation is

$$\begin{aligned} \sigma_i &= M_{p-i} \\ u_i(k) &= -\sqrt{2/p}\, sin(\omega_0 ik) \\ v_i(k) &= -\sqrt{2/p}\, sin(\omega_0 ik + \theta_i) \end{aligned} \tag{3-23}$$

The singular values of P give the magnitude frequency response. the column vectors of V are input sinusoidal signals, and the column vector of U are the corresponding output sinusoids. One must examine the singular vectors using discrete Fourier transforms to identify which singular value, and singular vector pair is associated with what frequency.

When p is not tending to infinity, we describe the corresponding values of $\sigma_i, v_{\mathrm{i}}, u_{\mathrm{i}}$ as giving the finite-time frequency response. This supplies a way to define a bandwidth for the finite-time problem. One can make a model of the input-output relation for each finite-time frequency response up to a chosen frequency, and invert only this part of the matrix P. By doing

so we avoid the common instability of P^{-1}. One can also avoid the need to use zero-phase low-pass filtering for robustification to unmodeled high frequency dynamics, assuming the bandwidth desired is not so high that the unmodeled parasitic poles affect the stability. One picks the bandwidth by simply choosing the singular values and vectors for frequencies up to this bandwidth, or perhaps a bit beyond if desired. We will illustrate this idea in detail in the next section. Note that during ILC iterations, it shows that there can be accumulation of error in the unaddressed part of the space [22]. One may want to limit the number of ILC updates for this reason.

3.6 Partial Inverse of the System *P* Matrix

Generally speaking, the command to the system is usually the desired output. The ideal command would be the system inverse times the desired output so that this command after it is given to the system, will automatically give the desired output. As mentioned before, this very often produces an unstable control action. This instability is manifested in matrix *P* by having a particularly small singular value (and associated singular vectors, one growing and the other decaying exponentially with time steps). These difficulties are avoided here by multiplying the desired output by the partial inverse that does not contain the small singular value(s), and only contains singular values for the chosen frequency range as the command to the system. Assume matrix *P* of the system in terms of the desired singular values of S_D, which corresponds to the magnitude in the frequencies you don't want to cut off, and the associated singular vectors U_D and V_D, and $S_\perp$ which corresponds to the frequency components above the cutoff frequency and set to zero.

$$P = [U_D \quad U_\perp] \begin{bmatrix} S_D & 0 \\ 0 & S_\perp \end{bmatrix} \begin{bmatrix} V_D^T \\ V_\perp^T \end{bmatrix} = U_D S_D V_D^T \tag{3-24}$$

By doing this, we are addressing the output below our desired frequency and we leave the ones above the frequency untouched. Introducing this into the input-output equation for the history vectors

$$\begin{aligned} \underline{y} &= U_D S_D V_D^T \underline{u} + \overline{A}x(0) + \underline{v} \\ \left(U_D^T \underline{y}\right) &= S_{\_D}\left(V_D^T \underline{u}\right) + U_D^T(\overline{A}x(0) + \underline{v}) \\ \underline{y}_{\_D} &= S_D \underline{u}_D + U_D^T(\overline{A}x(0) + \underline{v}) \end{aligned} \tag{3-25}$$

The desired trajectory is to be specified as a linear combination of the chosen output basis functions according to $\underline{y_D^*} = \left(U_D^T \underline{y^*}\right)$ and the input command that produces this output $\underline{u_D^*} = \left(V_D^T \underline{u^*}\right)$ is then given as

$$\underline{u_D^*} = S_D^{-1} \underline{y_D^*} - S_D^{-1} U_D^T(\overline{A}x(0) + \underline{v}) \tag{3-26}$$

Rewrite Equation (3-26) and one can see that the command $\underline{u^*}$ we use as command to the system is

$$\underline{u^*} = V_D S_D^{-1} U_D^T \underline{y^*} - V_D S_D^{-1} U_D^T(\overline{A}x(0) + \underline{v}) \tag{3-27}$$

If one plugs Equation (3-27) back to Equation (3-11), we could see this command $\underline{u^*}$will produce exactly the desired output y^* for frequencies below the desired frequency, that we have chosen, and leaves the other frequencies untouched.

For the next three sections, we will illustrate the use of partial inverse of the system in three cases.

3.7 Partial Inverse Solution to Raise Bandwidth When Model is Good Up to Desired Bandwidth

The stated aim of producing a higher bandwidth is not asking for high accuracy. At the bandwidth of a first order system the response to an input at the bandwidth frequency is

attenuated by 30% and the phase angle is wrong by 45 degrees. Since one is inverting the system only up to the chosen raised bandwidth, it could often be the case that the model is good enough in this frequency range that the partial inverse solution gives acceptable accuracy.

If this is the case, the procedure is very simple. Having a desired finite time trajectory, compute the needed input history from Equation (3-27). Then apply it as the command input to the feedback control system, instead of commanding the desired output as one normally does with a feedback control system.

To compute the new command, one needs the *A*, *B*, *C* system matrices of the state variable model. If there is no repeating disturbance $\underline{v}$, this is enough. Otherwise, one needs to know $\underline{v}$. In this case, one makes one run to modify the command to address the influence.

3.8 Handling a Deterministic Disturbance Associated with the Desired Trajectory

Various problems have a disturbance function related to the command being executed, that occurs every run. The gravity torque disturbance on a robot link as the link moves along the desired trajectory can be modeled this way. If one is confident of the *A*, *B*, *C* model but are not able to give a good model of the repeating disturbance, one can make one run to produce the needed information, and then apply the inverse solution. First, apply the inverse solution computed by Equation (3-27), but without the $\underline{v}$ term, to the system

$$\underline{u}_D^{**} = S_D^{-1}\underline{y}_D^* - S_D^{-1}U_D^T\overline{A}x(0) \tag{3-28}$$

The output is then $\underline{y}_D = \underline{y}_D^* + U_D^T\underline{v}$. Then, in the next run one has the inverse solution, including the effect of the repeating disturbance, by applying

$$\underline{u}_D^* = \underline{u}_D^{**} + S_D^{-1}(\underline{y}_D - \underline{y}_D^*) \tag{3-29}$$

3.9 Correcting for Model Error by ILC Iterations

If the partial inverse model does not supply the accuracy that one desires, one can perform iterative learning control iterations. Apply any input to the feedback control system for the first run, for example, the desired output, or the inverse solution obtained above. Then update the command from iteration *j* to iteration $j+1$ according to a chosen learning law

$$\underline{u}_{D,j+1} = \underline{u}_{D,j} + \phi L \underline{e}_{D,j} \tag{3-30}$$

where ϕ is an overall gain, *L* is the leaning gain matrix, and the error at run *j* is defined as $\underline{e}_{D,j} = \underline{y}_D^* - \underline{y}_{D,j}$. Then

$$\begin{aligned} \underline{y}_{D,j} &= S_D \underline{u}_{D,j} + U_D^T(\overline{A}x(0) + \underline{v}) \\ \underline{y}_{D,j+1} &= S_D \underline{u}_{D,j+1} + U_D^T(\overline{A}x(0) + \underline{v}) \\ \underline{e}_{D,j+1} - \underline{e}_{D,j} &= -S_D(\underline{u}_{D,j+1} - \underline{u}_{D,j}) \\ \underline{e}_{D,j+1} &= [I - \phi S_D L]\underline{e}_{D,j} \end{aligned} \tag{3-31}$$

The error in the addressed part of the error space will converge to zero as the iterations progress for all error histories in the initial run, if and only if all eigenvalues of the matrix $[I - \phi S_D L]$ are less than unity in magnitude. There are various choices for the learning gain matrix. The *P* transpose contraction mapping law picks *L* as S_D, making the eigenvalues equal $1 - \phi\sigma_i^2$. This law is particularly robust to model errors, but converges very slowly at high frequencies. The partial isometry law makes *L* the identity matrix and the eigenvalues are $1 - \phi\sigma_i$ which learns substantially faster, but still attenuates as the frequency goes up to account for the usual increased model error at higher frequencies. Of course, $L = S_D^{-1}$ produces the inverse solution and converges to zero error in one run if the model is perfect. One could use this for the

low frequencies where one trusts the model, and transition to one of the previous laws as the frequency goes up.

In general, in iterative learning control, one uses a zero-phase low-pass filter to cutoff learning at frequency to increase the robustness of the system to high frequency model uncertainty. Here, we are interested in correcting the errors below the desired bandwidth frequency, and welcome the fact that these can be eliminating faster without robustness issues because the model is likely to be reasonably accurate in this frequency range. Since the partial inverse based on bandwidth does not seek zero error, one can iterate until the error stops decreasing to correct for error in the partial inverse model, but one would not normally need a low-pass filter.

3.10 Simulation Plant Model

Simulations are performed for the robot link system model as shown in Equation (3-8). The system is sampled at 100 Hz. The robot link is tracking a trapezoidal trajectory y_d from 0 degrees to 90 degrees in 2 seconds, as shown in Figure 3-1. The trajectory holds 0 degrees for 0.2 seconds at the start, and the trajectory remains at 90 degrees for 0.2 seconds at the end. In between, the trajectory moves smoothly from 0 to 90 in 1.6 seconds using a constant velocity segment and parabolic blends. We assume that the robot link has all zero-initial conditions $x(0) = 0$, and the disturbance v is modeled as a constant -1 degree disturbance adding to the final output due to the gravity. The figures of simulation results are at the end of this chapter.

3.11 Using Partial Inverse of System as Prefilter to Modify the Command

The robot link input and output relation are characterized by Equation (1-4) after applying the zero initial conditions

$$\underline{y} = P\underline{u} + \underline{v} \tag{3-32}$$

Usually, the input $\underline{u}$ in this equation will use $\underline{y_d}$, which is the desired trapezoidal trajectory. Instead, use the command suggested by Equation (3-27) which uses the difference between the desired trajectory y_d and the constant disturbance v, and then applies it as the input to the robot link

$$\underline{u} = P^*(\underline{y_d} - \underline{v}) \tag{3-33}$$

Figure 3-2 compares the resulting output of the robot link using this technique as command, and the one using the desired output as a command. The red solid line is the desired output, the trapezoidal trajectory; the blue dash-dot line is the output of the robot link using the desired trajectory y_d as command, and the black dash line is the output of the robot link using the modified command by the prefilter in Equation (3-33) as the command. Using the desired output as command, the output produced is far from desired. On the other hand, using the partial inverse technique, the output is tracking the system much better, but with visible oscillation at the end of the trajectory.

The oscillation feature of the output using the partial system inverse as command appears from the start to the end. Figure 3-3 shows the enlarged portion of Figure 3-2 from 0.85 seconds to 0.95 seconds. One can see the tiny oscillation of the black dash line which is the output of the system using the partial inverse technique, while it is following the trajectory. Figure 3-4 shows the enlarged Figure 3-2 between 1.8 seconds and 2 seconds. The oscillation phenomenon is more obvious at the end of the trajectory compared to its start. The obvious oscillation effect at the end is not diminished even if one extends the trajectory. Experiments with a trapezoidal trajectory of 3 seconds varying from 0 degree to 90 degree with a duration of 90 degree hold still for 1 second still shows the oscillation effect at the tail of the trajectory. This means that this effect has little

to do with the transients of the system. One can also verify by checking the two peaks at Figure 3-4, one peak is at around 1.85 seconds, and the other peak is at around 1.97 seconds, thus the period of the signal is about 0.12 second, which is roughly equal to 8Hz, which is equal to the designed cut-off frequency of our partial inverse matrix.

This phenomenon resembles the Gibbs phenomenon where there is an overshoot of Fourier series and other eigenfunction series occurring at simple discontinuities. In this case, the singular values in partial inverse matrix increase from 1.0137 to 9.2556 and the rest are all zeros. There is a sharp discontinuity in the singular matrix. One might guess if we smooth the singular value in the partial inverse matrix, the oscillation effect at the tail might be reduced. And this is the case. One uses a modified partial inverse matrix where the singular value, starting from 9.256, decreases linearly to zero in 16 entries, which means the next singular value next to 9.2556 is set to 15/16 of this number. Figure 3-5 shows the comparison between the output using a modified partial inverse matrix as magenta dash line and output using partial inverse matrix as black dash line at the tail end of the trajectory. Red solid line is the desired output. One sees that with the smoothing of singular values of the partial inverse matrix, the oscillation effects at the end is reduced.

One can also reduce this oscillation effect by using a higher desired frequency. Figure 3-6 shows the output of the robot link using the partial inverse matrix with a 16Hz cut-off frequency. And one can barely see the oscillation effect at the tail end of the trajectory.

3.12 Use Partial Inverse of System as a Learning Matrix in One-Step of Iterative Learning

Another approach to use the partial inverse of the system is to use it as the learning gain matrix to do one-iteration of iterative learning, in order to eliminate the influence of a repeating

unmodeled disturbance. In Section 3.11, the output of the robot link is given by the command $L(\underline{y_d} - \underline{v})$, assuming the disturbance is known. Here, the repeated disturbance is not know. The initial command to the system is the partial inverse of the system L to multiply the desired trajectory y_d Then using this as the input $\underline{u_1} = P^* \underline{y_d}$ to the robot link produces output $\underline{y_1}$. In the next iteration, use the command in the previous iteration plus error times the partial inverse of the system L as the new command for the system, and use it as the command for the system afterwards. This corresponds to the one-iteration learning control, where

$$\underline{u} = \underline{u_1} + P^*(\underline{y_d} - \underline{y_1}) \tag{3-34}$$

Figure 3-7 shows the results of both approaches. The partial inverse of system is cutoff at 16Hz. Red solid line is the desired output $\underline{y_d}$, blue dash-dot line is the output of the robot link using the one-step of iterative learning as Equation (3-34), and the black dash line is the output of the robot link using as prefilter assuming we know the disturbance as Equation (3-33). In the simulation, the disturbance v is a constant -1 degree adding to the output. The different is that, for the first approach of the prefilter, we assume we know $v = -1$; for one-step ILC, we do not know the form of v, and the command to system only use the output information in the previous iteration. The one-step iterative learning approach is good at handling constant disturbance even when one does not know its form, and this is the strength of the iterative learning. Moreover, the one-step iterative learning also improves the tracking performance compared to simply using the partial inverse as the prefilter to modify. But the oscillation phenomenon still exists, and one can attenuate it by phasing out the cutoff, or raising the corresponding cut-off frequency of the partial inverse as stated in the previous section.

Figure 3-8 shows is the enlarged version of Figure 3-9, as we can see, use one-iteration of learning using the partial inverse of the system, the tracking performance improves.

3.13 Use Partial Inverse of the System in Iterative Learning Control

One of the strengths of using the partial inverse of the system as a learning gain matrix is that it can finish the learning process in one step. A limitation is that the model used to create the partial inverse is imperfect. The field of Iterative Learning Control iterates with the real world instead of a model of the world, aiming to get zero error in the world. Hence, it is compensating for model error. We can apply ILC instead aiming only for zero error in the space of the addressed frequencies up to the bandwidth. The result is an improved performance within the bandwidth. Because the bandwidth should normally be in a range where the model inaccuracy is small, the usual slow learning at high frequencies in ILC is not necessary for robustness in this situation. Here we investigate benefit of applying additional ILC iterations to improve the performance within the chosen bandwidth. A small number of iterations can significantly improve performance.

The simulation settings are the same. The system is still the 3^{rd} order robot link with constant disturbance of -1 degree on the system. The desired trajectory is still the trapezoidal curve from 0 to 90 in 2 seconds. A simulation is done for two learning gain matrices: one learning gain matrix is the partial inverse of this robot link up to 16 Hz, and the other learning gain matrix is the contraction mapping law as stated in Chapter 1. Figure 3-9 shows the result of this approach in ILC in 3 iterations. The red solid line the desired trajectory, the blue dash-dot line is the output of the robot link using the partial inverse of the robot link system as the learning gain matrix, and the black dash line is the output of the robot link using contraction mapping law as the learning gain matrix. After just three iterations, ILC using partial inverse show faster convergence than the traditional contraction mapping method. Figure 3-10, and

Figure 3-11 are enlarged version of Figure 3-9 show that the partial inverse of the system as the learning gain matrix can converge faster in small number of iterations.

3.14 Conclusion

When designing feedback control systems, one is not able to adjust the usual control gains to achieve any desired bandwidth. It is easy to have the situation where one is unable to make the feedback control system perform desired fast maneuvers that require a high bandwidth. Often the system model is good up to the desired higher bandwidth, but becomes poor at particularly high frequencies where there can be unmodeled parasitic poles or residual modes. For this case, a method is developed that allows the feedback control system to perform as if it has the higher bandwidth. A method is presented to make an inverse model limited to the frequency range needed. Then instead of simply commanding the desired trajectory to the feedback control system, and having a poor response because the trajectory has frequency content above the control system bandwidth, one can use the inverse model for this frequency range to find the command needed, and simply apply this as the command input to the control system. Provided the model is good in this frequency range, one achieves the performance of a system with the desired higher bandwidth, by simply applying a modified command. Bandwidth is a steady state frequency response concept and does not precisely apply to finite time trajectory tracking. The partial inverse needed for this approach is generated using a singular value decomposition of the input-output matrix. This can generate a finite-time frequency response model which is used to interpret the bandwidth requirement in terms of finite time trajectories.

If one wants to improve performance beyond what is achieved by use of the partial inverse model, one can apply ILC iterations to converge to zero error in the addressed finite-time

frequency response part of the output space. ILC usually aims for zero tracking error up to as high a frequency as possible. This introduces various issues of robustification to high-frequency model errors. This thesis seeks to address the bandwidth frustrations of feedback control system designers, and this modified objective bypasses many difficult issues in Iterative Learning Control. The partial inverse model may produce the desired tracking accuracy, or one can start from this result and apply ILC to improve the tracking within the chosen frequency range.

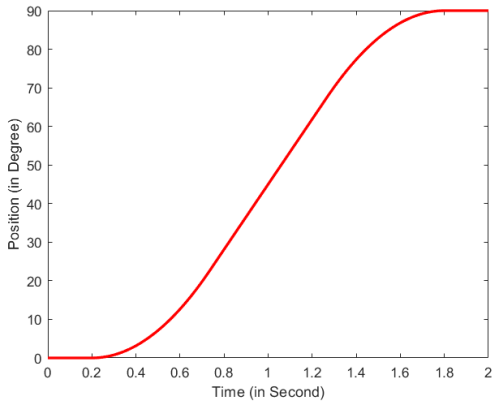

Figure 3-1. The desired output, a trapezoidal trajectory from 0 to 90 degrees in 2 seconds

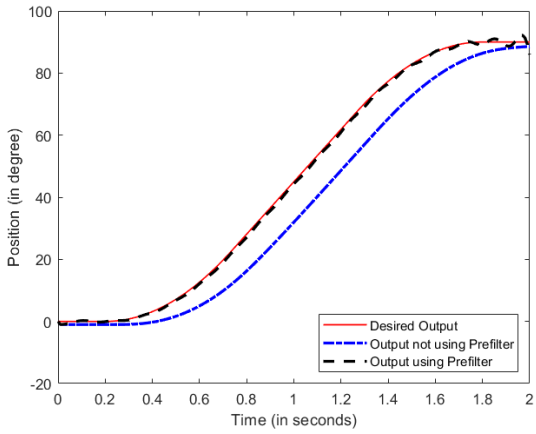

Figure 3-2. Robot link output using the partial inverse to modify command vs. using the desired output as the command

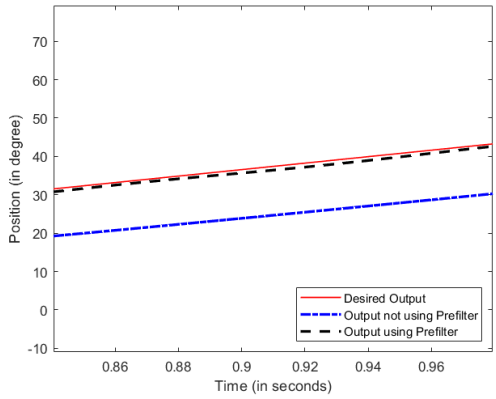

Figure 3-3. Oscillation effect of using partial inverse as a prefilter with cut-off at 8 Hz between 0.85s to 0.95s

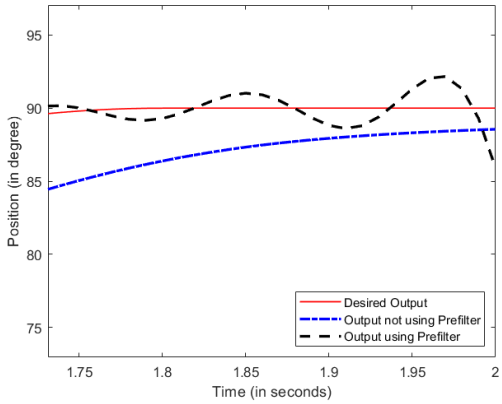

Figure 3-4. Oscillation effect of the partial inverse as a prefilter with a cut-off at 8 Hz between 1.8s to 2s

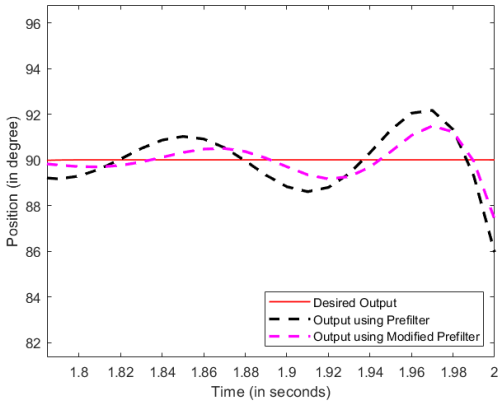

Figure 3-5. Reduced oscillation by using partial inverse as a prefilter with phasing out the cutoff

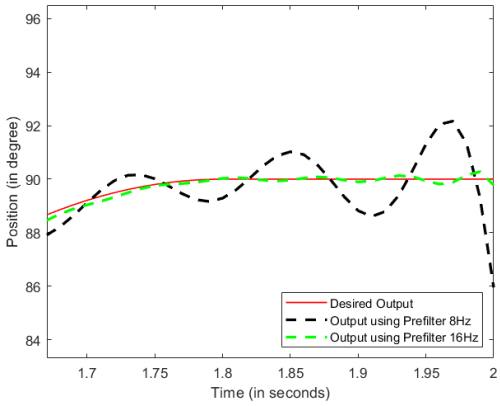

Figure 3-6. Reduced oscillation effect using partial inverse as a prefilter with cut-off frequency at 16 Hz

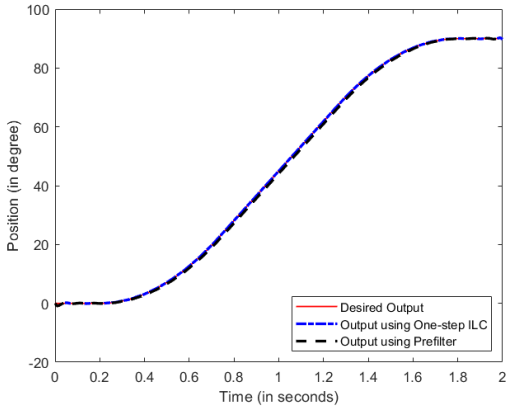

Figure 3-7. Robot link output using partial inverse in one-step learning vs prefilter

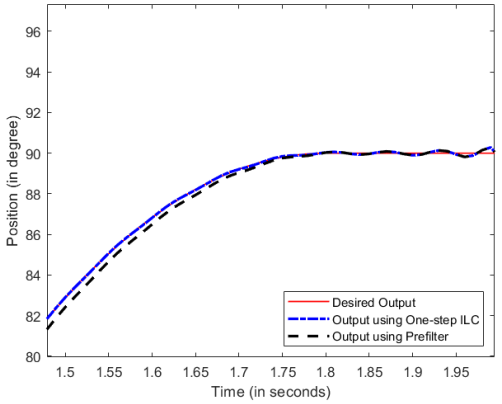

Figure 3-8. A reduced oscillation effect using a modified partial inverse matrix

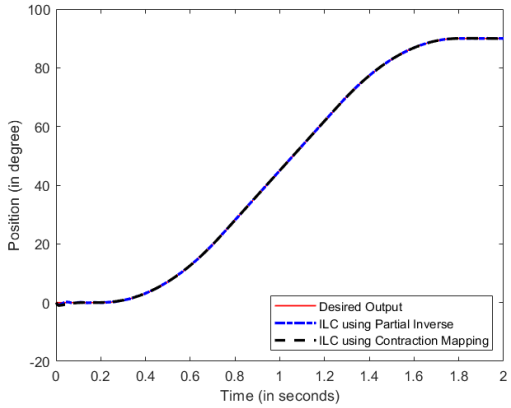

Figure 3-9. Robot link output using partial inverse as learning gain matrix vs output using contraction mapping law in 3 iterations

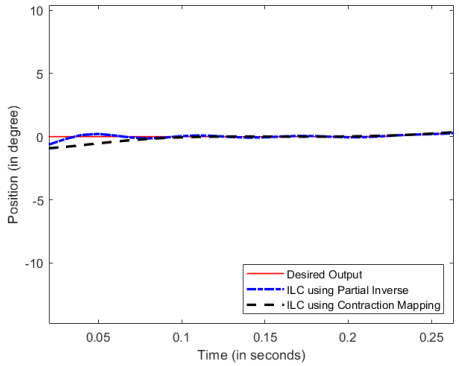

Figure 3-10. The enlarged version of Figure 9 from 0 sec to 0.25 sec

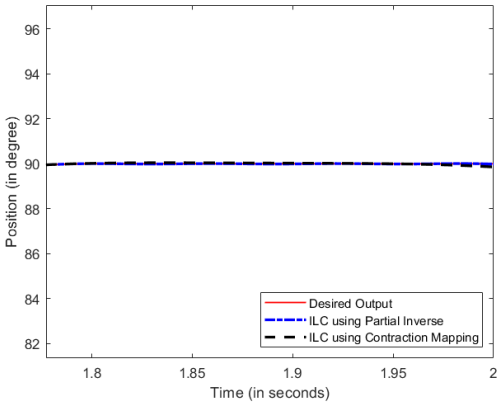

Figure 3-11. The enlarged version of Figure 9 from 1.9 sec to 2.0 sec

Chapter 4: On the Choice of Filtfilt, Circulant, and Cliff Filters for Robustification of Iterative Learning Control

The original aim of ILC is to converge to zero-tracking error at every time step for a finite-time trajectory, which is equivalent to zero error for all frequencies up to Nyquist frequency. This is in contrast to classical feedback control design that aims and sometimes struggles to achieve a desired bandwidth, the upper limit of reasonable performance. Achieving zero tracking error at all frequencies pushes our ability to create sufficiently accurate models. Parasitic poles or residual modes at high frequencies can destabilize the ILC learning process. Hence, a noncausal zero-phase low-pass filter is used for robustification to the model errors at high frequencies. Such zero-phase low-pass filter are often generated by the MATLAB Filtfilt command which uses a causal low-pass filter to filter the signal forward and backward. Initial conditions are needed for both forward and backward filtering processes, producing transients at the start and the end of the filtered result. These transients are unrelated to the frequency cutoff objective. The Filtfilt command calculates optimized initial conditions to reduce these transients [23]. Previous publication demonstrated that the initial condition picked by Filtfilt could cause instability of ILC [19][24]. Two alternative filters are presented here, a Circulant Filter, and a Cliff Filter. A math proof is presented to show that the Cliff Filter is a special case of a Circulant Filter. The Circulant Filter gives the steady-state response of a filter for a finite-time signal, and when it is used as a zero-phase low-pass filter in ILC, it gives the steady-state response without transients. This solves the instability issue of the Filtfilt command.

4.1 Zero-Phase Filtering in ILC

The solution to stabilization of ILC in the presence of high frequency model errors, is to use a zero-phase low-pass filter to filter the command given to the feedback system, cutting off

the learning process at frequencies where model error or unmodeled high frequency dynamics cause instability of the learning process. The phase error of the feedback system model is primarily responsible for this instability, thus the low-pass filter in ILC needs to be a zero-phase filter.

This chapter discusses zero-phase low-pass filtering used in ILC. For each run, the filtering is applied to the command to the feedback system, and the stability of ILC is achieved at the expense of not asking for zero tracking error above the cutoff frequency of the zero-phase low-pass filter. A zero-phase low-pass filter generated by the Filtfilt command in MATLAB, can starts with an IIR Butterworth low-pass filter, apply it to the finite-time signal from the first step to the last producing attenuation above the cutoff, and also put in phase lag since it is not a zero-phase filter. Then the filtered result is filtered backward in time producing twice as much attenuation above the cutoff and putting in phase lead that cancels the phase lag introduced by the forward filtering, then reverse the second filtered result. For both forward and backward filtering, the IIR filter needs initial conditions, and the Filtfilt command calculates the optimized initial condition to reduce the transients [24]. The previous ILC applications use the zero-phase low-pass filter generated by the Filtfilt command [25-26]. But the Filtfilt command does not eliminate the transients in the output, and research has shown that such transients can destabilize the ILC system [19][24]. Bing and Longman suggests to use a circulant Butterworth filter which gives a close approximation to the steady-state response of a filter for a longer input signals [24]. In this chapter, it is proved that the Circulant Filter gives the steady-state response of a filter for any length of the input signal, and when it is used as a zero-phased filter, it eliminates the initial condition. Plotnik and Longman suggest to use a Cliff Filter – a finite-time version of the ideal filter [19]. We prove that the Cliff Filter is a special case of Circulant Filter.

4.2 The Need of Frequency Cutoff in ILC

The simplest form of ILC uses a learning gain matrix $L = \phi I$, where ϕ is a scalar gain, and I is the identity matrix as in Equation (1-6). This modifies the command to the feedback system by a constant multiplying the error observed in the last run. If the feedback system is at the steady-state and the error e happens to be a sinusoidal function, then its product $Le = \phi e$, the corrective signal, is modified by the magnitude and phase change of the system. Many systems have a phase change of -180 deg or more at higher frequencies, and for any error above such frequencies, the phase change has the effect of reversing the sign of the corrective signal Le. Therefore, errors at these frequencies will be amplified. It is the task of the ILC designer to create a learning gain matrix L that acts as a compensator, doing what it can to cancel the phase change produced going through the feedback system.

However, such a compensator L requires one to have an accurate system model up to the Nyquist frequency and it is often not realistic. The issue can be illustrated by the ILC design for the Robotics Research Corporation robot at NASA Langley Research Center [25]. The feedback controllers for each joint of the 7 degrees of freedom robot had a bandwidth of 1.4 Hz. The Nyquist frequency was 200 Hz. Each joint is controlled by a DC motor on the previous joint running through a harmonic drive that has some flexibility. This makes a chain of masses connected by rotational springs. There will be a set of vibration modes for such a string of masses. The first mode was in the region of 5.7 Hz. The second mode was not easily identified but was around 18 Hz. It was not possible to perform frequency response tests at this frequency and above because of the small output signals. The dynamics between 18 Hz and Nyquist at 200 Hz are unknown. One kind of modeling suggests that there should be five more vibrations modes that were well above the range of frequencies where we could test the response.

ILC does not care if we were able to model at these high frequencies. If the model is sufficiently wrong in its phase, the ILC will persist in making the error grow. The error may appear to be buried in the noise level, but if the lack of a good model at these higher frequencies results in phase errors violating $|1 - PL| < 1$, then the error will grow. Eventually, it will rise above the noise level and produce clear exponential growth of the error.

4.3 MATLAB Filtfilt

Perhaps the default zero-phase low-pass filter is a Butterworth low-pass filter made into a zero-phase filter. One can use other filters, like Chebyshev, but Butterworth stays at or below the desired unity output up to the cutoff frequency, making it possible to cut off at a higher frequency as the model error grows with increasing frequency, and it starts with a continuous-time Butterworth filter, converts it to discrete-time using the bilinear transformation, and does the pre-warping to tune the cutoff [27]. Such a filter can be put in state variable form as in Equation (1-1), but with a direct feedthrough *D* term. Then there is an equation relating input to the output of the form of Equation (1-5).

To make a zero-phase filter, one first filters the signal forward in time, producing attenuation above the cutoff, but producing phase lag at the same time in the signal. The filter used in this first step is called forward filter. Then one reverses the time in the output sequence and filters it again. This doubles the attenuation above the cutoff and puts in phase lead to cancel phase lag produced in the forward filtering. The filter used in this second step is called backward filter. Then one reverses the time in the final output to revert to forward time. This is called forward-backward filtering. One could also use backward-forward filtering to achieve zero-phase as well, which is to reverse the input sequence first, filter it, and reverse the output and filter it

again. Note that there are initial conditions needed in forward filtering, and also initial conditions needed in backward filtering [23].

Recall Equation (1-5), we denote the P matrix of the forward filter by H_f, its observability matrix as O_f, and this IIR filter needs initial conditions as indicated by x_0; correspondingly, the backward filter's P matrix is denoted by H_b, observability matrix as O_b, and the initial conditions as x_{N-1}.The output of the forward filter and the backward filter can be expressed respectively as

$$\begin{aligned} Y_f &= H_f U + O_f x_0 \\ Y_b &= H_b Y_f^R + O_b x_{N-1} \end{aligned} \tag{4-1}$$

Use R to denote the action of reversing the output, which is a row reversing operator. Denote C as a column reversing operator. Below are some easily proved properties for row and column reversing operators.

$$A^{RC} = A^T \; if \; A \; Toeplitz \; matrix \tag{4-2}$$

Using the results in Equation (4-2), the output of forward-backward filtering Y_{fb} can be expressed as,

$$Y_{fb} = \left(H_b Y_f^R + O_b x_{N-1}\right)^R = H_b^T H_f U + H_b^T O_f x_0 + O_b^R x_{N-1} \tag{4-3}$$

and the output of backward-forward filtering Y_{bf} can be expressed as (note that, $H_b^T H_f \neq H_f H_b^T$)

$$Y_{bf} = H_f H_b^T U + H_f O_b^R x_{N-1} + O_f x_0 \tag{4-4}$$

MATLAB uses the command Filtfilt, using forward-backward filtering as in Equation (4-4), to achieve zero-phase filtering but with optimized initial conditions to reduce the transients. The parameters of the Filtfilt command include the input signal and the base filter system model, eg. the transfer function or state-space equation of the Butterworth filter. The Filtfilt command in an early version of MATLAB extends the input by reflected thorough the end-points [19]. A

number of points, which its number is equal to three times the filter order, are added to both ends of the input. The signal values are chosen as an odd reflection about the end value of the signal – making a mirror reflection of this number of endpoints in the signal, and then reflect in the vertical plane about the end value. Plotnik and Longman showed that this approach, when applied in the context of iterative learning control, could could destabilize the system [19].

The current MATLAB Filtfilt command uses the method of calculating the optimized initial conditions for both forward filter and backward filter to reduce the effects of transients [24]. The approach to pick the initial conditions for the forward filter and that for the backward filter seeks to make the forward-backward filtering result Y_{fb} and the backward- forward filtering Y_{bf} results as close as possible to each other, equivalent to minimize $\left\|Y_{fb} - Y_{bf}\right\|_2^2$, meaning the square of its Euclidean norm.

$$Y_{fb} - Y_{bf} = \left(H_b^T H_f - H_f H_b^T\right)U + (H_b^T - I)O_f x_0 + (I - H_f)O_b^R x_{N-1} \tag{4-5}$$

Note that Equation (4-5) is a linear equation for x_0 and x_{N-1}, the minimizing arguments are given by the least square estimate, and the optimal initial conditions are given as

$$\begin{pmatrix} x_0^{opt} \\ x_{N-1}^{opt} \end{pmatrix} = \left[(H_b^T - I)O_f, \quad \left(I - H_f\right)O_b^R\right]^+ (H_b^T H_f - H_f H_b^T)U \tag{4-6}$$

where $+$ superscript means the pseudoinverse. The corresponding output given by the Filtfilt command is expressed as

$$Y = H_b^T H_f U + H_b^T O_f x_0^{opt} + O_b^R x_{N-1}^{opt} \tag{4-7}$$

4.4 The Circulant Filter

The previous section presents how MATLAB creates a zero-phase filter: first design a base filter such as a low-pass Butterworth filter, then use forward-backward filtering to have a zero-phase filter, with optimized initial conditions reduce the transients in the output. This

section introduces the zero-phase Circulant Filter. A Circulant Filter H comes from making the Toeplitz matrix of the typical low-pass filter, like Butterworth filter, into a circulant matrix.

A general form of an $n \times n$ circulant matrix C_{cir} is

$$C_{cir} = \begin{bmatrix} c_0 & c_{N-1} & c_{N-2} & \cdots & c_1 \\ c_1 & c_0 & c_{N-1} & \cdots & c_2 \\ c_2 & c_1 & c_0 & \cdots & c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{N-1} & c_{N-2} & c_{N-3} & \cdots & c_0 \end{bmatrix} \tag{4-8}$$

The circulant matrix of a filter gives a steady-state response of a filter. Song and Longman first considered the concept of applying the circulant low-pass filter in the ILC problem [22]. One can derive the Circulant Filter, starting from *H* that is a lower triangular Toeplitz matrix of Markov parameters of a low-pass filter. The Circulant Filter is achieved by filling in the upper zero triangular entries with Markov parameters to produce matrix $\widehat{H}$. The first column is the *N* Markov parameters. The second column is obtained by moving the original column entries down one, and the last Markov parameter that is moving out of the bottom of the matrix is inserted in place of the zero above the diagonal. The remaining columns are filled analogously.

One can make it a zero-phase filter by applying forward-backward filtering as we did before. The Circulant Filter is shown in the following section to directly give the filter's steady-state frequency response. It is proved that if we use the circulant low-pass filter as the base filter, and we make it a zero-phase filter, the optimized initial condition in Equation (4-6) are zero.

The Toeplitz matrix of a filter does not give us a steady-state response of a filter, and it has transients in it. Chen and Longman consider the filter input and output relationship in matrix form as in Equation (1-5), where the *P* matrix is a lower-triangle Toeplitz matrix of Markov parameters [22]. Equation (1-5) uses *P* to represent the feedback system whose command is

adjusted by ILC. The same form represents the input-output relationship of a chosen filter (with appropriate adjustment of the time delay). Chen and Longman show that as the size of the *P* matrix increases, its singular values converge to the magnitudes of the steady state magnitude frequency response, and the relationship between the input and output singular vectors contains the phase of the frequency response [22]. Therefore, a finite size *P* matrix of the classic IIR or FIR filter as in Equation (1-5) only represents the intended steady-state response as the matrix size *N* tends to infinity.

4.5 Circulant Matrix Properties

Recall the definition of DFT, where $\omega = e^{-i2\pi/N}$

$$X(k) = \sum_{n=0}^{N-1} x(n)\omega^{kn} \tag{4-9}$$

The matrix form of Equation (4-9) is

$$X(k) = Wx(n) \tag{4-10}$$

The DFT matrix *W* has column partitions, $W = (w^0, w^1, \dots, w^{N-1})$, and *W* is a symmetric matrix with columns $w^k, k = 0,1, \dots, N-1$

$$w^k = \left[\omega^{0k}, \omega^{1k}, \omega^{2k}, \dots, \omega^{(N-1)k}\right]^T \tag{4-11}$$

Multiplying any vector by *W* produces its discrete Fourier transform (DFT).

Next show that the columns of the DFT matrix w^k are eigenvectors of a circulant matrix [28].

Property 1: For any $n \times n$ circulant matrix, its eigenvectors are the n columns of the DFT matrix.

Consider multiplying circulant matrix C_{cir} times w^k. For simplicity, use the index of the matrix from 0 to $N-1$. For the *l*th component in $(C_{cir}w^k)_l$

$$(C_{cir}w^k)_l = \sum_{j=0}^{N-1} c_{j-l}\omega^{jk} = W_N^{lk}\sum_{j=0}^{N-1} c_{j-l}\omega^{(j-l)k} \tag{4-12}$$

Note that the remaining sum is now independent of l because both c_j and ω^j are periodic in j with period N. Thus, Equation (4-12) can be written as

$$Cw^k = \lambda_k w^k$$

$$\lambda_k = \sum_{j=0}^{N-1} c_{j-l}\omega^{(j-l)k} \tag{4-13}$$

Therefore, the eigenvectors of a circulant matrix form a DFT matrix.

Property 2: The product of the DFT matrix with the first column of a circulant matrix gives a column vector containing its eigenvalues.

From Equation (4-12), by arranging its order, we can see that this equation is actually the DFT matrix times the first column of C_{cir} if one notices the periodicity in its column vector.

4.6 The Circulant Filter is a Steady-State Filter

Using Equation (1-5), a linear causal filter, like Butterworth filter H, can always be expressed in matrix form (adjusting for the time delay through the system as needed)

$$Y = HU + Ox_0 \tag{4-14}$$

where $Y = [y_0, y_1, \dots, y_{N-1}]^T$ is the vector of outputs, U is the vector of inputs, O is the observable matrix of dimension N, and x_0 is the initial condition. Then H is a Toeplitz matrix of the Butterworth pulse response coefficients

$$H = \begin{bmatrix} h_0 & 0 & 0 & \cdots & 0 \\ h_1 & h_0 & 0 & \cdots & 0 \\ h_2 & h_1 & h_0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{N-1} & h_{N-2} & h_{N-3} & \cdots & h_0 \end{bmatrix} \tag{4-15}$$

In order to distinguish the closed-loop feedback system state-space equation from the state-space equation of the filter, one writes the equation of the filter as

$$\begin{cases} x(k+1) = A_f x(k) + B_f u(k) \\ y(k) = C_f x(k) + D_f u(k) \;\; k = 0, 1, 2, \dots N-1 \end{cases} \tag{4-16}$$

Then

$$h_i = \begin{cases} D_f, & i = 0 \\ C_f A_f^{i-1} B_f, & i \geq 1 \end{cases}$$
$$O = \left[C_f, C_f B_f, \dots, C_f B_f^{N-1}\right]^T \tag{4-17}$$

Now write Equation (4-15) in its circulant matrix form as

$$\widehat{H} = \begin{bmatrix} h_0 & h_{N-1} & h_{N-2} & \cdots & h_1 \\ h_1 & h_0 & h_{N-3} & \cdots & h_2 \\ h_2 & h_1 & h_0 & \cdots & h_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{N-1} & h_{N-2} & h_{N-3} & \cdots & h_0 \end{bmatrix} \tag{4-18}$$

The corresponding filter is the circulant Butterworth filter, which is non-causal and is expressed as

$$\hat{Y} = \widehat{H}U + Ox_0 \tag{4-19}$$

Now we prove that $\widehat{H}$ gives the steady-state response of the filter. Recall Property 1 that any circulant matrix can be diagonalized by invertible matrix W, which creates the discrete Fourier transform. Then $\widehat{H}$ is expressed as

$$\widehat{H} = W^{-1}\Lambda W \tag{4-20}$$

From Property 2, the eigenvalues are equal to the DFT matrix times the first column of the circulant matrix, which here is $\widehat{H}$. For $\widehat{H}$, the first column is the unit pulse response of the filter so that the eigenvalues of its circulant matrix $\widehat{H}$ is the discrete Fourier transform of the unit impulse response of the filter

$$\text{col}\ (\Lambda) = W * [h_0, h_1, h_2, \ldots h_{N-1}]^T \tag{4-21}$$

where col (Λ) is a column vector of the eigenvalues along the diagonal of Λ. The frequency response is defined as the discrete-time Fourier transform of the unit pulse response, and the discrete Fourier transform is the sampled frequency version of the discrete-time Fourier transform with discrete frequencies $2\pi k/N$, k from zero to $N - 1$.

Thus, Equation (4-19) now becomes

$$Y = \widehat{H}U = W^{-1}\Lambda(WU) \tag{4-22}$$

where, first the input is converted by the discrete Fourier transform, WU, then multiplied by the frequency response of the discrete filter at the frequencies that can be observed in N steps, and then the inverse transform is taken to go back to the time domains. So $Y = \widehat{H}U$ gives the steady-state response of the filter.

4.7 Comparison Between a Toeplitz matrix of a Filter and the Circulant Matrix of a Filter

Let us closely compare, eg. two Butterworth filters, expressed in Equation (4-14) and Equation (4-22). Both filter matrices use the length N pulse response of the same Butterworth filterer. Equation (4-14) represents an input and output relationship of a classic causal low-pass Butterworth filter, with H its Toeplitz matrix. The transients in the output Y appear in two parts: the initial condition expressed in Ox_0, and the particular solution associated with zero initial conditions as contained in H. Thus, when we use Equation (4-14), even when we set the initial condition term to zero, one cannot conclude that all transients have been removed from the response.

Now consider Equation (4-19) using the circulant matrix, and gives us the input-output relationship of a Circulant Filter $\widehat{H}$. By setting the initial condition to zero, it removes the

transients from the initial condition and Equation (4-19) becomes Equation (4-22). As proved in the previous section, Equation (4-22) gives the steady-state response of a filter having same length *N* pulse response as Butterworth. Transients are eliminated.

4.8 The Optimal Initial Conditions for a Zero-Phase Circulant Filter

Suppose a classic infinite impulse response (IIR) Butterworth filter is expressed in matrix form H, and we create the corresponding circulant Butterworth filter $\widehat{H}$. As suggested by Equation (4-22), the initial condition for the circulant Butterworth filter is zero when one needs a steady-state response. Thus, when making this non-causal circulant Butterworth filter a zero-phase filter, intuitively, one would think the corresponding optimal initial condition should be a zero-initial condition as well. Now we verify this claim using the optimization method stated in the previous section.

Suppose that the circulant Butterworth filter $\widehat{H}$, e.g., derived from the low-pass Butterworth filter H, is expressed as Equation (4-19) with the initial condition to be decided. For both backward and forward filter, we use the same Circulant Filter. Therefore, the corresponding zero-phase filter will have output

$$Y = \widehat{H}^T\widehat{H}U + \widehat{H}^T O x_0 + O^R x_{N-1} \tag{4-23}$$

Recall Equation (4-6), where the optimal initial condition for the zero-phase filter is determined. Replace the filter matrix *H* by its corresponding circulant matrix $\widehat{H}$, thus Equation (4-6) becomes

$$\begin{pmatrix} x_0^{opt} \\ x_{N-1}^{opt} \end{pmatrix} = \left[\left(\widehat{H}^T - I\right)O, \quad \left(I - \widehat{H}\right)O\right]^{+} (\widehat{H}^T\widehat{H} - \widehat{H}\widehat{H}^T)U \tag{4-24}$$

Denote the entries in $\widehat{H}$ as h_{ij}, then the ij^{th} components in the parenthesis of Equation (4-21) can be expressed as

$$\left(\widehat{H}^T\widehat{H} - \widehat{H}\widehat{H}^T\right)_{ij} = \sum_{k=0}^{N-1}[(h_{ik})^T h_{kj} - h_{ik}\left(h_{kj}\right)^T] = \sum_{k=0}^{N-1}[h_{ki}h_{kj} - h_{ik}h_{jk}] = 0 \quad (4\text{-}25)$$

One of the properties of the circulant matrix is that each row and column only contains the same entries, but in a different order, by arranging it, Equation (4-25) is actually zero. Thus, the optimal condition for the circulant matrix zero-phase filter is zero initial condition for both x_0 and x_{N-1}.

$$x_0^{opt} = x_{N-1}^{opt} = 0 \quad (4\text{-}26)$$

Thus, the corresponding input-output relationship of a zero-phase Circulant Filter is given by

$$Y = \widehat{H}^T\widehat{H}U \quad (4\text{-}27)$$

The zero-phase Circulant Filter will have the form of $\widehat{H}^T\widehat{H}$.

Equation (4-7) gives a zero-phase filter using a typical Butterworth filter, and Equation (4-27) gives the zero-phase filter using the circulant Butterworth filter. Filtfilt command in Equation (4-7) has transients buried in all its three terms $H^T U, Ox_0^{opt}$ and $O^R x_{N-1}^{opt}$. Equation (4-27) using the circulant Butterworth filter has no such transients. First, note that Equation (4-27) removes the transient effect from the initial conditions x_0 and x_{N-1} since both terms are zero. Second, $\widehat{H}$ gives the steady-state response of the filter and $\widehat{H}$ has no transients in it. The drawback of using a Circulant Filter is that it is not a causal filter, and needs future inputs. But this is not a problem for ILC since it is a batch process and we know the input to the filter before each run.

In summary, a zero-phase circulant Butterworth filter, has two advantages over the typical MATLAB Filtfilt using Butterworth filter. First of all, the zero-phase circulant Butterworth filter gives the steady-state response of the filter and it eliminates the effects of the transients. The MATLAB Filtfilt, on the other hand, is only reducing the effects of the transients

instead of eliminating it. Second, the zero-phase circulant Butterworth filter has no need to compute initial conditions, but MATLAB Filtfilt needs to compute for the optimal initial conditions. Moreover, one can then create the zero-phase version using this matrix $\widehat{H}$ instead of the original Toeplitz matrix *H*. This approach addresses one aspect of the mismatch between steady-state frequency response and the finite time nature in ILC. ILC is a finite-time problem but the filtering is based on frequency response thinking suggesting the system is at the steady-state without transients. In the early application of zero-phase filter used in ILC, this is the fundamental mismatch that we try to address the frequency-based filtering for a finite-time signal. The Circulant Filter addresses this issue, and gives the steady-state response for a finite-time signal.

The Circulant Filter acts on the *N* time-step signals in ILC, assuming that the signals have a period of *N*-steps. The signal decomposed into these frequencies has the same start point and endpoint. If the signal does not have this property, then there will effectively be a step discontinuity in the frequency representation from the end of the signal to the beginning of the signal, producing the Gibb's phenomenon, and causing convergence to the average value at the discontinuity. The next chapter will address this issue in the application of the Circulant Filter in ILC.

4.9 A Step Further - Cliff Filter

The Filtfilt command and the Circulant Filter considered above, both rely on a chosen Butterworth filter (or other similar alternatives). The available causal filters are imperfect in both the passband, transition band, and the stopband: the passband magnitude response is not 1, and the stopband magnitude response is not zero. The imperfect stopband may leave high-frequency components in the command that destabilize the ILC. This section develops the Cliff Filter with

magnitude response is unity in the passband with zero phase, the magnitude response is zero in the stop band, and the phase lag is 0. For the Cliff Filter, forward-backward filtering is not necessary.

4.10 Cliff Filter Formation

Each ILC iteration contains *N* time steps. These can be perfectly represented using Fourier series as a sum of sine and cosine functions. The number of such functions depends on whether *N* is odd or even. To understand, start with the unit circle $z = e^{j\omega}$ and divide the circle into *N* evenly spaced values, $\omega_k = \frac{2\pi k}{N}, k = 0, 1, 2, \ldots, N-1$, starting with $\omega = 0$, or DC. The DC term only needs a constant term, $k = 1$ and $k = N-1$ produce complex conjugates which together deliver the sine and cosine for the fundamental frequency for *N* time-step signal, and $k = 2$ and $k = N-2$ produce sine and cosine for the first harmonic, etc. If *N* is odd, then there is one term for $\omega_k, k = 0$, then there are $\frac{N-1}{2}$ frequencies represented by the complex conjugate pairs, $\omega_k = \frac{2\pi k}{N}$ and $\omega_k = \frac{2\pi(N-k)}{N}$. If *N* is even, then there is a *k* for which $\omega_k = \pi$ corresponding to Nyquist frequency, which is fully represented by the cosine and needs no complex conjugate pair.

The Cliff Filter is implemented as follows. (1) Take the Discrete Fourier Transform (DFT) of the *N* steps of data. (2) Set the components to zero for the complex conjugate pairs corresponding to frequencies above the cutoff $\omega_c = \frac{2\pi m}{N}$. (3) Take the inverse DFT to get the filtered signal in the time domain with frequencies above the cutoff deleted.

The ideal filter is

$$H_d\left(e^{j\omega}\right) = \begin{cases} 1 & |\omega| < \omega_c \\ 0 & \omega_c < |\omega| < \pi \end{cases} \tag{4-28}$$

where ω_c is the cutoff frequency. The interval between frequencies that can be seen in the N samples of data is $2\pi/N$. The Cliff Filter makes the magnitude and phase of those frequency samples to be the same as the ideal filter, and leave the frequencies one could not see alone. Take the N samples of an ideal filter $H_d(e^{j\omega})$ at the frequencies $\omega_k = \frac{2\pi k}{N}, k = 0, 1, 2, \ldots, N-1$, and assign them to a vector $H(k)$, where $H(k) = H_d(e^{j\omega_k})$, then the Cliff Filter's impulse response (also the coefficient in z-domain) is expressed as,

$$h(n) = \frac{1}{N}\sum_{k=0}^{N-1} H(k)e^{j2\pi kn/N} \quad n = 0, 1, 2, \ldots, N-1 \tag{4-29}$$

In order to make $h(n)$ a real value, it needs to satisfy the following conditions. When N is an odd number

$$\begin{gathered} H(0)\ is\ real, \\ H(N-k) = H^*(k)\ \ k = 1,2, \ldots, N-1 \end{gathered} \tag{4-30}$$

When N is an even number

$$\begin{gathered} H(0)\ is\ real, H(N/2) = 0 \\ H(N-k) = H^*(k), \quad k = 1,2, \ldots, \frac{N}{2} - 1 \end{gathered} \tag{4-31}$$

To design the Cliff Filter from the N samples of ideal filter, form a diagonal matrix called H_c containing the N samples on its diagonal. Then the Cliff Filter in matrix form is

$$Y = (W^{-1}H_cW)U \tag{4-32}$$

where W is the DFT matrix, $W^{-1}$is the inverse DFT matrix, U is the input to be filtered, and Y is the output or filter result. The Cliff Filter matrix H_{cl} is then

$$H_{cl} = W^{-1}H_cW \tag{4-33}$$

The matrix H_c is a diagonal matrix with indices starting from 0 and progressing to $N-1$

$$H_c = diag(1,1,\cdots,1,0,\cdots,0,1,1,\cdots 1) \tag{4-34}$$

The ones on the diagonal go from index 0 to index $m-1$, and from index $N-m+1$ to index $N-1$. The rest of the entries are all zeros. Thus, in the frequency domain, the corresponding Cliff Filter H_{cl} cutoff is at $\omega_c = \frac{2\pi m}{N}$.

4.11 Cliff Filter Characteristics

The Cliff Filter is derived from the ideal filter at the sampled frequency. It belongs to the type of filters designed by the frequency-sampling method. For a typical frequency sampling filter, one only determines the frequency response of the filter at sampled frequencies ω_k, without considering frequencies in between. Since it only has a finite pulse response, the Cliff Filter is an FIR filter.

For a typical application, where the input sequence length is larger than the length of the FIR filter. It could be a problem when the in-between frequencies are not addressed [29]. For example, for a 100-length FIR, the frequency sampling method could only determine the frequency response at $\omega_k = \frac{2\pi k}{100}\ \text{rad/sample}, k = 0, 1, 2, \ldots 99$. But when the input length is 1000, the DFT of such input can see frequency components as small as $\frac{2\pi}{1000}$ rad/sample. However, the ILC problem is a finite-time problem, and the Cliff Filter addresses all frequencies that can be seen. At every frequency that can be seen in the DFT, there is full control of the frequency response.

4.12 The Cliff Filter is a Special Case of the Circulant Filter

This section shows that the Cliff Filter matrix in Equation (4-33) is a circulant matrix. As above, we consider a size *N* square matrix with index going from 0 to $N-1$.Equation (4-33) can be rewritten as

$$H_{cl} = W^{-1}H_cW = W^{-1}\left(I - \overline{H}_C\right)W = I - W^{-1}\overline{H}_CW \tag{4-35}$$

which $\overline{H}_C$ is a diagonal matrix with ones from the index (m, m) to index $(N - m, N - m)$ on the diagonal, i.e. wherever there is a one/zero on the diagonal of H_c there is a zero/one at the same position on the diagonal of $\overline{H}_C$, and vice versa. Note that the identity matrix is circulant, and that the sum or difference of circulant matrices is circulant. We know that the addition of Circulant Filters gives the Circulant Filter [29]. Hence, proof that the final term in Equation (4-33) is circulant, proves that H_{cl} is circulant.

Matrix W is the DFT matrix and W^{-1} is its inverse. The ijth component in the N by N DFT matrix can be expressed as $W_{ij} = \omega^{ij}$, where $\omega = e^{-\frac{j2\pi}{N}}$ and $i = 0,1,2, \dots, N-1$, $j = 0,1,2, \dots, N-1$. The power of exponential is the product of i and j and we write in this way since this also indicates the location of the entries. For example, ω^{23} is the entry at row 2 and column 3 and its value is ω^6 in the DFT matrix W. The same notation applies for W^{-1}as well. The i,j th component in the inverse DFT matrix can be expressed as $(W^{-1})_{ij} = \frac{1}{N}(\omega^*)^{ij}$, where $\omega^* = e^{\frac{j2\pi}{N}}$ and $i = 1,2, \dots, N-1$, $j = 1,2, \dots, N-1$. Note that ω and ω^* are complex conjugate pairs. We use an asterisk to indicate the complex conjugate. W^{-1} has a coefficient $\frac{1}{N}$ in the front, and this will not affect our proof so we neglect it.

We develop the proof in two stages. First we show that the component in the ith row and jth column is equal to the component in $i + 1$th row and $j + 1$th column for $i = 0, 1,2, \dots, N-2$. This leaves out the entry that moves out of the matrix at the bottom and enters at the top in the next column, treated in the second stage.

Matrix W^{-1} can be written in terms of its column partitions

$$W^{-1} = [w^{*0}, w^{*1}, w^{*2}, \dots w^{*N-1}] \tag{4-36}$$

where w^{*i} is a column vector, the ith column in W^{-1} matrix, $i = 0, 1, 2, \dots, N-1$,

$$w^{*i} = [(\omega^*)^{0i} \quad (\omega^*)^{1i} \quad (\omega^*)^{2i} \quad \dots \quad (\omega^*)^{(N-1)i}]^T \tag{4-37}$$

The DFT matrix W has corresponding row partitions

$$W = col[w^0, w^1, w^2, \dots, w^{N-1}] \tag{4-38}$$

where col indicated to display the entries as a column partitioned vector, and where w^i is a row vector given by the ith row of W, $i = 0, 1, 2, \dots, N-1$. The ith row of the W matrix is,

$$w^i = [\omega^{i0}, \omega^{i1}, \omega^{i2}, \dots, \omega^{i(N-1)}] \tag{4-39}$$

Since $\overline{H}_C$ only has ones and zeros on the diagonal, it only picks columns m to column $N-m$ in W^{-1} and rows m to row $N-m$ in W. Then $W^{-1}\overline{H}_C W$ becomes

$$\begin{aligned} W^{-1}\overline{H_c}W &= \sum_{k=m+1}^{N-m+1} (w^*)^k \, w^k \\ &= \sum_{k=m+1}^{N-m+1} [\,(\omega^*)^{0k}(\omega^*)^{1k}, (\omega^*)^{2k}, \dots, (\omega^*)^{(N-1)k}]^T \\ &\cdot [\omega^{k0}, \omega^{k1}, \omega^{k2}, \dots, \omega\text{^}k(N-1)] \end{aligned} \tag{4-40}$$

The ij components in Equation (4-40), where $i = 0, 1, \dots, N-1, j = 0, 1, 2., \dots N-2$, are

$$(W^{-1}\overline{H}_C W)_{ij} = \sum_{k=m+1}^{N-m+1} (\omega^*)^{ik}\omega^{kj} = \sum_{k=m+1}^{N-m+1} (\omega^*)^{k(i-j)} \tag{4-41}$$

Now study the relationship between the jth column and $j+1$th column of $W^{-1}\overline{H}_C W$

$$\begin{aligned} (W^{-1}\overline{H}_C W)_{ij} &= \sum_{k=m+1}^{N-m+1} (\omega^*)^{k(i-j)} \\ &= \sum_{k=m+1}^{N-m+1} (\omega^*)^{k[(i+1)-(j+1)]} = (W^{-1}\overline{H_c}W)_{i+1,j+1} \end{aligned} \tag{4-42}$$

The second stage of the proof is to show that the last component in the jth column becomes the first component in column $j+1$. When $i = N-1$

$$(W^{-1}\overline{H_c}W)_{N-1,j} = \sum_{k=m+1}^{N-m+1} (\omega^*)^{k(N-1-j)} = \left\{ \sum_{k=m+1}^{N-m+1} (\omega^*)^{k(0-j)} \right\} (\omega^*)^{kN} \qquad (4\text{-}43)$$

Recall that $\omega^* = e^{\frac{j2\pi}{N}}$, and $(\omega^*)^{kN} = 1$, thus Equation (4-43) becomes

$$(W^{-1}\overline{H_c}W)_{N-1,j} = \sum_{k=m+1}^{N-m+1} (\omega^*)^{k(0-j-1)} = (W^{-1}\overline{H_c}W)_{0,j+1} \qquad (4\text{-}44)$$

This is the needed result to conclude that the Cliff Filter is a Circulant Filter.

As a summary, Cliff Filter has the good properties of both ideal filter and Circulant Filter. Compared to the circulant Butterworth filter we presented above, first, Cliff Filter has a perfect passband and a stopband, and it has no phase lag like ideal filter, thus it does not need to use forward-backward filtering technique to make it a zero-phase filter; second, Cliff Filter is also a Circulant Filter, it gives the steady-state response of the filter and it does not need to compute for the initial condition as does the MATLAB Filtfilt.

4.13 Numerical Simulation

This section compares the outputs of the Filtfilt based on Butterworth filter H, the circulant zero-phase Butterworth filter, and the Cliff Filter. The filter details are:

(1) Filtfilt and the circulant filter use a 5th order low-pass Butterworth filter with a cutoff at 15Hz and sampling frequency 100Hz. The Cliff Filter also uses a 15Hz cutoff.

(2) Filtfilt uses the matrix form in Equation (4-13), with output based on Equation (4-7).

(3) The circulant zero-phase Butterworth filter uses Equation (4-18), with output based on Equation (4-26).

(4) The output of the Cliff Filter is from Equation (4-31) where H_c has the value 15 for m, which corresponds to a 15Hz cut off, as shown in Equation (4-34).

(5) Three cosine test signals are uses: 5Hz, 10Hz, and 20Hz. The first two are in the passband where the ideal response would be identical to the input, and the third is above the cutoff, and the ideal response would be zero.

(6) Test sinusoids are also considered at 5.1, 10.1, and 20.1Hz to see the behavior of Circulant Filter if the input signal has a frequency not seen by DFT.

The major difference between the Filtfilt and the zero-phase Circulant Filter is that the former calculates initial conditions, and the latter is guaranteed steady-state frequency response. The Butterworth settling time is about 0.16s, so after the 17^{th} time step the response is close to steady state. Figures 4-1 to 4-6 show the first 10 time steps, and the last 10 time steps of the output for the input sinusoids for different frequencies to focus on where the differences are most obvious, Figures 4-1 and 4-2 for 5Hz, Figures 4-3 and 4-4 for 10Hz, and Figures 4-5 and 4-6 for 20Hz. The input cosine, i.e. the desired output, is shown as a dashed line for Figures 4-1 through 4-4, and desired output, which is zero, is shown as the dashed line in Figures 4-5 and 4-6. To the resolution of the figures, the zero-phase circulant and the Cliff Filter are close to the dashed line in each case, but the Filtfilt result deviates significantly, with the deviation significantly larger for higher frequencies.

Figures 4-7 to Figure 4-12 show corresponding plots when the input is a cosine of 5.1Hz, 10.1Hz, and 20.1Hz which are not among the discrete frequencies for which the filters were designed. This time the circulant and the Cliff Filter show some deviation, but the Filtfilt still has the largest error.

Because the results appear similar after the first 10 steps and before the last 10 steps, Table 4-1 gives a more detailed analysis of the differences at all time steps. The Root Mean Square (RMS) of the error is given, where the error is based on error between the test filter and the ideal filter. Consider 5, 10 and 20Hz. The first column of results is for the 5^{th} order zero-phase Butterworth, the basis for the Filtfilt and the zero-phase circulant. Observe that the Filtfilt RMS error is larger than the Butterworth because it has transients as initial conditions, while the circulant is precisely the same as the steady state Butterworth, indicating that the circulant, as discussed earlier, gives the steady-state Butterworth response. On the other hand, the Cliff Filter column show essentially perfect RMS error levels that are numerical zeros, 10^{-16}, 10^{-15} and 10^{-15}. The results of 5.1, 10.1, and 20.1Hz are also given in the table, and the Filtfilt, circulant, and Cliff Filters all have similar error levels, except for Filtfilt at 20.1Hz has somewhat larger error than the others. These results suggest that the Cliff Filter should be the preferred cutoff filter in ILC.

Since these results are all simulation results, it is hard to tell what happens between samples and how stable each of the outputs are in between samples. A potential experimental result shows how the output behaves between samples.

4.14 Conclusion

This chapter investigates several candidates for zero-phase low-pass filters for use in ILC to create stability and robustness to unmodeled high frequency dynamics. The default choice is perhaps the Filtfilt command in MATLB. For ILC one prefers to apply this to Butterworth filters because they do not exceed unity gain in the passband, allowing a higher cutoff. This filters the signal forward, and then filters the result backward (or vice versa) to cancel the phase change. Both forward and backward filters need initial conditions, and this introduces transients at the

start and end of the filtered result, unrelated to the frequency cutoff purpose of the filter. Filtfilt picks these initial conditions, currently it aims to make the forward and backward results as similar as possible. An earlier version used a different method, and Plotnik and Longman demonstrated that the initial conditions chosen could make the ILC iterations unstable.

Two alternatives to the Filtfilt approach are presented here, each of which does not ask for initial conditions. The first approach uses the convolution sum solution of a low pass filter, choosing a Butterworth filter, packaged as a Toeplitz matrix of its Markov parameters. Then the matrix is modified to be a circulant matrix, which is shown to give the steady state frequency response. We call such a filter a circulant Butterworth filter if we use Butterworth filter as a base. When converted to zero phase by the forward-backward approach to cancel phase, it is shown that the optimal initial conditions are zero. The issue of picking initial conditions is avoided. And the serious issue of transients is avoided, because the filter produces steady state behavior directly, and this is the intension when designing a cutoff filter.

The second approach simply asks for an ideal filter, we term it a Cliff Filter. The Butterworth fails to maintain perfect unity gain up to the cutoff, may require a transition zone, and simply decays with increasing frequency above this. The ideal filter has zero phase and unity gain from DC to the cutoff frequency, and zero gain above the cutoff frequency. In the finite time ILC problem, any signal can be represented by a Fourier series containing a finite number of frequencies based on the number of time steps in the data. By making a filter that computes these components without gain or phase distortion up to the cutoff, and eliminates the remaining components above the cutoff, one creates the Cliff Filter. We prove that the Cliff Filter is also a Circulant Filter, and compared to the typical zero-phase Circulant Filter, the Cliff Filter not only

gives the perfect cutoff with no phase lag and it does not need to compute for initial conditions which forward-backward filtering technique requires.

When applying a cutoff filter in hardware implementations of ILC, the most common objective is to create ILC that learns to as high a frequency as possible. The cutoff is used to prevent unmodeled high frequency dynamics (residual modes, parasitic poles) from destabilizing the learning process. Since one usually does not know what is wrong with one's model, the cutoff is tuned in hardware. The usual slow growth of the instability makes this feasible.

Table 4-1. RMS Error between Output from Three Zero-phase Filters and the Desired Output, and between Output from Zero-phase Butterworth Filter at Steady-state and the Desired Output

Frequency	Zero-phase Butterworth Filter at Steady-State	Filtfilt	Circulant Filter	Cliff Filter
5Hz	5.96×10^{-6}	5.22×10^{-3}	5.96×10^{-6}	6.34×10^{-16}
10Hz	7.78×10^{-3}	3.13×10^{-2}	7.78×10^{-3}	1.07×10^{-15}
20Hz	1.98×10^{-2}	1.19×10^{-1}	1.98×10^{-2}	1.81×10^{-15}
5.1Hz	7.35×10^{-6}	5.62×10^{-3}	8.06×10^{-3}	8.34×10^{-3}
10.1Hz	8.68×10^{-3}	3.56×10^{-2}	1.87×10^{-2}	1.70×10^{-2}
20.1Hz	1.86×10^{-2}	1.45×10^{-1}	3.94×10^{-2}	3.79×10^{-2}

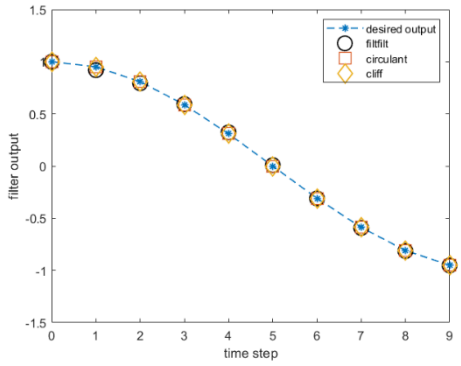

Figure 4-1. First 10 time-step output of three zero-phase filters with 5Hz pure sinusoid inputs

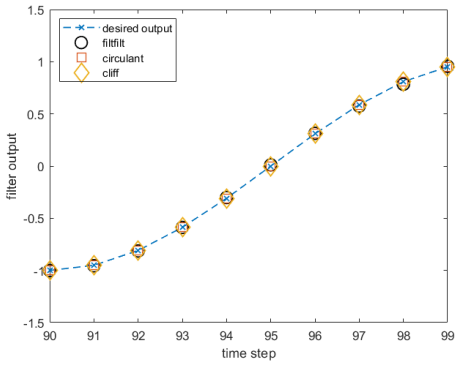

Figure 4-2. Last 10 time-step output of three zero-phase filters with 5Hz pure sinusoid inputs

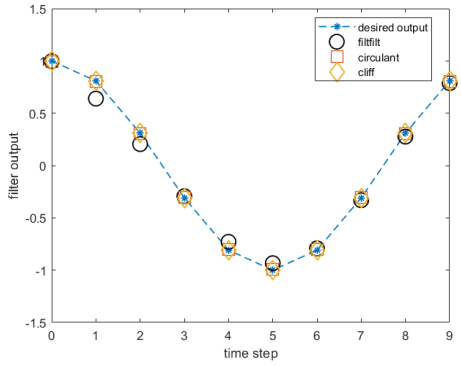

Figure 4-3. First 10 time-step output of three zero-phase filters with 10Hz pure sinusoid inputs

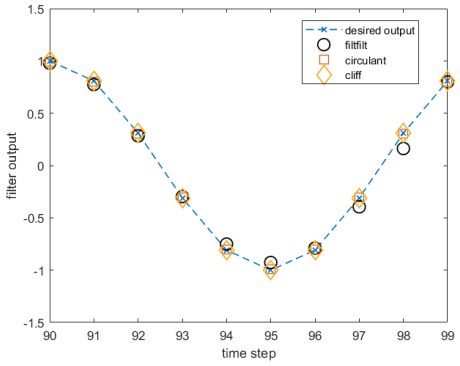

Figure 4-4. Last 10 time-step output of three zero-phase filters with 10 Hz pure sinusoid inputs

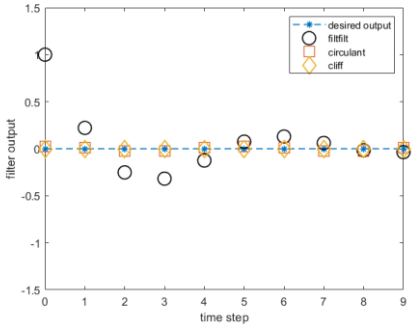

Figure 4-5. First 10-time step output of three zero-phase filters with 20Hz pure sinusoid inputs

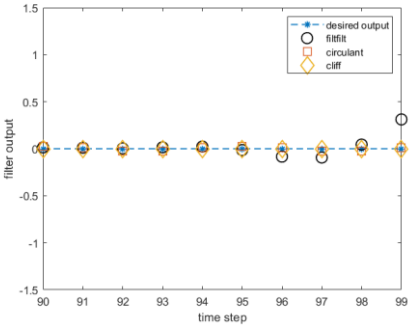

Figure 4-6. Last 10-time step output of three zero-phase filters with 20Hz pure sinusoid inputs

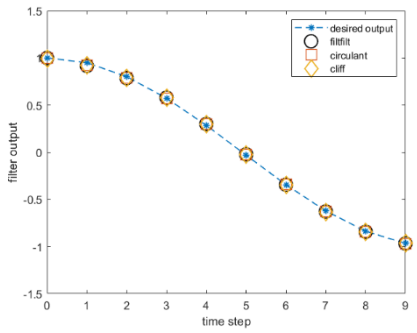

Figure 4-7. First 10-time step output of three zero-phase filters with 5.1Hz pure sinusoid inputs

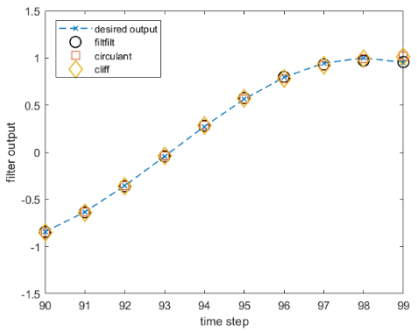

Figure 4-8. Last 10-time step output of three zero-phase filters with 5.1Hz pure sinusoid inputs

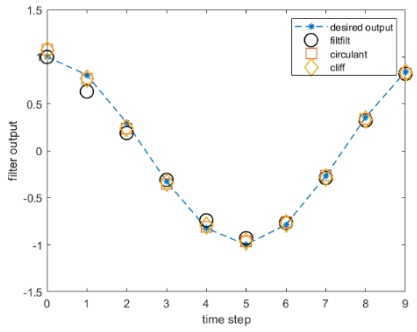

Figure 4-9. First 10-time step output of three zero-phase filters with 10.1Hz pure sinusoid inputs

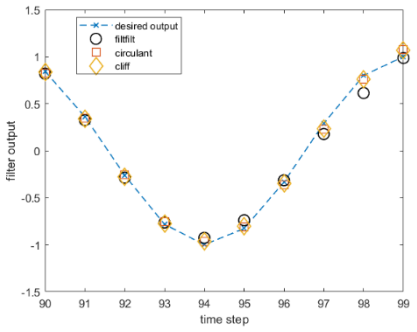

Figure 4-10. Last 10-time step output of three zero-phase filters with 10.1Hz pure sinusoid inputs

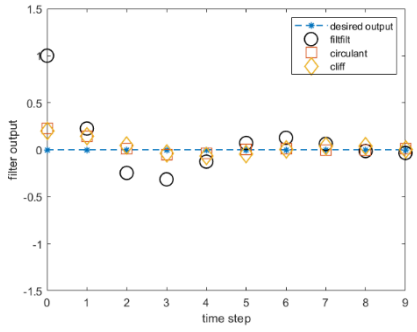

Figure 4-11. First 10 time-step output of three zero-phase filters with 20.1Hz pure sinusoid inputs

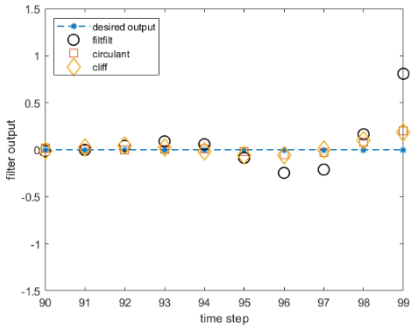

Figure 4-12. Last 10-time step output of three zero-phase filters with 20.1Hz pure sinusoid inputs

Chapter 5: Designing Steady-State Filter for the Finite-Time Signal in Iterative Learning Control

The previous chapter discussed the need of zero-phase low-pass filter to address the stability and robustness in ILC. A typical choice of such zero-phase low-pass filter is given by the Filtfilt command in MATLAB. It creates a mismatch in the ILC design process, the filter is designed based on the frequency thinking, which steady state frequency response, but it is used to finite-time signals for iteration in ILC. The previous chapter addressed this mismatch introducing two steady-state filters, the Circulant Filter and the Cliff Filter, for the finite-time signal. Both eliminate the transients produced by the typical filters. But, both filters present issues of the frequency leakage and the Gibbs phenomenon. The frequency leakage appears when the signal is not one of the discrete frequencies that one can see in the number of time steps in each signal.. The Gibbs phenomenon appears if the signal's start and end points are not equal, which is nearly always the case during ILC iterations. Both will reduce the tracking accuracy and convergence rate of ILC. Two approaches, single reflection and double reflection, are studied in this chapter: One is to do an even reflection about the endpoint of the signal, filter the extended signal, and then use the first half of the resulting signal. The second approach does an odd reflection about the endpoint of the original signal, then does an even reflection of this odd-reflected signal and then uses the first one fourth of the filtered signal. This is done to not only have continuity across the endpoints of the extended signal, eliminating the discontinuity at the endpoints, but to maintain continuity of the first derivative of the signal. A math proof is provided to show that both methods can reduce Gibbs phenomenon, and also provide a formula indicating when it is important to use single/double reflection on signals with different start and

end points. Simulation results show that both single reflection and double reflection can reduce the tracking error of ILC.

5.1 Introduction

As discussed in Chapter 4, ILC needs a zero-phase low-pass filter. A typical candidate of such filter is given by the Filtfilt command in MATLAB. This filter is designed based on desired steady-state frequency response, i.e. the frequency response after all initial condition effects have become negligible. But ILC contains initial condition influence at the start of every run, so there is a mismatch in the modeling. The previous chapter addresses this mismatch introducing steady-state filter for finite-time signal, and this chapter addresses the issues of the frequency leakage and the Gibbs phenomenon in the application of such filters.

5.2 Stability and Robustness Issues in ILC

The stability condition for ILC in the frequency domain is that $|1 - L(z)G(z)| < 1$ be satisfied for all ω up to Nyquist, where $G(z)$ is the system transfer function, $L(z)$ is the transfer function of the ILC law, and $z = \exp(i\omega T)$ with T the sample time interval and ω the radian frequency. It is a necessary and the sufficient condition for ILC stability independent of the number of time steps in the desired trajectory [11]. This condition requires one have a sufficiently accurate model up to the Nyquist frequency so that one can design $L(z)$ to cancel the phase of $G(z)$ of the real world, instead of one's model of the real world, to within -90 and $+90$ degrees when the magnitude of $L(z)G(z)$ is arbitrarily small, and the phase must be within a reduced interval for larger magnitudes. To get some intuition on these limits, consider a compensator $L(z)$ that is just a constant multipling $G(z)$. If the phase of $G(z)$ is -180 degrees, then the absolute value on the left of the convergence condition is clearly larger than one. The ILC law $L(z)$ is needed that aims to cancel the phase of $G(z)$ to prevent this from happening. In

the real world, ILC is a finite-time system that may not enter into steady state, and one uses the stability criteria of the time-domain version. But this frequency thinking explains what the ILC law must do.

The accuracy of the available model usually deteriorates as the frequency increases. One expects that there are missing high frequency dynamics, sometimes described as parasitic poles or residual modes. Confidence in one's model is usually expressed as a function of frequency. One usually needs to introduce a zero-phase low-pass filter $F(z)$ applied to the ILC command the system to increase the robustness of the ILC to the model errors. Then the ILC is prevented from trying to fix tracking errors at high frequencies where the model is uncertain. The stability condition becomes $|F(z)(1 - L(z)G(z))| < 1$ [20], so the cutoff frequency is chosen to attenuate $|1 - L(z)G(z)|$ when it becomes larger than one at high frequencies. The cutoff can be based on one's confidence in the model, but it can also be tuned in hardware based on observed error transformed to the frequency domain. The approach is introduced in experiments on a robot at NASA Langley Research Center [20][25]. A model developed from test data for the command to response of the feedback controllers for each joint, was good up to 18Hz, while Nyquist frequency was 200Hz. Analytically we knew to expect more vibration modes between 18Hz and 200Hz, that were not visible in the data. The resulting final error level after convergence of the ILC was below the reproducibility level of the hardware when evaluated on a day-to-day basis. If the phase of $L(z)G(z)$ with $G(z)$ being the real-world behavior, is outside the error limits described above, then the error grows at these frequencies eventually appearing above the noise level, unless there is a filter cutoff. Reference 30 suggests using this as a technique for experiment design for system identification.

A separate stability issue is commonly produced by the conversion of a continuous time differential equation fed by a zero-order hold input, with output sampled synchronously. Most discrete control systems have this applied to the plant. For reasonable sample time intervals T, perhaps a majority of physical systems will have a zero or zeros introduced in the equivalent plant discrete time transfer function, that are outside the unit circle [17]. ILC is an inverse problem, given the desired output of the discrete control system, converge to a command input to the discrete system to produces it. This converts the zeros into poles, and makes the ILC problem aims to converge to an unstable command needed for zero error. The poles outside are on the negative real axis of the z-domain which corresponds to a growing oscillation at Nyquist frequency. It shows that this instability can be eliminated by the zero-phase low-pass filter discussed above, that is introduced for the different purpose of robustification to high frequency model error [12].

5.3 The Mismatch between ILC and Frequency-based Cutoff and Resulting Issues

The ILC problem is a finite-time tracking problem, but the filter discussed above aims to cut off the learning based on frequency, i.e. it is designed using frequency response thinking considering the system is in steady-state. The ILC system may not enter into the steady state.

In spite of the finite-time natures of ILC, we choose to try to produce the cutoff based on system steady-state frequency response because the robustness considerations are likely based on model confidence as a function of frequency. We need this cutoff since we do not want the control action to contain any frequency component corresponding to signal growth with iteration. In Chapter 4, we discuss several approaches to designing the filter based on steady-state frequency response thinking. Zero phase IIR filters are applied to the signal going forward though the data, and then they are applied in backward time, classified as Filtfilt designs. Each

direction needs initial conditions which create unwanted filter transient at the beginning and the end of the filter result. It has been shown that the filter designed by using the Filtfilt command producing the initial conditions can destabilize the ILC algorithm [24]. In Chapter 4, we discussed the steady-state filter for the finite-time signal which is based on converting signals using Discrete Fourier Transforms (DFT). This eliminates the issue of filter transients. We provide two candidates which are Circulant Filter and Cliff Filter.

Another issue to consider is that every ILC run restarts from a repeating initial condition, and the initial portion of the response then contains transient response not related to frequency response. Our thinking is that whatever the command input is, including this transient, the filter should eliminate any component that might produce excitation of undesired frequencies. Thus, the cutoff filter should remain based on frequency components of the signal.

The use of DFT to decompose the finite-time command input to be filtered, introduces other issues. The DFT of a finite time signal exhibits phenomena referred to as leakage effects. When the endpoint is not the same as the start point in the data set to be filtered, then it is represented by sines and cosines that have the same start point and end point, i.e. of the period of the number of time steps. This results in a finite time version of the Gibbs phenomenon occurring to handle the discontinuity of the signal at the start/end point. Particular attention is paid here to finding ways to have the DFT of the finite-time signal be as little affected as possible by these phenomena, trying to make it be as close as possible to the steady-state frequency response.

5.4 Three DFT-Based Steady-State Filter

Perhaps the earliest use of zero-phase low-pass filtering in ILC is the MATLAB Filtfilt at the time [20][25]. Plotnik and Longman showed that the handling of the initial conditions could

result in unstable ILC, and then introduced the concept of a Cliff Filter [19]. Bing and Longman introduce the concept of the Circulant Filter and use the system identification to indicate that it gives the steady-state frequency response in approximation [18]. In the last chapter, it is proved that the Cliff Filter is a special case of a Circulant Filter, and proved that the Circulant Filter does give the steady-state response, and shows that one does not need to calculate the optimal initial conditions for the Circulant Filter when making it a zero-phase filter by using the forward-backward filtering technique. Juang and Longman generalizes such ideas and introduce the harmonic filters [12].

5.5 Zero-Phase Circulant Filter, and New Proof to Show it Gives the Steady-State Response

Chapter 4 introduces the Circulant Filter. A Circulant Filter starts with the typical Toeplitz matrix P of a filter, e.g. a Butterworth filter, as shown in Equation (1-5). The first column of both filters is the same, which are the length-N filter pulse response history. To form the Circulant Filter, the next column is the result of the previous column shifting downwards by one entry and the last entry move to the top, and similarly for the remaining columns. The Circulant Filter is given below where the hat denotes that this matrix is a circulant matrix.

$$\widehat{H} = \begin{bmatrix} h_0 & h_{N-1} & h_{N-2} & \cdots & h_1 \\ h_1 & h_0 & h_{N-3} & \cdots & h_2 \\ h_2 & h_1 & h_0 & \cdots & h_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{N-1} & h_{N-2} & h_{N-3} & \cdots & h_0 \end{bmatrix} \tag{5-1}$$

Chapter 4 uses the property of circulant matrix to prove that the Circulant Filter gives the steady-state response of the associated filter. Below is a new version of proof, based on the properties of pulse response and DFT, to show that it gives the steady-state response of the filter.

Recall that the system response can be expressed as the linear convolution of its impulse response $h[n]$ and the input $x[n]$

$$y[n] = \sum_{k=0}^{N-1} x[k]h[n-k] \tag{5-2}$$

In matrix form, it can be written as,

$$y[n] = \begin{bmatrix} y[0] \\ y[1] \\ \vdots \\ y[N-1] \end{bmatrix} = [h[n] \quad h[n-1] \quad \dots \quad h[n-N+1]] \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[n-N+1] \end{bmatrix} \tag{5-3}$$

Using the A, B, C, D of the system state-space equation

$$h[n] = \begin{bmatrix} CB \\ CAB \\ \vdots \\ CA^{N-1}B] \end{bmatrix} \; h[n-1] = \begin{bmatrix} 0 \\ CB \\ \vdots \\ CA^{N-2}B] \end{bmatrix} \dots \dots h[n-N+1] = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ CB] \end{bmatrix} \tag{5-4}$$

Equation (5-3) and (5-4) connect Equations (1-4) and (1-5). Recall the N by N DFT matrix W is given by

$$\begin{aligned} W &= \left\{\omega_N^{ij}\right\} \; i,j = 0,1,2,\dots N-1, \\ \omega_N^k &= e^{\frac{-j2\pi k}{N}}, j \; is \; pure \; complex \end{aligned} \tag{5-5}$$

The steady state response of the system at sampled frequencies, denoted as $H[k]$, are given by multiplication of DFT and the impulse response of the system as shown below. Note that, the frequency resolution frequencies observable by the DFT are determined by the signal length N.

$$H[k] = Wh[n] \tag{5-6}$$

Recall the shifting property of DFT, that the DFT of $h[n-m]$ is $\omega_N^{km}h[k]_N$, where subscript N means modulo N. Then

$$\omega_N^k H[k] = Wh[n-1]_N, \omega_N^{2k} H[k] = Wh[n-2]_N,$$
$$\ldots . \omega_N^{k(N-1)} H[k] = Wh[n-N+1]_N \tag{5-7}$$

Combine Equations (5-6) and (5-7) produces

$$\left[H[k], \omega_N^k H[k], \ldots, \omega_N^{k(N-1)} H[k]\right] = [Wh[n], Wh[n-1]_N, \ldots Wh[n-N+1]_N] \tag{5-8}$$

Rewrite Equation (5-7) in matrix form. The right-hand side of the equation is equal to

$$[Wh[n], Wh[n-1]_N, \ldots Wh[n-N+1]_N]$$
$$= W[h[n], h[n-1]_N, \ldots h[n-N+1]_N] = W\widehat{H} \tag{5-9}$$

The left-hand side is equal to

$$\left[H[k], \omega_N^k H[k], \ldots, \omega_N^{k(N-1)} H[k]\right] = diag(H[k])W \tag{5-10}$$

where $diag(H[k]) = diag(H[0], H[1], \ldots, H[N-1])$. From Equations (5-9) and (5-10), one can write

$$diag(H[k]) = W\widehat{H}W^{-1}$$
$$\widehat{H} = W * diag(H[k])W^{-1} \tag{5-11}$$

Thus, $\widehat{H}$ gives us the steady-state frequency response of the filter at sampled frequencies. One can derive a circulant Butterworth filter from the Toeplitz matrix of filter as in Equation (5-1). Then, the corresponding Circulant Filter will give the steady-state response of the Butterworth filter. However, one should notice that such a circulant Butterworth filter is not a zero-phase filter. It can be made into a zero-phase filter by using forward-backward filtering discussed in Chapter 4.

To make a zero-phase filter, one first filters the signal forward in time, producing attenuation above the cutoff, but producing phase lag in the signal at the same time. Then one reverses the time in the output sequence and filters it again. This doubles the attenuation above the cutoff and puts in phase lead to cancel phase lag produced in the forward filtering. Then one

reverses the time in the final output to revert to forward time. We refer to this as forward-backward filtering. One could also use backward-forward filtering to achieve zero-phase, which is to reverse the input sequence first, filter it, and reverse the output and filter it again. Note that there are initial conditions needed in forward filtering, and also initial conditions needed in backward filtering. In Chapter 4, we have proved that when the Circulant Filter is made as the zero-phase filter using forward-backward filtering, its optimal initial conditions are zeros. Thus, the zero-phase Circulant Filter H_c has the form

$$H_c = \widehat{H}^T \widehat{H} \tag{5-12}$$

Thus, the corresponding input-output relationship of a zero-phase Circulant Filter is given by

$$Y = \widehat{H}^T \widehat{H} U \tag{5-13}$$

Note that a typical input-output relation of a low-pass filter can be expressed as in Equation (1-4) with the P matrix defined by Equation (1-5) for the state variable model of the filter dynamics. Note that even if one sets the initial condition to zero, Equation (1-4) does not give the steady-state response of the filter since the Toeplitz matrix P itself contains transients. Chen and Longman show the relationship between steady-state frequency response and the singular value decomposition (SVD) of P [22]. As the length of the filtered signal tends to infinity and the size of matrix P increases to infinity, the singular values of P converge to the steady state magnitude frequency response of the system, and the right and left singular vectors become sinusoids whose phase difference is the steady state phase frequency response. For a finite-length signal, Equation (1-4) does not give steady-state response.

5.6 Weighted Harmonic Filter

Juang and Longman propose a similar idea of the Circulant Filter [12]. Instead of starting by making the Toeplitz matrix of a Butterworth filter into a circulant one, it starts with having

the magnitude frequency response at sampled frequencies of a Butterworth filter, entered on the diagonal of a diagonal matrix M, and then multiplies both sides by a DFT matrix pair, or a real DFT matrix pair, to get a harmonic filter that produces the steady-state response of the filter. Suppose that the frequency responses of a Butterworth filter at sampled frequencies are evenly distributed between 0 and 2π, and the corresponding magnitude frequency response is entered on the diagonal of matrix M. Denote by W the DFT matrix. Then the harmonic Butterworth filter H_h is given by

$$H_h = W^{-1}MW \tag{5-14}$$

H_h is also a circulant matrix. Filter H_h has phase lag in it, and if one wants to have a zero-phase filter, one should use $H_h^T H_h$ instead. One could make a zero-phase Butterworth filter only by using the magnitude response M' of a Butterworth filter, then the new filter $H_h{}'$ will be a zero-phase filter but with the same magnitude response as the Butterworth filer at the sampled frequency

$$H_h{}' = W^{-1}M'W \tag{5-15}$$

5.7 Difference Between Circulant Butterworth Filter and Harmonic Butterworth Filter

We demonstrate the difference between a Circulant Filter and a harmonic filter using the Butterworth filter to illustrate. A discrete Butterworth filter can be either characterized by its difference equation, its impulse response, its *z*-transfer function, and its frequency response. All these expressions can be interchangeable and are equivalent. Suppose a discrete Butterworth filter can be expressed by the following difference equation and pulse response $h\{n\}$. Note that the pulse response of a discrete Butterworth filter is infinite

$$y(k) + a_1 y(k-1) + \cdots + a_n y(k-n) = b_1 u(k-1) + \cdots b_m u(k-m) \tag{5-16}$$

$$h\{n\} = \{h[0], h[1], \dots, h[n], \dots\}$$

The two are connected by the z-transfer function

$$G(z) = \frac{b_1 z^{-1} + b_2 z^{-2} + \cdots + b_m z^{-m}}{a_1 z^{-1} + a_2 z^{-2} + \cdots + a_n z^{-n}} = \sum_{n=0}^{\infty} h[n] z^{-n} \tag{5-17}$$

The corresponding frequency response $G(\omega)$ is a continuous function

$$G(\omega) = G(z)|_{z=e^{-j\omega}} = \sum_{n=0}^{\infty} h[n] e^{-j\omega n} \tag{5-18}$$

This equation can be written in two terms, one is the summation from 0 to $N-1$, and the other term sums from N to infinity

$$G(\omega) = \sum_{n=0}^{N-1} h[n] e^{-j\omega n} + \sum_{n=N}^{\infty} h[n] e^{-j\omega n} \tag{5-19}$$

If we want to know the frequency response at sampled frequency $\omega = \frac{2\pi k}{N}$, write

$$G(\omega)|_{\omega=\frac{2\pi k}{N}} = \sum_{n=0}^{N-1} h[n] e^{-\frac{j2\pi kn}{N}} + \sum_{n=N}^{\infty} h[n] e^{-\frac{j2\pi kn}{N}} \tag{5-20}$$

The first term, on the right side of Equation (5-20) is actually the DFT of the first N term pulse response of the Butterworth filter. Recall that for the circulant Butterworth filter, one knows its eigenvalues by computing the DFT matrix multiplied by its first column, i.e. the first N terms of the Butterworth filter. We also prove that such eigenvalues are the steady-state response of the filter expressed by this circulant matrix. This means that the circulant Butterworth filter gives us steady-state response coming from the first N terms pulse response of the Butterworth filter instead of the infinite pulse response. The harmonic Butterworth filter gives us the steady-state response of the Butterworth filter from the infinite pulse response.

Therefore, we have a new explanation for Equation (5-20). The left-hand side of the equation, gives the steady-state response of a Butterworth filter which is what harmonic Butterworth filter gives to us. The first term on the right-hand side of the equation gives the steady-state response of an FIR filter whose pulse response is the same as the first N terms of the Butterworth filter which is what the circulant Butterworth filter gives us. The remaining is then the error term. This explains the mismatch in Table 1 of Reference 22 where the system identification of the Circulant Filter is always 0.01 smaller than that of the Harmonic Butterworth filter.

The error term is useful since it can estimate the discrepancy between the two filters, and it can indicate how large the filter matrix should be so that the errors are reduced to a chosen threshold level

$$|error| = |\sum_{n=N}^{\infty} h[n]e^{-\frac{j2\pi kn}{N}}| \leq \sum_{n=N}^{\infty} |h[n]| \left|e^{-\frac{j2\pi kn}{N}}\right| = \sum_{n=N}^{\infty} |h[n]| \tag{5-21}$$

For a typical 5th order Butterworth filter, the error term is less than 0.01 when one uses more than 100-terms in the filter pulse response. This means that at sampled frequencies, the magnitude difference between Harmonic Butterworth Filter and Circulant Butterworth Filter is less than 0.01.

5.8 Cliff Filter

Cliff Filter $H_{cliff} = W^{-1}M_cW$, is a special case of weighted harmonic filter, where all the diagonal terms in M are either ones or zeros. Matrix M_c is a diagonal matrix with indices starting from 0 and progressing to $N-1$

$$M_c = diag(1,1,\cdots,1,0,\cdots,0,1,1,\cdots 1) \tag{5-22}$$

Where there are $m+1$ ones for the first set and there are m ones for the second set. Entry 1 applies to DC and entry i and $N-i$ are complex conjugates associated with the same frequency. The ones on the diagonal go from index 1 to index $m+1$, and from index $N-m+1$ to index N. The rest of the entries are all zeros. Then one multiplies both sides by the inverse DFT and DFT matrix to get the Cliff Filter. Thus, in the frequency domain, the corresponding Cliff Filter cutoff is at $\omega_c = \frac{2\pi m}{N}$.

5.9 Gibbs Phenomenon

The Gibbs phenomenon in the Fourier expansion of a continuous-time signal is observed when the signal has a step discontinuity in time. The partial sum of the Fourier series converges to the midpoint of the step discontinuity, and before and after the discontinuity there is overshoot/undershoot whose maximum value is determined by the height of the discontinuity. The Fourier series partial sum converges pointwise at all points before and after the discontinuity, but convergence is not uniform as the points of overshoot / undershoot move while the height remains constant as more terms are included in the series.

Consider the Fourier series expansion of a continuous-time square wave of the magnitude of one with period 2π. Its Fourier series is a sum of sine waves with odd frequencies

$$f(x) = \frac{4}{\pi} \sum_{n=1,3,5\ldots}^{\infty} \frac{1}{n} \sin(nx) \tag{5-23}$$

Also consider the Fourier series of the integral of the square wave which produces a triangle wave of the same period. Its Fourier series is given as

$$f(x) = \frac{\pi}{2} - \frac{4}{\pi} \sum_{n=1,3,5\ldots}^{\infty} \frac{1}{n^2} \cos(nx) \tag{5-24}$$

Figures 5-1 presents the partial sum of the square wave with the amplitude 1 and the period of 2π including only 10 terms in the sum in Equation (5-23), while Figure 5-2 shows the partial sum including 50 terms. The overshoot / undershoot oscillation move closer to the location of the discontinuity as more terms are included, but the maximum amplitude does not decay. Figures 5-3 and 5-4 are plots for a triangular wave of the same period which is for the integral of the square wave as Equation (5-24). The integration removes the step discontinuity. Remaining evidence of the overshoot / undershoot and oscillation are not visible to graphical accuracy, even after using only 10 terms in the sum.

For continuous-time signals, the Fourier series expansion converges pointwise to the original signal in the limit as the number of frequencies included tends to infinity. If one terminates the series prematurely, then the Gibbs phenomenon appears. Analogous behavior occurs when one takes the DFT of a signal with a given number *N* of data points. The DFT can only see a finite number of frequencies, all the frequencies one can observe in the data samples (roughly *N*/2 frequencies, differing based on whether *N* is odd or even). If all frequencies in the DFT are used to make a time domain reconstruction of the signal, then the reconstruction is perfect at every point in the discrete time signal -- analogous to the infinite number of frequencies in the continuous time Fourier series. But if the reconstruction of the time signal uses a smaller number of frequencies, then the Gibbs phenomenon becomes evident. The signal will converge to the midpoint of a step discontinuity, and there will be oscillation behavior before and after the discontinuity.

Both the zero-phase Circulant Filter and the Cliff Filter investigated here, are based on DFT analysis, and cutting out some higher frequencies to produce the desired cutoff. When we use DFT to represent the signal, the math implies that such signal is periodic even if it is a finite-

time signal. Under this periodic assumption, signals with different starting and ending points will have a jump discontinuity. This jump discontinuity contributes to the oscillations of the filtered result. When we use these DFT-based filters to filter a command with different starting and ending points in our ILC, then filtered result will have oscillation at both ends. When the start and end of the signal being filtered are the same, then this phenomenon is no present.

We illustrate the above discussion in Figures 5-5 to 5-9 using three signals represented by 100 evenly distributed samples. The first signal is one fourth of a sine wave in 100 samples, the second signal is one half of a sine wave in 100 samples, and the third has a full period of the sine wave in 100 steps. Each signal can be expressed by its DFT, and the signal in the time domain can be rebuilt by adding all sinusoidal components from DC to Nyquist frequency. In each figure the dashed line shows the original signal and the circles give the signal rebuilt from its DFT. Figures 5-5 and 5-6 examine the ¼ sine wave function and present the time function produced using a partial sum of the DFT result including 11 terms, and 31 terms respectively, i.e. adding frequencies from DC to $\frac{\pi}{5}$ rad/s, and DC to $\frac{3\pi}{5}$ rad/s respectively. The difference between the start point and the endpoint of the one fourth of a sine wave tells us to expect to see a finite time version of the Gibbs phenomenon produced by the discontinuity going from the end of one period of the DFT signals to the start of the next. Large deviations from the dashed line are observed. The continuous time result converges to the midpoint of the discontinuity. In the sampled time result, if one uses interpolation between the starting point and the ending point in Figure 5-5, the result does pass through the midpoint of the discontinuity. Figure 5-7 shows that this finite-time version of the Gibbs phenomenon disappears completely when no frequencies are eliminated from the DFT, and the time function reconstructed – it is guaranteed to pass through all 100 points of the original signal.

For the ½ sine wave, Figure 5-8 shows the 11-term partial sum of its DFT. The oscillation behavior at the end points is much less than the corresponding result for the ¼ sine wave where there is an implied jump discontinuity. As one increases the frequencies included, it will quickly converge to the original signal. One should notice that the implied periodicity suggests that this ½ sine wave is periodic without a discontinuity, but will have a cusp at the end, i.e. a step discontinuity of the first derivative. Figure 5-9 shows the 31-term partial sum of its DFT. With continuity and first derivative continuity maintained, no Gibbs phenomenon and the convergence rate is faster.

For the full cycle sine wave, due to the property of DFT, one needs to add only one frequency sinusoid and the signal is rebuilt. Notice that the full cycle sine wave at both its first and last points, it has first derivative continuity. This suggests that if our signal has not only continuity but also the first-derivative continuity, the oscillation because of the frequency cutoff will disappear much faster.

5.10 Improving the Frequency Response Representation of the Signal for Steady-State Response Filters

The Cliff Filter aims for a perfect cutoff at the chosen cutoff frequency. This would allow the ILC law to have zero tracking error up to the highest cutoff frequency possible in the presence of high frequency model error. The Cliff Filter has zero magnitude in the stopband which is desirable in ILC since an imperfect filter magnitude response decay with frequency in the stopband contains some frequencies above the cutoff that might still be able to trigger the instability of ILC because of the model error above the cutoff.

The Cliff Filter will exhibit a sampled time version of the Gibbs phenomenon. Figure 5-10 illustrates this effect. The input signal is a 5^{th} order polynomial from 0 to 1 sampling at 100Hz

with a signal length of 100. The polynomial satisfies boundary conditions of zero and zero slope at the start and zero slope when reaching the endpoint at 1. The solid line in Figure 5-10 gives this curve. The circles are the result of applying the Cliff Filter to 100-time steps samples of the signal, using a 15Hz cutoff, when Nyquist frequency is 50Hz. Since the start point and end point of the trajectory are not equal, there is an implied jump discontinuity producing the Gibbs phenomenon. Eliminating the high frequency components above the cutoff in this finite time signal results in the oscillation with substantial deviation from the original polynomial history at both ends of the trajectory.

To address this, we seek a method to make the jump discontinuity disappear making a signal that can be expressed purely in terms of sinusoids of the period of the number of time steps. An intuitive solution to this problem would be a single reflection. The original signal of length N is reflected about its end point to create a signal of length $2N$ steps, having the start point and the end point the same as shown in Figure 5-11. It illustrates the single reflection of the same 5^{th} order polynomial from 0 to 1 in 100-time step. The red portion of the curve shows the original signal and the red portion plus the black portion shows the single-reflected signal. Then one filters this single-reflected signal and only uses the first half of the filter result.

The 5^{th} order polynomial with zero slope at the end, makes a smooth function at the end point when reflected. Picking a different desired trajectory that has a non-zero slope at the end, when reflected will have a cusp at the end, and the discontinuity of the first-derivative across the end point might produce undesirable behavior of the filter result. We address this by considering a double reflection.

For the same length N signal, extend the trajectory with an odd reflection about the end point of the original signal, to create a signal of length $2N - 1$. Then do a further extension that

is an even reflection of the previously reflected signal to create the signal of length $4N-2$. Figure 12 illustrates this or the 5th order polynomial desired trajectory. The red curve is the original 5th order polynomial signal from 0 to 1 in 100-time step. The black curve shows the result of the double reflection, guaranteeing that at the end point the signal is continuous with continuous first derivative, and that the full signal returns to the start point. Note that the starting point of the 5th order polynomial is also continuous with its first derivative. This is a good property for the chosen desired trajectory to have so that the ILC does not have to work hard to suddenly get from the initial conditions onto the desired trajectory initial slope in one time step. In the 5th order polynomial case, the continuity of the zero initial condition and zero initial slope matches the final value and slope after the double reflection.

5.11 Math of Single Reflection Method

Suppose the input to the plant at iteration $j+1$ is $\underline{u}_{j+1}$ with length N steps. Then extend the signal by an even reflection of $\underline{u}_j + L\underline{e}_j$ about the end point, creating the new signal of length $2N$. Apply the cutoff filter to this $2N$ signal, and only use the first half of the result to form $\underline{u}_{j+1}$. Below we present a mathematical analysis to study the behavior of this approach.

Before the input reflection, the command at iteration j to the plant is

$$\underline{u}_{j+1} = F(\underline{u}_j + L\underline{e}_j) \tag{5-25}$$

After the input reflection, the signal after filtering becomes

$$\begin{bmatrix} \underline{u}_j + L\underline{e}_j \\ \underline{u}^R{}_j + L^R\underline{e}_j \end{bmatrix} \tag{5-26}$$

where R is the row reversing operator. Denote by F^* the filter but now the size is $2N$ by $2N$. Divide F^* in four blocks each of size N by N such that $F^* = \begin{bmatrix} f_{11} & f_{12} \\ f_{21} & f_{22} \end{bmatrix}$. Since we only use the first half of the filtered single-reflected signal, the command at iteration j to the plant is

$$\underline{u}_{j+1} = [f_{11} \quad f_{12}] \begin{bmatrix} \underline{u}_j + L\underline{e}_j \\ \underline{u^R}_j + L^R \underline{e}_j \end{bmatrix} \tag{5-27}$$

Recall the equation $\underline{e}_j = -Pu_j + f$, and $AB^R = A^c B$, where R and C are row operator and column operator respectively to reverse the order of rows and columns.

$$\underline{u}_{j+1} = (f_{11} + f_{12}^c - f_{11}LP - f_{12}LP)\underline{u}_j + (f_{11}L + f_{12}^c L)f \tag{5-28}$$

At steady state

$$\underline{u}_\infty = [I - (f_{11} + f_{12}^c)(I - LP)]^{-1}(f_{11} + f_{12}^c)Lf \tag{5-29}$$

The only difference between Equation (1-13) and Equation (5-29) is that you change F to $f_{11} + f_{12}^c$

5.12 Math of Double Reflection Method

The single reflection approach does not consider the potential effects of a discontinuity in derivatives of the signal, e.g. although the step discontinuity is gone in Equation (5-24), there is still a cusp in Figures 5-3 and 5-4. The double reflection extends the trajectory in a way that obtain continuity of the first derivative. For the simplicity of math, we will create a signal of $4N$ compared to one of $4N - 2$ in the Section 5.10. When the N is large, there would not be much difference between the two. We do odd reflection at the end but doubling the end point to have the signal of length $2N$, and do even reflection of this $2N$ signal creating a new signal of length $4N$. Then we filter it, but only use the first N-time steps as our input to the plant. After the input reflection, the signal after the filter becomes

$$\begin{bmatrix} \underline{u}_j + L\underline{e}_j \\ 2U - \underline{u^R}_j - L^R \underline{e}_j \\ 2U - \underline{u}_j - L\underline{e}_j \\ \underline{u^R}_j + L^R \underline{e}_j \end{bmatrix} \tag{5-30}$$

where R is again the row reversing operator, and U is the end point of the unreflect signal. This time define F^* as the filter but now the size is $4N$ by $4N$. Partition F^* into 16 blocks with each block of size N by N such that that $F^* = \begin{bmatrix} f_{11} & \cdots & f_{14} \\ \vdots & \ddots & \vdots \\ f_{41} & \cdots & f_{44} \end{bmatrix}$. Since we only use the first half of the filtered double-reflected signal, the command at iteration j to the plant can be written as,

$$\underline{u}_{j+1} = [f_{11} \quad f_{12} \quad f_{13} \quad f_{14}] \begin{bmatrix} \underline{u}_j + L\underline{e}_j \\ 2U - \underline{u}^R{}_j - L^R\underline{e}_j \\ 2U - \underline{u}_j - L\underline{e}_j \\ \underline{u}^R{}_j + L^R\underline{e}_j \end{bmatrix} \tag{5-31}$$

Then,

$$\underline{u}_{j+1} = (f_{11} - f_{12}^c - f_{13} + f_{14}^c)(I - LP)\underline{u}_j + (f_{11} - f_{12}^c - f_{13} + f_{14}^c)Lf + 2(f_{12} + f_{13})U \tag{5-32}$$

Since U is changing every iteration, we do not have an equation to express the command to the system after the learning process is finished as Equation (5-29) for the double reflection method.

5.13 Single Reflection Reduces Gibbs Phenomenon

This section studies conditions under which the single reflection method reduces the Gibbs phenomenon. Denote a length-N input signal as $u[n]$, and its DFT as

$$U[k] = \sum_{n=0}^{N-1} u[n]e^{-\frac{j2\pi kn}{N}} \tag{5-32}$$

The DFT of the reflected signal $R[k]$ is

$$R[k] = \sum_{n=0}^{N-1} u[n]e^{-\frac{j\pi kn}{N}} + \sum_{n=N}^{2N-1} u[2N-1-n]e^{-\frac{j\pi kn}{N}} \tag{5-34}$$

Make a change of variables $p = 2N - 1 - n$, and change variables from p to n again

$$R[k] = \sum_{n=0}^{N-1} u[n] e^{-\frac{j\pi kn}{N}} + e^{\frac{j\pi k}{N}} \sum_{n=0}^{N-1} u[n] e^{\frac{j\pi kn}{N}} \tag{5-35}$$

Given an input signal $u[n]$ of length N, then the reflected signal $r[n]$ is of length $2N$, which is an even number. At Nyquist frequency $k = N$, one can easily prove that the reflected signal DFT $R[N] = 0$.

$$R[N] = \sum_{n=0}^{N-1} u[n]\{(co\,s(\pi n) + cos(\pi n + \pi)\} = 0 \tag{5-36}$$

Consider a high frequency range $k = N - m$, where m is a positive integer, and $m \ll N$. In ILC the number N can easily be a large number. Since $m \ll N$, then we can think of $\frac{\pi m}{N} \approx 0$, and the DFT of the reflected signal $r[n]$ at high frequencies is

$$\begin{aligned} R[N-m] &= \sum_{n=0}^{N-1} u[n]\{e^{-jn\left(\pi - \frac{\pi m}{N}\right)} + e^{-j(n+1)\left(\pi - \frac{\pi m}{N}\right)}\} \\ &\approx \sum_{n=0}^{N-1} u[n]\left\{e^{-jn\pi} + e^{-j(n+1)\pi}\right\} = 0 \end{aligned} \tag{5-37}$$

We conclude that a single reflection method will make the reflected signal's DFT to be roughly zero in the high frequency range.

Now examine the DFT of the initial signal $x[n]$ of length N in the same range. When N is an even number, its Nyquist frequency corresponds to $k = N/2$, and one can show that

$$U\left[\frac{N}{2}\right] = \sum_{n=0}^{N-1} u[n] e^{-j\pi n} = \sum_{n=0}^{N-1} (-1)^n u[n] \tag{5-38}$$

For a high frequency range $k = \frac{N}{2} - m$, where m is a positive integer, and $m \ll N$, since $\frac{2\pi m}{N} \approx 0$, then the DFT simplifies

$$U\left[\frac{N}{2}-m\right]=\sum_{n=0}^{N-1}u[n]e^{-jn(\pi-\frac{2\pi m}{N})}\approx\sum_{n=0}^{N-1}u[n]e^{-j\pi n}=\sum_{n=0}^{N-1}(-1)^n u[n] \quad (5\text{-}39)$$

When N is an odd number, $X[k]$ does not sample at Nyquist frequency, but we still can denote its high frequency range as $k=\frac{N-1}{2}-m$, where m is a positive integer, and $m \ll N$. The DFT in this case still can be simplified since $\frac{2\pi(m+\frac{1}{2})}{N}\approx 0$

$$U\left[\frac{N-1}{2}-m\right]=\sum_{n=0}^{N-1}u[n]e^{-jn(\pi-\frac{2\pi(m+\frac{1}{2})}{N})}\approx\sum_{n=0}^{N-1}u[n]e^{-j\pi n}=\sum_{n=0}^{N-1}(-1)^n u[n] \quad (5\text{-}40)$$

Now we can see that the reflected signal $r[n]$ has a DFT roughly equal to zero in the high frequency range but the counterpart of original signal $x[n]$ is roughly equal to $\sum_{n=0}^{N-1}(-1)^n x[n]$.

Statement 1. If the input signal $x[n]$ has the property that the magnitude of this summation $\sum_{n=0}^{N-1}|(-1)^n x[n]| \gg 0$, then the single reflection will reduce the Gibbs phenomenon.

The suggests that we can make three claims. First, if the signal $x[n]$ is symmetric or close to symmetric about its mid point, there is no need for making a single reflection. For this case, the Gibbs phenomenon is not present or it can be negligible. Second, if the signal $x[n]$ is increasing or decreasing, or its increasing portion is significantly larger than its decreasing portion or vice versa, then the single reflection will help reduce the Gibbs phenomenon. Third, if the signal $x[n]$ is a periodic signal or close to a periodic signal, the single reflection will not help reduce the Gibbs phenomenon. It is better to use the original signal in the filtering process.

5.14 Double Reflection Reduces Gibbs Phenomenon

This section studies the double reflection aiming to understand when it reduces the Gibbs phenomenon. Compared to the single reflection, the double reflection tries to preserve first

derivative continuity in the discrete signal. Consider the DFT of the double reflection signal $dr[n]$, denoted $DR[k]$ and define $W_{4N} = e^{-\frac{j2\pi}{4N}}$.

$$DR[k] = \sum_{n=0}^{N-1} u[n]W_{4N}^{nk} + \sum_{n=N}^{2N-1} (2u[N-1] - u[2N-1-n])W_{4N}^{nk} + \sum_{n=2N}^{3N-1} (2u[N-1] - u[n])W_{4N}^{nk} + \sum_{n=3}^{4N-1} u[2N-1-n]W_{4N}^{nk} \quad (5\text{-}41)$$

By changing of variables, one can simplify the equation

$$DR[k] = \sum_{n=0}^{N-1} u[n](W_{4N}^{nk} - W_{4N}^{(2N-1-n)k} - W_{4N}^{(2N+n)k} + W_{4N}^{(4N-1-n)k}) + \sum_{n=N}^{3N-1} 2u[N-1]W_{4N}^{nk} \quad (5\text{-}42)$$

We denote the first summation on the right side of the Equation (5-42) by $DR_1[k]$, and the second summation by $DR_2[k]$, so that $DR[k] = DR_1[k] + DR_2[k]$. For $DR_1[k]$, note that the 2nd term and 3rd term contain $W_{4N}^{2Nk} = (-1)^k$, and the last term has $W_{4N}^{4Nk} = 1$, then

$$DR_1[k] = \sum_{n=0}^{N-1} u[n](W_{4N}^{nk} + (-1)^{k+1}W_{4N}^{-(n+1)k} + (-1)^{k+1}W_{4N}^{nk} + W_{4N}^{-(n+1)k}) \quad (5\text{-}43)$$

When k is an even number, $DR_1 = 0$; when k is an odd number, we need to check its high frequency range value. For a high frequency range like $k = 2N - m$, where m is a positive integer, and $m \ll N$, since $e^{\frac{\pi m}{2N}} \approx 0$,

$$DR_1[2N-m] = 2\sum_{n=0}^{N-1} u[n](W_{4N}^{nk} + W_{4N}^{-(n+1)k})$$

$$= 2\sum_{n=0}^{N-1} u[n]\{e^{-jn\left(\pi-\frac{\pi m}{2N}\right)} + e^{-j(n+1)\left(\pi-\frac{\pi m}{2N}\right)}\} \tag{5-44}$$

$$\approx 2\sum_{n=0}^{N-1} u[n]\{e^{-jn\pi} + e^{-j(n+1)\pi}\} = 0$$

Thus, we know that when k is even, $DR_1[k] = 0$; and when k is odd, $DR_1[k] \approx 0$. For the second summation $DR_2[k]$, when $k = 0$, $DR_2[k] = 4Nu[N-1]$; $k \neq 0$, then

$$DR_2[k] = \sum_{n=N}^{3N-1} 2u[N-1]W_{4N}^{nk} = 2u[N-1]\sum_{n=N}^{3N-1} \frac{W_{4N}^{nk}\left(1 - W_{4N}^{2Nk}\right)}{1 - W_{4N}^{k}} \tag{5-45}$$

Previously, we calculated that $W_{4N}^{2Nk} = (-1)^k$, when k is an even number but not a zero, $DR_2[k] = 0$; when k is odd, in the high frequency range as $k = 2N - m$, where m is positive integer, and $m \ll N$, so that $e^{-\frac{j\pi m}{N}} \approx 1$

$$DR_2[k] = 2u[N-1]\sum_{n=N}^{3N-1} \frac{W_{4N}^{nk}\left(1 - W_{4N}^{2Nk}\right)}{1 - W_{4N}^{k}} = 2u[N-1]\frac{\mp 2j}{1 + e^{-\frac{j\pi m}{N}}} \tag{5-46}$$

$$\approx \pm 2u[N-1]j$$

In short,

$$DR[k] = \begin{cases} 2Nu[N-1], & k = 0 \\ 0, & k \neq 0, but\ k\ is\ even \\ \pm 2u[N-1]j, & k\ is\ odd, in\ high\ freqeucny\ range \end{cases} \tag{5-47}$$

Recall that in the DFT transformation pair, there is a coefficient of $\frac{1}{N}$ in front of the inverse DFT matrix for a length N signal. This coefficient is $\frac{1}{N}$ for the original signal, but for single reflection, this becomes $\frac{1}{2N}$, and in double reflection it becomes $\frac{1}{4N}$, since the signal length is double and quadruple. If we move this coefficient from inverse DFT matrix to DFT matrix,

meaning the DFT definition is $U[k] = \frac{1}{N}\sum_{n=0}^{N-1} u[n]e^{-\frac{j2\pi kn}{N}}$ instead of Equation (5-32), then Equation (5-47) can be rewritten as

$$DR[k] = \begin{cases} \frac{u[N-1]}{2}, & k = 0 \\ 0, & k \neq 0, but\ k\ is\ even \\ \pm\frac{u[N-1]j}{2N}, & k\ is\ odd, in\ high\ freqeucny\ range \end{cases} \quad (5\text{-}48)$$

This means that the odd term in the high frequency range can still be treated as 0 if N is large enough. The same statement applied to single reflection also applies here that if the input signal $x[n]$ has the property that the magnitude of this summation $\sum_{n=0}^{N-1} |(-1)^n x[n]| \gg 0$, then the single reflection will reduce the Gibbs phenomenon.

Double reflection will not help if the signal is periodic or has the same starting and ending point. It helps for the signal having a significant discontinuity.

Compared to the single reflection, there are more terms that can be thought of as 0 in the high frequency range since all the even k in the high frequency range are actually zero. One should also notice that in this double reflection, the term we are neglecting is $\frac{\pi m}{2N}$ but in single reflection that term is $\frac{2\pi m}{N}$, this suggests that in double reflection, we can treat more high frequency component terms as 0 compared to a single reflection. Also, in the double reflection, the DFT gives the result that the signal's frequency spectrum is moving further towards the low-frequency range compared to the single reflection case.

5.15 Simulation

ILC simulations are performed using a 3rd order system that models all links of a Robotic Research Corporation robot treated in Reference 9. The Laplace transfer function model for each link is

$$G(s) = \left(\frac{a}{s+a}\right)\left(\frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2}\right) \tag{5-49}$$

where $a = 8.8, \zeta = 0.5, \omega_n = 37$. The input comes through a zero-order hold updating at 100Hz. The discrete transfer function $G(z)$ producing the same output at the sample times, will have two zeros introduced by the conversion to discrete time: one inside the unit circle and one outside the unit circle. The partial isometry ILC law is used. After converting to a discrete time state space model and computing matrix *P* with singular value decomposition $P = USV^T$, the ILC law is given by $L = VU^T$. The desired trajectory is chosen as a 5th order polynomial from 0 to 1 in 100 time steps. The command is the sampled version of this polynomial, and the polynomial has the property that its value and its first derivative are both zero at time zero, and the first derivative is zero at the end of the trajectory.

The numerical study tests three types of zero-phase low-pass filters: the Cliff Filter with a sharp cutoff, the MATLAB default Filtfilt command using the 5th order Butterworth filter, and a Harmonic Butterworth filter which only uses the magnitude response of the same 5th order Butterworth filter. Since the Filtfilt command gives the zero-phase Butterworth filter having the square of magnitude of the Butterworth filter, the corresponding Harmonic filter also has its magnitude response to be the square of the magnitude response of the Butterworth filter for comparison. All the filters compared has the same 20Hz cutoff, and the desired output of the robot linnk is the 5th order polynomial from 0 to 1 in 100 time steps. Figure 5-13 presents the command to the system with input signal to the filter of no reflection, single reflection, and double reflection after 5000 iterations. The first row gives command to the system using Cliff Filter cutoff at 20Hz (Nyquist frequency is 50Hz) after 5000 iterations, the second row is the result of Filtfilt, and the third row is the result of Harmonic Butterworth Filter respectively.

From Figure 5-13, if there is no reflection of the input signal, both the Cliff Filter and the Harmonic Butterworth Filter have large undesirable oscillation at both ends. Introducing a single-reflection or a double-reflection, significantly reduces the oscillation for the Cliff Filter and the Harmonic Butterworth Filter. The Filtfilt is not sensitive to whether you reflect or do not reflect the signal for the smooth signal considered here.

Figure 5-14 presents corresponding results when the desired trajectory is changed to a parabolic signal increasing from 0 to 4 in 100 time steps. This signal starts from zero with zero slope. However, when it is represented by sine and cosine functions of period 100 time steps, these functions try to fit a cusp occurring at the end of the trajectory. Without reflecting the signal, both the Cliff Filter and the Harmonic Butterworth Filter will have substantial undesired oscillation. The single reflection significantly reduces the oscillation for both filters. The double reflection also reduces the oscillation. For Filtfilt, its Gibbs phenomenon is not obvious for all three reflections compared.

From Equation (1-14), one can calculate the RMS error between the desired output and the output of the 3^{rd} order robot link model after the learning process finishes for filters without reflection and for filters using single reflection. Table 5-1 gives the RMS errors of the outputs for the 5^{th} order polynomial input for no reflection and single reflection of the input signal to the filters. The Root Mean Square (RMS) errors in the table indicate that the single reflection method for this smooth signal does not significantly affect Filtfilt results. But single reflection does help with both the Cliff and the Harmonic Butterworth filter cases. Table 5-2 gives the RMS errors of the output for the parabolic signal after the learning process finishes. For the parabolic signal, the, single reflection methods reduce the RMS error of the output for all three filters.

As suggested by Equation (5-32), for the double reflection method, one could not have an analytical formula of the command to the system after the learning process finishes, thus, one could not know its RMS error between the desired output and the output of the 3rd order robot link model after the learning process finishes. But, we can simulate such results for large number of iterations. Figure 5-15 and Figure 5-16 gives the history of RMS error between the desired output and the output of the robot link for the 5th order polynomial and parabolic signal as the desired output respectively in 5000 iterations of learning. In both figures, the RMS error is expressed in the log-scale of base 10 in the y-axis, and the x-axis shows the numbers of iterations. Table 5-3 and 5-4 shows the RMS error between the desired output and the output of the 3rd order robot link for the 5th order polynomial and the parabolic as the desired output after 5000 iterations of learning respectively.

One compares the Table 5-1 and Table 5-3, one can see that for Filtfilt, after 5000 iterations, its RMS error of the two are not same, but for Cliff Filter and Harmonic Filters they are the same for both no reflection and single reflection. The same applies to Table 5-2 and Table 5-4. This indicates that for Filtfilt command, in 5000 iterations, it does not finish the learning process, but for Cliff and Harmonic Filter, they finish the learning process. In fact, for the polynomial desired output, the Cliff Filter with no reflection has the RMS error same as its final value after the 2139 iterations, but for Filtfilt, its RMS error drops to the lowest level at after 121 iterations and starts to slowly increase afterwards. It is more apparent in Figure 5-16. The RMS error of Filtfilt with no reflection drops to the lowest level 2.5250×10^{-5} at iteration of 2027 and then it starts to gradually increasing to 1.1385×10^{-4} with an increase of 1×10^{-8} for every iteration after 4000 iterations. Based on Figure 5-15 and Figure 5-16, the RMS errors for Cliff Filter and Harmonic Filter converges to RMS error of the final level when the ILC learning

process finishes. But for Filtfilt, its RMS error is still converging to the RMS error of the final level very slowly.

5.16 Discussion

Both single reflection and double reflection methods increase the computational complexity. The filter's input and output relation is characterized by a matrix P of size N by N if zero-initial conditions are considered. To compute the first element of the length N output, it involves N multiplication and $N-1$ addition, and there are N elements in the output in total. If we use the big O notation in computer science to quantify the worst-case time complexity of the running time of the computing, the filtering process's running time is $O(N^2)$, where N is the size of the input signal. The running time is growing as a quadratic function of the input signal length N. The single reflection of the input signal doubles the original signal length, and double reflection method quadruples the length of the original input signal. The running time is at least 4-times and 16-times of the running time of no reflection of the signal, respectively. This is the weakness of the single reflection and double reflection method, it reduces the Gibbs phenomenon and tracking error of the output at the cost of increasing the time complexity of computation. Converging faster, versus smaller tracking error. If you decide to stop learning at certain error level, maybe it is less time.

One could argue a potential of using multi-reflection methods to further reduce the effects of transients for Filtfilt command. For example, in the dissertation, we only illustrate single-reflection and double reflection method and use the first half and first quarter of the filtered signal as the command to the system. A multi-reflection method, for example, can reflect signal twice: one could do an even reflection at the end of the original signal of length N, and then another even reflection at the end of the result of the first reflection. The new signal is

reflected twice to be a new signal of length $4N$. One could then filter this new signal, and use the third quarter of the filtered output as the command to the system. Since the length of the signal is increasing, the third quarter of signal is likely closer to steady state. However, one need to consider the computing complexity of multi-reflection methods. The time complexity of the filtering process is $O(N^2)$, where N is the length of input signal. Multiple reflection will significantly increase the running time. For the multi-reflection method mentioned above, its running time will be at least 16 times that of filtering the original signal with no reflection. Therefore, multi-reflection methods improve filtered result at the costs of increasing running time.

In the discussion, we use the reflection of the original signal to build a new signal to reduce the Gibbs phenomenon introduced by the implied jump discontinuity. In fact, one could design any signal extension that brings the signal back to its initial value. The reason we choose the reflection of the original signal is to keep the signal frequency domain spectrum after the extension to be as close as possible to that before the extension. If we use a signal extension method that significantly changes the frequency spectrum of the original signal in the low-frequency range, then after the filtering process, those changes in the low-frequency domain are kept. This reduces the Gibbs phenomenon but it changes the frequency spectrum of the original signal, and the filtered result may be far from our desired trajectory. Thus, it is intuitive and desired to use the reflection of the original signal as the extension of the original signal that gives the minimal modification to the original signal low-frequency components.

5.17 Conclusion

In this thesis, we discuss the reasons why the ILC problem needs a zero-phase low-pass filter in the applications. We also suggest that such a zero-phase low-pass filter should give the

steady-state response since the transients may destabilize the ILC system. We give three methods of producing a steady-state frequency response filter, all of which are based on use of DFT. A series of tests are reported for a Cliff Filter with a sharp frequency cutoff, a DFT based Harmonic Filter with the cutoff based on a chosen Butterworth filter cutoff, and the Filtfilt cutoff picks forward and backward filter initial conditions to minimize the difference between the two filter results. One expects that in essentially all ILC iterations the start and the end of the signal to be filtered will not be the same. Once converged they can be the same if the desired trajectory has this property, but it needs not. Both extending the signal to be filtered with a single reflection around the end time, and a double reflection that aims to preserve continuity of the first derivative help reduce undesirable oscillation at both ends for the Cliff Filter and the Harmonic Butterworth Filter. It also suggests that a single reflection is enough and it gives the best tracking performance at the steady state.

Table 5-1. RMS Error of Output with 20Hz Cutoff for 5th Order Polynomial After the Learning Finishes

Filter Type	No Reflection	Single Reflection
Cliff	1.5348×10^{-4}	1.2684×10^{-6}
Filtfilt	2.5778×10^{-5}	2.5771×10^{-5}
Harmonic	5.1795×10^{-4}	4.0831×10^{-6}

Table 5-2. RMS Error of Output with 20Hz Cutoff for the Parabolic Trajectory After the Learning Finishes

Filter Type	No Reflection	Single Reflection
Cliff	7.8069×10^{-4}	1.7752×10^{-5}
Filtfilt	1.0612×10^{-4}	5.2444×10^{-5}
Harmonic	2.6×10^{-3}	2.5956×10^{-5}

Table 5-3. RMS Error of Output with 20Hz Cutoff for 5th Order Polynomial After 5000 Iterations of Learning

Filter Type	No Reflection	Single Reflection	Double Reflection
Cliff	1.5348×10^{-4}	1.2684×10^{-6}	1.2938×10^{-6}
Filtfilt	4.4285×10^{-6}	1.6346×10^{-6}	1.6021×10^{-6}
Harmonic	5.1795×10^{-4}	4.0831×10^{-6}	4.0702×10^{-6}

Table 5-4. RMS Error of Output with 20Hz Cutoff for the Parabolic Trajectory After 5000 Iterations of Learning

Filter Type	No Reflection	Single Reflection	Double Reflection
Cliff	7.8069×10^{-4}	1.7752×10^{-5}	5.3610×10^{-5}
Filtfilt	1.1385×10^{-4}	2.4613×10^{-5}	4.6804×10^{-5}
Harmonic	2.6×10^{-3}	2.5956×10^{-5}	4.7524×10^{-5}

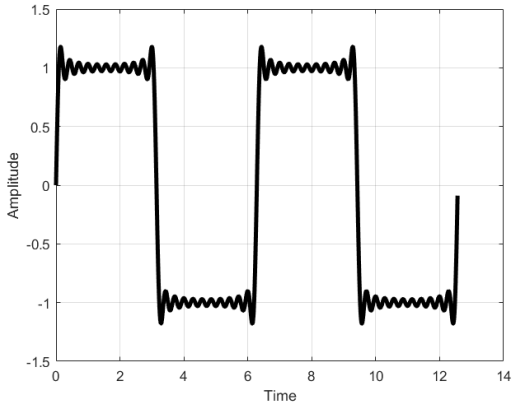

Figure 5-1. 10-term partial sum of the Fourier series of a square wave

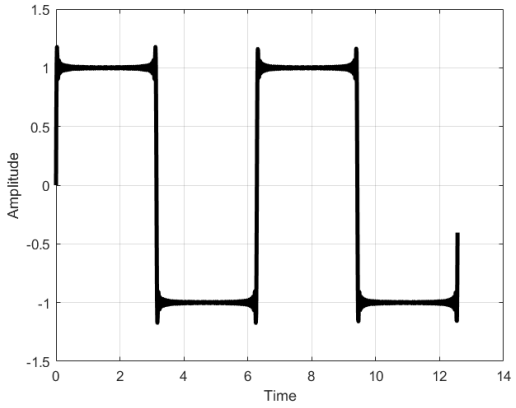

Figure 5-2. 50-term partial sum of the Fourier series of a square wave

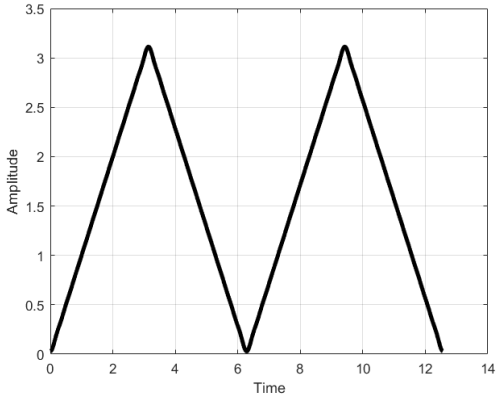

Figure 5-3. 10-term partial sum of the Fourier series of the triangle wave

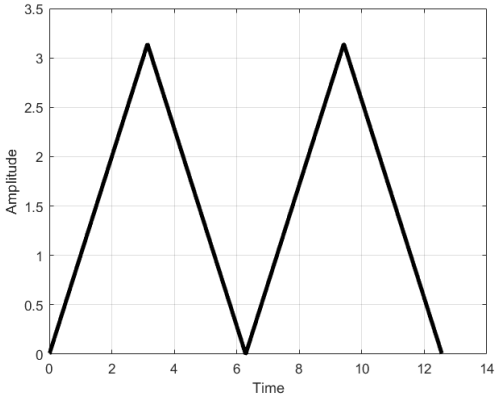

Figure 5-4. 50-term partial sum of the Fourier series of the triangle wave

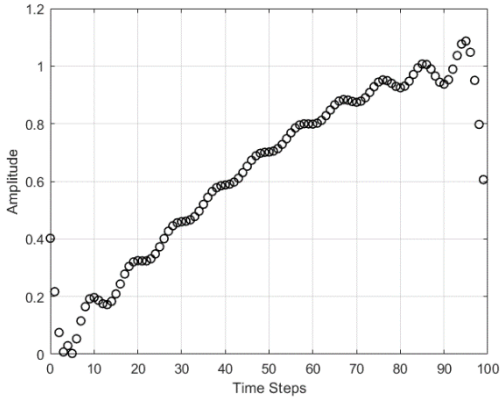

Figure 5-5. 11-terms summation of DFT a ¼ sine wave of Length 100

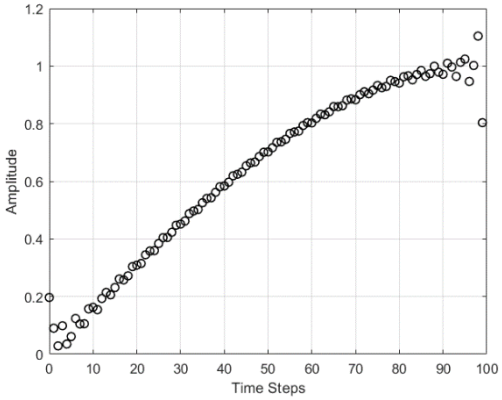

Figure 5-6. 31-terms summation of DFT of a ¼ sine wave of length 100

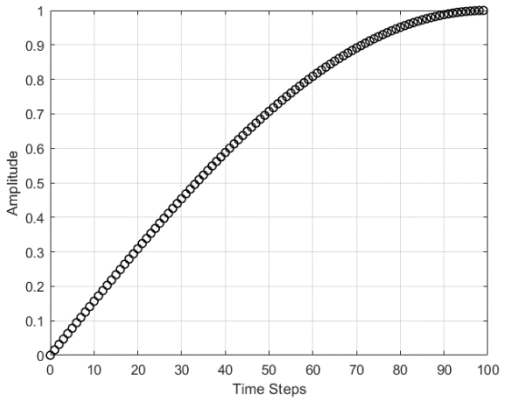

Figure 5-7. Adding all terms of DFT for a ¼ sine wave

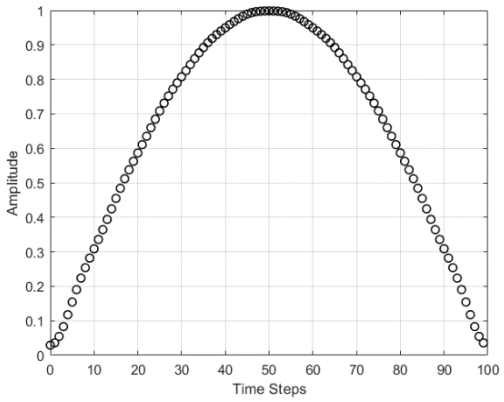

Figure 5-8. Adding 11 terms of DFT for a ½ sine wave of length 100

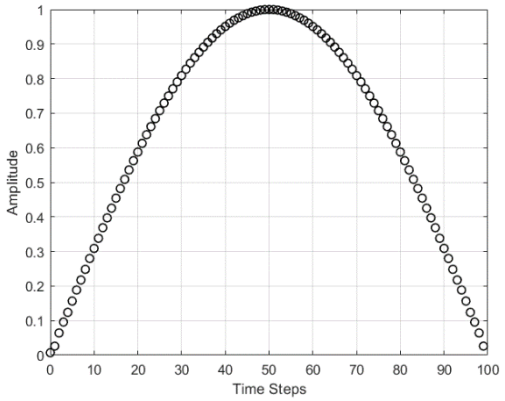

Figure 5-9. Adding 21 terms of DFT for a ½ sine wave of length 100

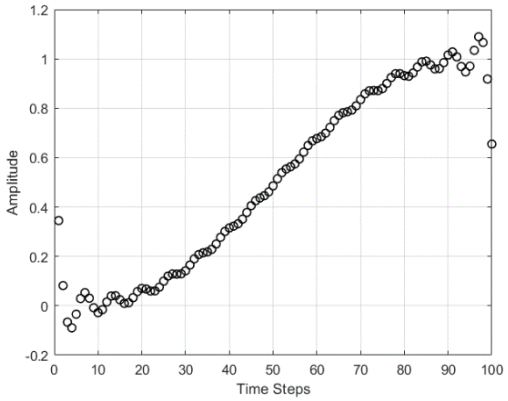

Figure 5-10. 5th order polynomial filtered result using Cliff Filter of 15Hz cutoff

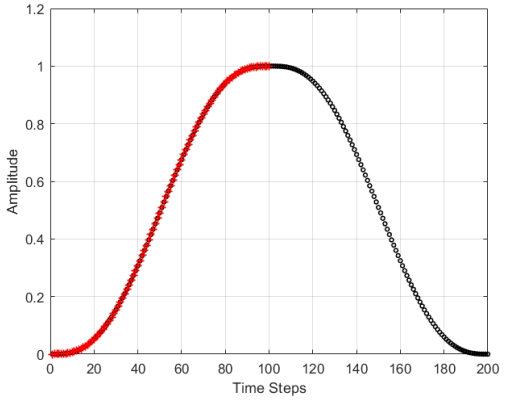

Figure 5- 11. Single reflection illustration

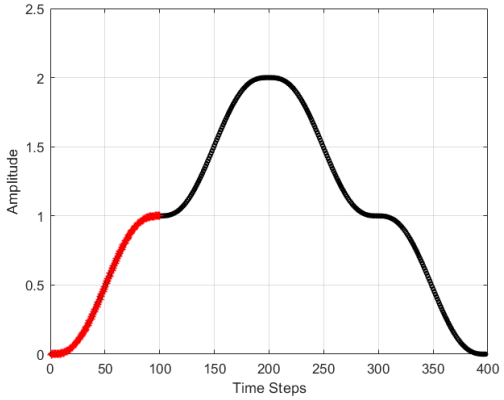

Figure 5-12. Double reflection illustration

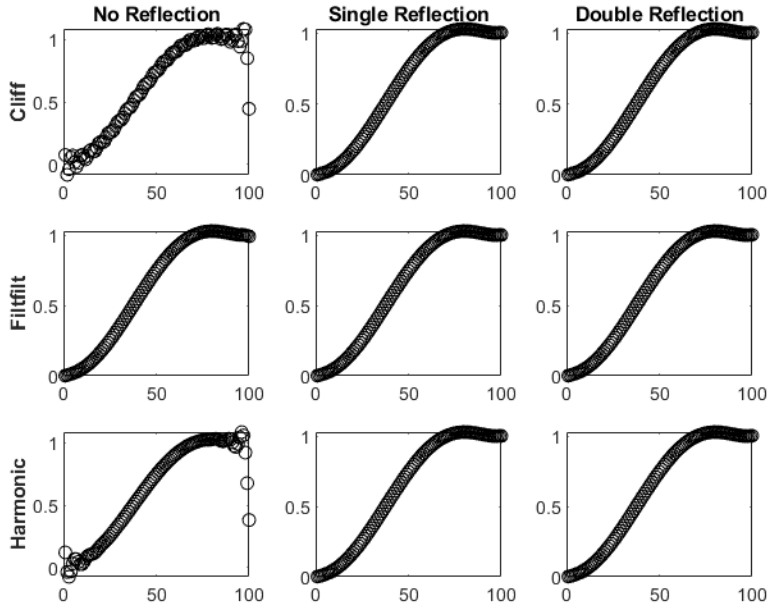

Figure 5-13. The command to the system for the 5th order polynomial as the desired trajectory, and a 20Hz cutoff filters after 5000 iterations of ILC

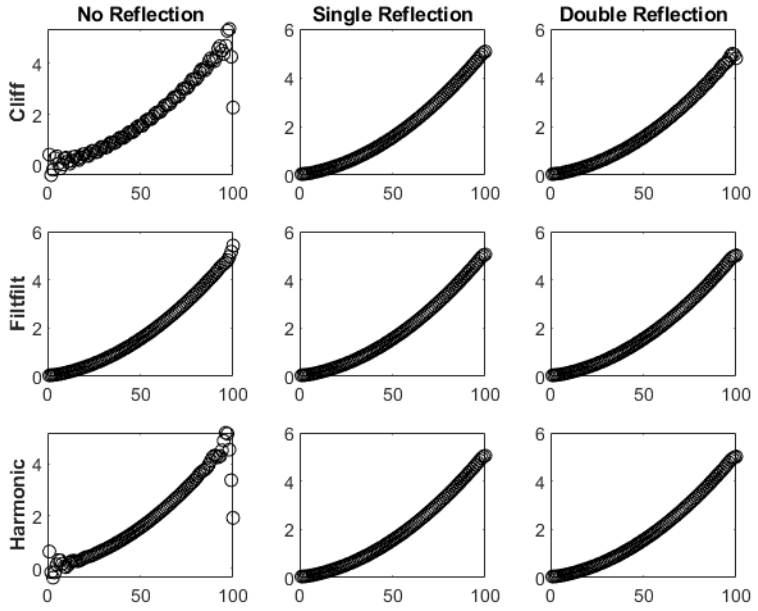

Figure 5-14. The command to the system for the parabolic input as the desired trajectory, and a 20Hz cutoff filters after 5000 iterations of ILC

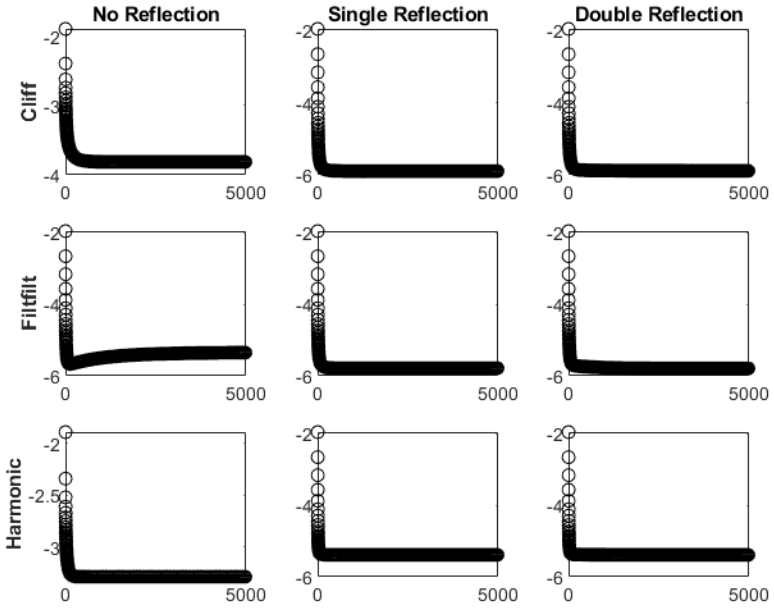

Figure 5-15. The history of RMS error in log scale of the output for the 5th order polynomial as the desired trajectory in 5000 iterations of ILC

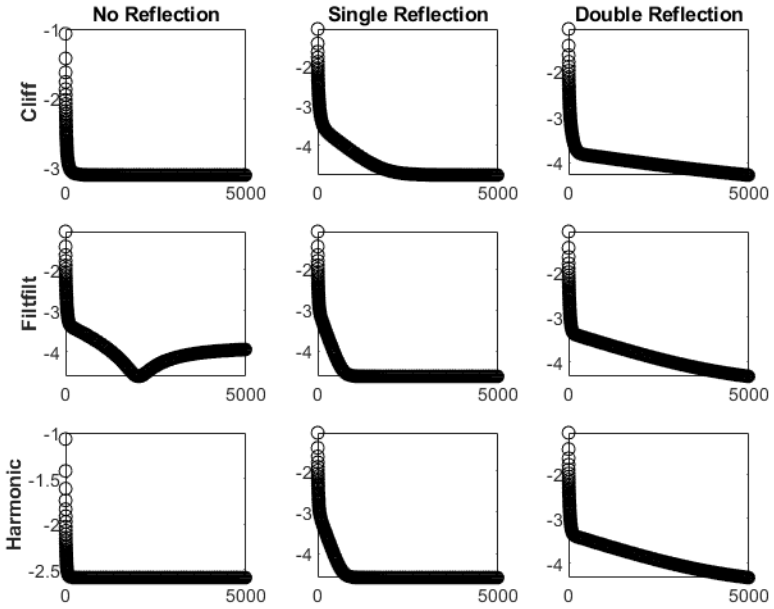

Figure 5-16. The RMS error in log scale of the output for the parabolic as the desired trajectory in 5000 iterations of ILC

Conclusion

Both Repetitive Control (RC) and Iterative Learning Control (ILC) aim at zero tracking error of the command. The RC problem is to tracking a periodic command, and the tracking error for each period of the command decreases as periods progress with the constant or periodic disturbance existing in the feedback system; ILC is to track the command repeatedly, and the tracking error for each time step of the command decreases as runs progress with the repeated disturbance in the feedback system. In ILC, the feedback system returns to the same initial conditions at the start of each run. Both RC and ILC design involve the inverse of the feedback system. The inverse of the feedback system is often unstable, and it is the challenge for both RC and ILC design. This dissertation addresses this challenge in RC and ILC.

In RC design, one of the challenges is to compensate for the sampling zeros outside the unit circle from the feedback system discrete-time model. The previous RC design methods are discussed and tested assuming those sampling zeros are from the conversion of a continuous-time feedback system model with zero-order hold. In the physical world, however, the feedback system can either be continuous or discrete and can have different structure. Both the number of sampling zeros and their asymptotic locations are affected by the feedback system structure and its continuous/discrete components. The thesis discusses the asymptotic location of the sampling zero for different feedback systems, and shows that the previous RC design is still applicable. Moreover, the thesis also compared the performance of each RC design method. The desired RC law would have a faster learning speed in the low-frequency range to achieve a faster convergence to the command and a slower learning speed in the high-frequency range to increase the robustness to the model error of the system. The thesis shows that the RC design method using the cost function has the desired learning speed with a faster speed in low frequencies and

a slower learning speed for high frequencies, but RC design methods based on partial sum of Taylor expansion of the system inverse transfer function does not have this characteristic and they are more likely to be unstable if its RC law only uses a fewer number of errors in the previous period. This discussion is presented in Chapter 2.

ILC is a finite-time problem since the command is finite-time. ILC decreases its tracking error by updating the command to the feedback system every run. The zero-tracking error suggests that the command to the feedback system is the multiplication of the system inverse and the desired output. As the inverse transfer function of the feedback system is often unstable, the modified command by RC/ILC is often unstable. This is the instability issue in ILC. Moreover, ILC is very sensitive to model error, and this is the robustness issue in ILC. The solution to instability and robustness issues of ILC is to use a zero-phase low-pass filter. The filter is designed using the frequency thinking, but ILC is a finite-time problem. This is the basic mismatch in ILC, the second part of dissertation from chapter 3 to 5 address this mismatch.

One solution to this mismatch, discussed in this thesis, is the partial inverse of system based on singular value decomposition in chapter 3. This partial inverse of the system can be used as the learning gain matrix in ILC with the function of the frequency cutoff at the same time. This partial inverse of the system can also be used as a prefilter to modify the command to the feedback system to raise its bandwidth. However, in both applications, one needs to have an accurate model of the feedback system up to a certain frequency.

The other solution to this mismatch, discussed in chapter 4 and 5 in the thesis, is to have a zero-phase low-pass filter that gives the steady-state response for a finite time system. Chapter 4 discusses two such filters: Circulant Filter and Cliff Filter. It proves that Circulant Filter's eigenvector matrix is the DFT matrix and the eigenvalues are the steady-state response of a filter.

It also proves that the Cliff Filter is a special case of the Circulant Filter with the characteristics of an ideal filter. A typical Circulant Filter has a phase lag, and one needs to make the Circulant Filter a zero-phase filter. It is proved that the Circulant Filter used as a zero-phase filter, its optimal initial conditions are zero, a verification that the Circulant Filter gives the steady-state response of the filter with no transients in the output.

Chapter 5 discusses the issues in the application of Circulant Filter and Cliff Filter. One issue of the Circulant Filter, for example, is the previous research does not explain the frequency response difference between the Circulant Filter designed based on Butterworth filter parameters and a classic Butterworth filter at the steady state. It is proved that the circulant Butterworth filter, if one gets the Circulant Filter from Butterworth filter parameters, gives the frequency response of the FIR filter whose pulse response of length N is the same as the first length-N pulse response of the Butterworth filter. The derivation process also gives a formula to estimate the error of the magnitude response between circulant Butterworth filter and that Butterworth filter at the steady state, and one also can use the formula to find the size of Circulant Filter needed to ensure the difference between the two is less than a threshold.

The second issue is the Gibbs phenomenon for the input with different starting and ending points. Both Cliff and Circulant Filter belong to DFT-based filter. DFT suggests the signal is periodic and input with different starting and ending points implies jump discontinuity, introducing oscillation at both ends for the filtered result, which is the Gibbs phenomenon. Such oscillation reduces the tracking accuracy in ILC. Chapter 5 discusses the single reflection and double reflection of the signal to address this issue. It gives a math formula to estimate the DFT of the original signal and single-reflected/double reflected signal in the high frequency range The formula suggests that single reflection/double reflection methods reduce the Gibbs phenomenon

if the $\sum_{n=0}^{N-1}(-1)^n u[n]$ is significantly larger than 0, where $u[n]$ is the original signal having different starting and ending points. It also demonstrates that for DFT-based filters, e.g. Circulant Filter and Cliff Filter, single/double reflection method can increase the tracking accuracy in ILC for the command with different starting and ending points.

References

[1] S. Arimoto, S. Kawamura, and F. Miyazaki, 1984, Bettering operation of robots by learning. *Journal of Robotic Systems*, Vol. 1, 1984, pp.123-140.

[2] G. Casalino, and G. Bartolini, "A learning procedure for the control of movements of robotic manipulators". *IASTED Symposium on Robotics and Automation*, Amsterdam, The Netherlands, 1984, pp. 108-111.

[3] J. J. Craig, "Adaptive control of manipulators through repeated trials". *Proceedings of the American Control Conference*, San Diego, USA, 1984, pp. 1566-1573.

[4] T. Inoue, M. Nakano, and S. Iwai, 1981, High accuracy control of a proton synchrotron magnet power supply. *Proceedings of the 8th World Congress of IFAC*, Vol. 20, 1981, pp. 216-221.

[5] T. Omata, M. Nakano, and T. Inoue, "Applications of repetitive control method to multivariable systems". Transactions of SICE, 20, 1984, pp. 795-800.

[6] S. Hara, and Y. Yamamoto, Y., "Synthesis of repetitive control systems and its applications". *Proceedings of the 24th IEEE Conference on Decision and Control*, Fort Lauderdale, FL, 1985, pp. 326-327.

[7] M. Phan and R.W. Longman, "A mathematical theory of learning control for linear discrete multivariable systems". *Proceedings of the AIAA/AAS Astrodynamics Conference*, Minneapolis, Minnesota, USA, 1998, pp. 740-746.

[8] H. S. Jang and R. W. Longman, "A New Learning Control Law with Monotonic Decay of the Tracking Error Norm," *Proceedings of the Thirty-Second Annual Allerton conference on Communication, Control, and Computing*, Monticello, IL, September, 1994, pp. 314-323.

[9] 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.

[10] J. Bao, and R. W. Longman, "Unification and Robustification of Iterative Learning Control Laws," *Advances in the Astronautical Sciences*, Vol. 136, 2010, pp. 727-745.

[11] R. W. Longman, "Iterative Learning Control and Repetitive Control for Engineering Practice," *International Journal of Control*, Special Issue on Iterative Learning Control, Vol. 73, No. 10, July 2000, pp. 930-954.

[12] J. Juang and R. W. Longman, "Iterative Learning Control Inverse Problem Using Harmonic Frequency Filters," *Advances in the Astronautical Sciences*, to be published

[13] Y. C. Huang and R. W. Longman, "The source of the often observed property of initial convergence followed by divergence in learning and repetitive control". *Advances in the Astronautical Sciences*, Vol. 90, 1996, pp. 555-572.

[14] B. Panomruttanarug and R. W. Longman, "Repetitive Controller Design Using Optimization in the Frequency Domain," *Proceedings of the 2004 AIAA/AAS Astrodynamics Specialist Conference*, Providence, RI, August 2004.

[15] K. Xu and R. W. Longman, "Use of Taylor Expansions of the Inverse Model to Design FIR Repetitive Controllers," *Advances in the Astronautical Sciences*, Vol. 134, 2009, pp. 1073-1088.

[16] P. Prasitmeeboon, and R. W. Longman, "Using Quadratically Constrained Quadratic Programming to Design Repetitive Controllers: Application to Non-Minimum Phase Systems," *Advances in the Astronautical Sciences*, Vol. 156, 2016, pp. 1647-1666.

[17] K. Åström, P. Hagander, and J. Strenby, "Zeros of Sampled Systems," *Proceedings of the Nineteenth IEEE Conference on Decision and Control*, 1980, pp. 1077-1081.

[18] R. W. Longman and W. Kang, "Issues in Robustification of Iterative Learning Control Using a Zero-Phase Filter Cut Off," *Advances in the Astronautical Science*, Vol. 127, 2007, pp. 1683–1702.

[19] A. M. Plotnik and R. W. Longman, "Subtleties in the Use of Zero-Phase Low-Pass Filtering and Cliff Filtering in Learning Control," *Advances in the Astronautical Sciences*, Vol. 103, 1999, pp. 673-692.

[20] H. Elci, R. W. Longman, M. Phan, J.-N. Juang, and R. Ugoletti, "Discrete Frequency Based Learning Control for Precision Motion Control," *Proceedings of the 1994 IEEE International Conference on Systems, Man, and Cybernetics*, San Antonio, TX, Oct. 1994, pp. 2767-2773.

[21] Phan, Minh Quang. *A mathematical theory of learning control*. Doctoral Dissertation. Columbia University, 1989.

[22] K. Chen, and R. W. Longman, "Creating a Short Time Equivalent of Frequency Cutoff for Robustness in Learning Control," *Advances in the Astronautical Sciences*, Vol.114, 2003, pp. 95-114.

[23] F. Gustafsson, "Determining the Initial States in Forward-Backward Filtering," *IEEE Transactions on Signal Processing*, Vol. 44, No. 4, 1996, pp.988-992.

[24] B. Song and R. W. Longman, "Circulant Zero-Phase Low Pass Filter Design for Improved Robustification of Iterative Learning Control," *Advances in the Astronautical Sciences*, Vol. 156, 2016, pp. 2161-2180.

[25] H. Elci, R. W. Longman, M. Q. Phan, J.-N. Juang, and R. Ugoletti, "Simple Learning Control Made Practical by Zero-Phase Filtering: Applications to Robotics," *IEEE Transactions*

on Circuits and Systems I: Fundamental Theory and Applications, Special Issue on Multidimensional Signals and Systems, Guest Editors: S. Basu and M. N. S. Swamy, Vol. 49, No. 6 June 2002, pp. 753-767.

[26] H. Elci, M. Phan, R. W. Longman, J.-N. Juang, and R. Ugoletti, "Experiments in the use of Learning Control for Maximum Precision Robot Trajectory Tracking," *Proceedings of the 1994 Conference on Information Sciences and Systems*, Princeton, NJ, March 1994, pp. 951-958.

[27] A. V. Oppenheim, *Discrete-time signal processing*. Pearson Education, 1999.

[28] R. M. Gray, "Toeplitz and circulant matrices: a review." *Foundations and Trends in Communications and Information Theory.* Vol. 2, Issue. 3, 2006, pp. 155-240.

[29] T. McCreary, "On frequency sampling digital filters." *IEEE Transactions on Audio and Electroacoustics*, Vol. 20, Issue. 3, 1972, pp.222-223.

[30] R. W. Longman, K. Xu, and B. Panomruttanarug, "Designing Learning Control that is Close to Instability for Improved Parameter Identification," *Modeling, Simulation and Optimization of Complex Processes*, Bock, Kostina, Phu, and Rannacher Editors, Springer-Verlag, Heidelberg, 2008, pp. 359-370.