New Stable Inverses of Linear Discrete Time Systems and Application to Iterative Learning Control

Xiaoqiang Ji

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

COLUMBIA UNIVERSITY

2019

ABSTRACT

New Stable Inverses of Linear Discrete Time Systems and Application to Iterative Learning Control

Xiaoqiang Ji

Digital control needs discrete time models, but conversion from continuous time, fed by a zero order hold, to discrete time introduces sampling zeros which are outside the unit circle, i.e. non-minimum phase (NMP) zeros, in the majority of the systems. Also, some systems are already NMP in continuous time. In both cases, the inverse problem to find the input required to maintain a desired output tracking, produces an unstable causal control action. The control action will grow exponentially every time step, and the error between time steps also grows exponentially. This prevents many control approaches from making use of inverse models.

The problem statement for the existing stable inverse theorem is presented in this work, and it aims at finding a bounded nominal state-input trajectory by solving a two-point boundary value problem obtained by decomposing the internal dynamics of the system. This results in the causal part specified from the minus infinity time; and its non-causal part from the positive infinity time. By solving for the nominal bounded internal dynamics, the exact output tracking is achieved in the original finite time interval.

The new stable inverses concepts presented and developed here address this instability problem in a different way based on the modified versions of problem states, and in a way that is more practical for implementation. The statements of how the different inverse problems are posed is presented, as well as the calculation and implementation. In order to

produce zero tracking error at the addressed time steps, two modified statements are given as the initial delete and the skip step. The development presented here involves: (1) The detection of the signature of instability in both the nonhomogeneous difference equation and matrix form for finite time problems. (2) Create a new factorization of the system separating maximum part from minimum part in matrix form as analogous to transfer function format, and more generally, modeling the behavior of finite time zeros and poles. (3) Produce bounded stable inverse solutions evolving from the minimum Euclidean norm satisfying different optimization objective functions, to the solution having no projection on transient solutions terms excited by initial conditions.

Iterative Learning Control (ILC) iterates with a real world control system repeatedly performing the same task. It adjusts the control action based on error history from the previous iteration, aiming to converge to zero tracking error. ILC has been widely used in various applications due to its high precision in trajectory tracking, e.g. semiconductor manufacturing sensors that repeatedly perform scanning maneuvers. Designing effective feedback controllers for non-minimum phase (NMP) systems can be challenging. Applying Iterative Learning Control (ILC) to NMP systems is particularly problematic. Incorporating the initial delete stable inverse thinkg into ILC, the control action obtained in the limit as the iterations tend to infinity, is a function of the tracking error produced by the command in the initial run. It is shown here that this dependence is very small, so that one can reasonably use any initial run. By picking an initial input that goes to zero approaching the final time step, the influence becomes particularly small. And by simply commanding zero in the first run, the resulting converged control minimizes the Euclidean norm of the underdetermined control history. Three main classes of ILC laws are examined, and it is shown that all ILC

laws converge to the identical control history, as the converged result is not a function of the ILC law. All of these conclusions apply to ILC that aims to track a given finite time trajectory, and also apply to ILC that in addition aims to cancel the effect of a disturbance that repeats each run.

Having these stable inverses opens up opportunities for many control design approaches. (1) ILC was the original motivation of the new stable inverses. Besides the scenario using the initial delete above, consider ILC to perform local learning in a trajectory, by using a quadratic cost control in general, but phasing into the skip step stable inverse for some portion of the trajectory that needs high precision tracking. (2) One step ahead control uses a model to compute the control action at the current time step to produce the output desired at the next time step. Before it can be useful, it must be phased in to honor actuator saturation limits, and being a true inverse it requires that the system have a stable inverse. One could generalize this to p-step ahead control, updating the control action every p steps instead of every one step. It determines how small p can be to give a stable implementation using skip step, and it can be quite small. So it only requires knowledge of future desired control for a few steps. (3) Note that the statement in (2) can be reformulated as Linear Model Predictive Control that updates every p steps instead of every step. This offers the ability to converge to zero tracking error at every time step of the skip step inverse, instead of the usual aim to converge to a quadratic cost solution. (4) Indirect discrete time adaptive control combines one step ahead control with the projection algorithm to perform real time identification updates. It has limited applications, because it requires a stable inverse.

Contents

List of Figures

List of Tables

Acknowledgements

Now is approching the end of my incredible journey as a PhD student at Columbia University. I am extremely grateful as well as proud at this moment. I would like to thank many people who made this adventure happen.

First, my deepest gratitude goes to my academic advisor Prof. Richard W. Longman, for your continuous support, patience, and immense help, and it is you who truly introduced me to the world of academic research. Our weekly meetings forged me with the attitudes towards my life and have been always providing a direction when I got lost.

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. Fred R. Stolfy, for participating in my defense. And special thanks to Prof. Minh Phan, Prof. Matei Ciocarlie, and Prof. Qingze Zou for helping me proposing the research topics. I would also thank the Department of Mechanical Engineering at Columbia supporting my studies.

I would like to thank my parents for your unconditional love have made this study possible. I am also greatly thankful to my girlfriend for everything. I thank all my lab members Dr. Zhu, Dr. Li, Dr. Song, Dr. Vicario, Dr. Prasitmeeboon, Dr. Yao, Tianyi and

Ayman.

Finally, I thank myself...

Chapter 1

Introduction

Motivation and Background

Typical feedback control systems do not do what you ask them to do. The concept of bandwidth is created to describe up to what frequency such a system will do something reasonably close to the command. Various control approaches aim to fix this problem, and produce zero tracking error following the commanded trajectory. These include Iterative Learning Control (ILC), Repetitive Control (RC), one step ahead control, indirect adaptive control, etc. Each aims to produce that input command that produces the desired output, i.e. solve the inverse problem. To implement such control laws, one must use discrete time models which represent the continuous world with inputs coming through a zero-order hold. Assume a one time-step delay through the system, since the time lag from change in input at a given time step to the first time step influenced in the output should be one. Thus, new zeros are introduced during the discretization, and these are termed sampling zeros. When two or more zeros have thus been introduced, at least one zero is outside the unit circle for

reasonable sample rates, making a non-minimum phase system (NMP) in References [1] and [2]. The inverse problem makes these zeros into poles, producing an unstable control action. References [1] and [2] tell the asymptotic locations of the zeros introduced as the sample time interval tends to zero, for each value of pole excess (relative degree), i.e. the number of poles minus the number of zeros. Of course, the original continuous time system might be NMP, and then the image of the zero(s) in the right half of the s-plane will be mapped outside the unit circle in the z-plane. Such zeros are termed intrinsic zeros, and again they make the inverse problem unstable.

There is an existing stable inverse theory developed to address this problem in References ([3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]). The stable inverse concepts presented and fully developed in this thesis address this instability problem in a different way. References [3] and [4] studied the problem for continuous time, nonlinear, NMP systems, aiming to produce a stable non-causal inverse mapping. Researches try to use the concept in the context of Iterative Learning Control (ILC). Reference [8] developed the relationship of adjoint-type ILC and stable inversion. Reference [9] designs a new inversion-based algorithm which works for both minimum and non-minimum phase systems with gain and time-constant uncertainty. Reference [11] aiming at the ILC problem, which examine sampling zeros only, and do not ask for zero error on the output for the first few steps. Reference [12] studies the convergence of stable inverse of sampled-data system to the continuous-time counterpart. Reference [15] proposed optimal state-to-state transition to shorten the preactuation time.

The author and co-workers developed a set of new stable inverses, and the concepts presented in References [16], [17] and [18] and fully developed in this dissertation address

this instability problem in a different which is more practical for implementation. Analytical and numerical on the various ways in which these results can be employed in control system design are presented including ILC.

Iterative learning control (ILC) is a relatively new method of control that aims to achieve zero tracking error of a finite time tracking maneuver that is repeated. The original ILC idea dates back to the late 1970s when Uchiyama introduced the concept on high-speed motion control of a robot arm following a desired trajectory through iterative trials in Reference [19]. In addition to a considerable amount of journal and conference papers, there are also major surveys, books, and special issues in References ([20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33]). ILC has been applied in a wide range of areas. According to the survey performed in Reference [28], the top three application fields are robots, rotary systems and process control including batch/factory/chemical process. Take computer disk drives for example. The data are written while the disk is rotating so there is vibration which results in some high frequency wiggles in the tracks. In order to read data on a disk, a control system follows these tracks on the disk. At the factory, ILC is used on each track to improve the accuracy when the control system follows the track, so that the storage can be increased. There are also applications to spacecraft operations for repeated scanning maneuvers of fine pointing equipment. The learning process can learn to compensate for both repeating effects from structural flexibility, and deterministic control system error is response to time varying tracking commands. There is the potential for high precision pointing control achieved through a learning process.

ILC stores data from the previous run, so that it is a digital control method solving a discrete-time inverse problem. The real world for digital control systems is governed by

ordinary differential equations, but the digital controller creates the forcing function applied to this equation, updating it each sample time. Each update is continuously applied to the differential equation until a new update arrives from the controller - called a zero-order hold. If one looks at the solution to the differential equation at the sample times, one can make a linear difference equation that has identical solution to the differential equation. Reference [20] proves that the process of converting to a difference equation model introduces the forcing function at additional sample times, enough to make the most recent output time step in the equation be one step ahead of the most recent forcing function input time step. When the discretization introduces three or more additional terms, and the sample rate is reasonable, the characteristic polynomial of the forcing function side of the equation will contain a root or roots that are larger than one in magnitude. This makes the discrete-time inverse problem unstable for a majority of digital control systems in the world. The implication is, if one wants to have perfect tracking of a desired discrete-time trajectory at all time steps, the control action needed is unstable, and grows exponentially with time steps. The inverse problem error must be zero at the sample times, but between sample times the solution of the differential equation (after some initial time steps) is growing in magnitude exponentially, and alternating in sign each time step. Of course, this exponential error growth when perfectly following the discrete-time desired trajectory does not address the initial intended problem of finding the input to accurately follow the desired continuous time output.

Thesis Outline

Chapter 2 gives the problem statement and calculation for the existing stable inverse theorem. Chapter 3 develops a series of new stable inverses based on the modified problem statements satisfying different objectives. Chapter 4 applies the stable inverse ideas on ILC design problems.

This page intentionally left blank.

Chapter 2

The Zeros of Discretized Systems

2.1 Introduction

Digital control systems typically contain a system governed by a differential equation whose input comes through a zero-order hold. One designs to make the sampled output perform well, after converting the plant Laplace transfer function to its equivalent z– transfer function, or equivalently convert the plant differential equation to an equivalent difference equation that has no approximation, i.e. the difference equation solution is exactly the same as the differential equation solution at the sample times. Poles and zeros in the Laplace transfer function are mapped into poles and zeros in the z-transfer function. In the design process, it is important that one wants to know the locations of these zeros, which is studied by References [1], [2], [34] and [35].

The mapping of zeros locations in the s-plane to the z-plane is interesting while the mapping of poles is a simple easily understood manner, each pole in the z-plane is only a function of the location of the pole in the s-plane. Reference [34] studies that the locations

of mapped zeros are not only a function of the locations of the s-plane zeros, but also a function of the poles of the system.

2.2 Types of Zeros

There are two types of zeros when one converts a continuous time transfer function $G(s)$ fed by a zero-order hold to the corresponding discrete time transfer function $G(z)$. When the zero-order hold input is updated at the start of a time step, one should see a change in the output sample at the end of the time step. This means that the most advanced time step in input should be one less than the most advanced time step in output. Thus, new zeros are introduced during the discretization, and these are termed sampling zeros. Reference [1] gives the locations of the zeros introduced by the discretization as the sample time interval T tends to zero, as a function of the pole excess in the original Laplace transfer function, i.e. the numbder of poles minus the number of zeros. These are presented in Table 2.1. They are all on the negative real axis (corresponding to Nyquist frequency). Odd pole excesses introduce an even number of zeros, half of which are inside the unit circle and the other half are outside the unit circle, located at the reciprocals of those inside. Even pole excesses introduce an odd number of zeros, and this extra one is asymptotically located at -1. Note that when two or more zeros have been introduced, at least one zero is outside the unit circle for reasonable sample rates, making a non-minimum phase system in Reference [3].

When there are zeros in the continuous time transfer function $G(s)$, there are images of these zeros in $G(z)$ called intrinsic zeros. Of course the original continuous time system

Table 2.1: Asymptotic zero locations outside and on the unit circle

Pole Excess	Zero Locations
2	-1.0000
3	-3.7321
4	-9.8990, -1.0000
5	-23.2039, -2.3225
6	-51.2184, -4.5419, -1.0000
7	-109.3052, -8.1596, -1.8682
8	-228.5110, -13.9566, -3.1377, -1.0000
9	-471.4075, -23.1360, -4.9566, -1.6447
10	-963.8545, -37.5415, -7.5306, -2.5155, -1.0000
11	-1958.6431, -59.9893, -11.1409, -3.6740, -1.5123

might be non-minimum phase, and then the image of the zero in the right half of the s-plane will be mapped outside the unit circle in the z-plane.

2.3 Instability of Inverse Problems due to NMP Zeros

Very often, discrete time systems have zeros outside the unit circle, i.e. sampling or intrinsic non-minimum phase zeros. This means that the inverse problem is unstable, i.e. finding the input necessary to produce the desired output results in a control action that grows exponentially in magnitude with time. For example, as shown in Table 2.1, a system with pole excess of 3 has a sampling zero at -3.7321 asymptotically as sample rate tends to infinity, then the solution to the homogeneous difference equation consists a constant determined by initial conditions times -3.7321 (using the asymptotic value) to the k^{th} power, where k is the time step number. This solution indicates that control action requited growing exponentially and alternating in sign every time step, and the actuator will hit saturation after not that many time steps. Of course, control system designers would be happy to design systems that produce zero tracking error, but his is prevented by this instability.

Instead, control systems fail to produce the desired output as demonstrated by the achieved system bandwidth.

Chapter 3

Stable Inverse Theorem

3.1 Introduction

For Non-Minimum Phase (NMP) systems, the required inputs found through standard inversion tend to be unbounded as described in the previous chapter, and cannot to be used in practice. There is an existing stable inverse theorem for NMP systems originated from References [3], [4], [5] and [6], which yield bounded inputs for output-tracking problems. Future information of the reference output trajectory is needed for the stable inverse method, which ensures stability by giving up causal characteristic, hence it is non-causal. Non-causal stable inverse is calculated offline given the need of the whole future information, Reference [6] proposed the preview-based stable inverse method in order to calculate the solution online using a finite time window rather than the whole trajectory.

3.2 Stable Inverse Theorem Scheme

The stable inverse theorem as applied to linear systems, can be described as follows

Problem Statement

Consider a Single-Input-Single-Output(SISO) linear system

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{cases} \tag{3.1}$$

where $x(t) \in \mathbb{R}^n$, $u(t) \in \mathbb{R}$ and $y(t) \in \mathbb{R}$ are the state, input and output trajectories. Define the desired reference output trajectory $y_d(t)$ satisfying $y_d^{(i)} \in \mathbb{L}_1 \cap \mathbb{L}_\infty, i = 0, 1, ..., r$ where r is the relative degree of the system, i.e. the smallest positive integer such that $CA^{r-1}B \neq 0$.

Definition in References [3], [4], [5] and [6] For the system Equation 3.1, let the finite time reference trajectory $y_d(t)$, $t \in [t_i, \quad t_f]$ to be tracked, if there exists a nominal input-state-output trajectory $[u_{ref}(t), \quad x_{ref}(t), \quad y_{ref}(t)]$ which

- Satisfies the system:

$$\begin{cases} \dot{x}_{ref}(t) = Ax_{ref}(t) + Bu_{ref}(t) \\ \qquad\qquad\qquad\qquad\qquad\qquad t \in (-\infty, +\infty) \\ y_{ref}(t) = Cu_{ref}(t) \end{cases} \tag{3.2}$$

- Yields the desired output exactly:

$$y_{ref}(t) = y_d(t), \quad t \in [t_i, \ \ t_f] \tag{3.3}$$

- $u_{ref}(t)$, $x_{ref}(t)$, and $y_{ref}(t)$ are bounded for $t \in (-\infty, +\infty)$ and satisfy

$$u_{ref}(t) \to 0, \quad x_{ref}(t) \to 0, \quad y_{ref}(t) \to 0, \quad t \to \pm\infty \tag{3.4}$$

then the input $u_{ref}(t)$ is defined as the stable inverse input of the desired output $y_d(t)$.

Computation

The relative degree of the system Equation 3.1 is r, then keep differentiating the output till the input appears as follows

$$\frac{d^r y(t)}{dt^r} = CA^r x(t) + CA^{r-1}Bu(t) = A_x x(t) + B_y u(t) \tag{3.5}$$

For the inverse problem, the input can be written as

$$u(t) = {B_y}^{-1}\left(y^{(r)}(t) - A_x x(t)\right) \tag{3.6}$$

where B_y is invertible since the well-defined relative degree assumption. In the ideal case, the stable inverse $u(t)$ could be found by substituting $Y^{(r)}(t) = Y_d^{(r)}(t)$ such that the exact output tracking could be achieved. Define the outer state as the function of the output and

its derivatives till $(r-1)^{th}$ order

$$\xi(t) = \begin{bmatrix} y(t) & \frac{dy(t)}{dt} & \cdots & \frac{d^{r-1}y(t)}{dt^{r-1}} \end{bmatrix}^T \tag{3.7}$$

namely, $\xi_i(t) = CA^{i-1}x(t)$, $i = 1, 2, ..., r$. Choose the inner states $\eta(t)$ appropriately, such that the system state is decomposed into outer and inner states through an invertible linear transformation as follows

$$\begin{bmatrix} \xi(t) \\ \eta(t) \end{bmatrix} = T_1 x(t) \tag{3.8}$$

Then the system Equation 3.1 could be rewritten in the new coordinates as

$$\begin{aligned} \dot{\xi}(t) &= \hat{A}_1\xi(t) + \hat{A}_2\eta(t) + \hat{B}_1 u(t) \\ \dot{\eta}(t) &= \hat{A}_3\xi(t) + \hat{A}_4\eta(t) + \hat{B}_2 u(t) \end{aligned} \tag{3.9}$$

where

$$\hat{A} = T_1 A {T_1}^{-1} = \begin{bmatrix} \hat{A}_1 & \hat{A}_2 \\ \hat{A}_3 & \hat{A}_4 \end{bmatrix}, \quad \hat{B} = T_1 B = \begin{bmatrix} \hat{B}_1 \\ \hat{B}_2 \end{bmatrix}, \quad \hat{C} = C {T_1}^{-1} \tag{3.10}$$

Substituting Equation 3.9 to Equation 3.6, get

$$u(t) = {B_y}^{-1}\left(y^{(r)}(t) - A_\xi\xi(t) - A_\eta\eta(t)\right) \tag{3.11}$$

where

$$\begin{bmatrix} A_\xi & A_\eta \end{bmatrix} = A_x T_1^{-1} \tag{3.12}$$

To maintain the output tracking give the desired output $y_d(t)$, the input could be written

$$u(t) = {B_y}^{-1}\left({y_d}^{(r)}(t) - A_\xi \xi_d(t) - A_\eta \eta(t)\right) \tag{3.13}$$

since the outer state $\xi(t)$ is the function of the output only.

Substitue Equation 3.13 to the second equation of Equation 3.9, then one constructs the internal dynamics represented by

$$\dot{\eta}(t) = \hat{A}_\eta \eta(t) + \hat{B}_\eta Y_d(t) \tag{3.14}$$

where

$$\hat{A}_\eta = \hat{A}_4 - \hat{B}_2 {B_y}^{-1} A_\eta, \quad \hat{B}_\eta = \begin{bmatrix} \hat{B}_2 {B_y}^{-1} & \hat{A}_3 - \hat{B}_2 {B_y}^{-1} A_\xi \end{bmatrix}, \quad Y_d = \begin{bmatrix} y_d^{(r)}(t) \\ \xi_d(t) \end{bmatrix} \tag{3.15}$$

[6] stats that finding the inverse input-state trajectory is equivalent to finding bounded solution to the system's internal dynacmis as Equation 3.14. If a bounded solution $\eta_d(t)$ could be found, then the stable inverse control is found through Equation 3.12 by replacing $\eta(t)$ with $\eta_d(t)$. The eigenvalues of the internal dynamic matrix coincide with the zeros of the original system Equation Equation 3.1. There exists a linear transformation T_2 such that the internal dynamics can be decoupled into the stable inner state equation and the unstable

inner state equation

$$\begin{bmatrix} \dot{\eta}_{ds}(t) \\ \dot{\eta}_{du}(t) \end{bmatrix} = \begin{bmatrix} \hat{A}_{\eta s} & 0 \\ 0 & \hat{a}_{\eta u} \end{bmatrix} \begin{bmatrix} \eta_{ds}(t) \\ \eta_{du}(t) \end{bmatrix} + \begin{bmatrix} \hat{B}_{\eta s} \\ \hat{B}_{\eta u} \end{bmatrix} Y_d(t) \tag{3.16}$$

where

$$\begin{bmatrix} \eta_{ds}(t) \\ \eta_{du}(t) \end{bmatrix} = T_2 \eta_d(t), \quad T_2 \hat{B}_\eta = \begin{bmatrix} \hat{B}_{\eta s} \\ \hat{B}_{\eta u} \end{bmatrix}, \quad T_2 \hat{A}_\eta T_2^{-1} = \begin{bmatrix} \hat{A}_{\eta s} & 0 \\ 0 & \hat{A}_{\eta s} \end{bmatrix} \tag{3.17}$$

Then the bounded solution to the internal dynamics can be found by a forward causal integration for the stable inner state part, while a backward non-causal integration for the unstable inner state part

$$\begin{aligned} \eta_{ds}(t) &= \int_{-\infty}^{t} e^{\hat{A}_{\eta s}(t-\tau)} \hat{B}_{\eta s} Y_d(\tau) d\tau \\ \eta_{du}(t) &= -\int_{t}^{\infty} e^{-\hat{A}_{\eta u}(\tau - t)} \hat{B}_{\eta u} Y_d(\tau) d\tau \end{aligned} \tag{3.18}$$

After solving for the internal dynamics, then the stable inverse control solution can be found in these coordinates.

Implementation

Of course one cannot perform these integrals from minus infinity to plus infinity. Instead, start the forward integral from time $(t_i - \delta t_i)$ instead of minus infinity, and start the backward integral from $(t_f + \delta t_f)$. Pick δt_i large enough that it is beyond the settling time of the zeros in the left half plane, considered as poles. Pick δt_f large enough that it is beyond the settling

time of the zeros in the right half plane treated as poles, and going backward in time. As δt_i and δt_f tend to infinity the computation converges to the desired stable inverse solution. When these are not infinite, then the stable inverse obtained produces approximate tracking. How approximate is determined by how far beyond the settling times when these integrals start.

The above can be reformulated for discrete time systems in which case it handles both intrinsic sampling NMP zeros. In place of derivatives of the desired trajectory one uses analogous time shifts. The differentiability requirements of the desired trajectory no longer apply because the action is spread out over steps. Without the specified continuity in the time domain, the discrete time solution may require rather large wiggles. For particularly fast sample rates, these can be beyond the actuator limits, but the mathematics is happy to compute the results.

This page intentionally left blank.

Chapter 4

New Results for Stable Inverses of Discrete Time Systems

4.1 Introduction

Typical feedback control systems do not do what you ask them to do. The concept of bandwidth is created to describe up to what frequency such a system will do something reasonably close to the command. Various control approaches aim to fix this problem, and produce zero tracking error flowing the commanded trajectory. These include Iterative Learning Control (ILC), Repetitive Control (RC), one step ahead control, indirect adaptive control, etc. Each aims to produce that input command that produces the desired output, i.e. solve the inverse problem. To implement such control laws, one must use discrete time models which represent the continuous world with inputs coming through a zero-order hold. Assume a one-time step delay through the system, since the time lag from the change in

input at a given time step to the first time step influenced in the output should be one. Thus, new zeros are introduced during the discretization, and these are termed sampling zeros. When two or more zeros have thus been introduced, at least one zero is out the unit circle for reasonable sample rates, making a non-minimum phase system. The inverse problem makes these zeros into poles, producing an unstable control action. References [1] and [2] tell the asymptotic locations of the zeros introduced as the sample time interval tends to zero, for each value of pole excess (relative degree), i.e. the number of poles minus the number of zeros. Of course, the original continuous time system might be non-minimum phase, and then the image of the zero(s) in the right half of the s-plane will be mapped outside the unit circle in the z-plane. Such zeros are termed intrinsic zeros, and again they make the inverse problem unstable.

There is an existing stable inverse theorem developed to address this problem as presented in the previous chapter. The author and co-workers have developed a series of new stable inverses presented in this chapter address this instability problem in a different way, and in a way that is more practical for implementation. There are seven new stable inverse laws, three for each of the modified problem statements. One factors the matrix P into a product of a matrix or matrices for the zeros outside, times a matrix related to all poles and zeros that are inside the unit circle. These stable inverses are referred to as Longman JiLLL stable inverses, where JiLLL refers to people who have contributed to the results: Xaioqiang Ji, Te Li, Yao Li, and Peter LeVoci. The notation for each stable inverse is Longman JiLLL FI, NI, FS, NS – FI for solution of the initial delete problem factored, NI for not factored, and FS for solution of the skip step problem factored, NS for not factored. And the last stable inverse solution is referred as the ”clean” solution. See References [36],

[16], [37], [18], [38], [39], [40] and [41].

4.2 The System and Its True Inverse

Suppose we are given a continuous time transfer function $G(s)$ or equivalently a state space differential equation, having a zero-order hold input, and we convert it to a discrete time transfer function $G(z)$ or a discrete time difference equation. Consider single-input, single-output systems expressed as n^{th} order difference equation

$$y[k+n]+a_1y[k+(n-1)]+\cdots+a_ny[k] = b_1u[k+(n-1)]+b_2u[k+(n-2)]+\cdots+b_nu[k] \tag{4.1}$$

Or equivalently, with a time backward-shift operator $z^{-1}\{f(k)\} = f(k-1)$, one gets

$$[1 + a_1z^{-1} + \ldots + a_nz^{-n}]\{y(k+n)\} = [b_1 + \ldots + b_nz^{-(n-1)}]\{u(k+n-1)\} \tag{4.2}$$

In the contest of linear discrete time-invariant systems, use of the $z^{-1}\{\cdot\}$ operator and the z-transform variable can often be done interchangeably. We term the roots of the right hand side polynomial in the bracket as the zeros, and correspondingly the roots of the left hand side as poles. Generically, the discretization process will introduce the number of extra zeros needed to produce the power $n-1$ in the right hand polynomial.

Equivalently, the state realization of Equation 4.1 is

$$x(k+1) = Ax(k) + Bu(k); \quad y(k) = Cx(k) \tag{4.3}$$

with

$$A = \begin{bmatrix} 0 & 1 & 0 & \cdots \\ \vdots & 0 & \ddots & \vdots \\ 0 & \vdots & \cdots & 1 \\ -a_n & -a_{n-1} & \cdots & -a_1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} b_n & b_{n-1} & \dots & b_1 \end{bmatrix} \tag{4.4}$$

where $A \in R^{nxn}$, $B \in R^{nx1}$, $C \in R^{1xn}$. By iterating the solution of Equation 4.1 through p time steps, one could package the relationship between the p-step input history and the p-step output history

$$\underline{y} = P\underline{u} + \bar{A}x(0) \tag{4.5}$$

where $\underline{u} = [u(0) \quad u(1) \quad \cdots \quad u(p-1)]^T$, $\underline{y} = [y(1) \quad y(2) \quad \cdots \quad y(p)]^T$, and $x(0)$ is an initial condition, $\bar{A}$ is and observability matrix, and P is a lower triangular Toeplitz matrix

$$P = \begin{bmatrix} CB & 0 & \cdots & 0 \\ CAB & CB & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA^{p-1}B & CA^{p-2}B & \cdots & CB \end{bmatrix} \quad ; \quad \bar{A} = \begin{bmatrix} CA \\ CA^2 \\ \vdots \\ CA^p \end{bmatrix} \tag{4.6}$$

Note that in Equation 4.5, the output can be divided into two parts, the zero state response $P\underline{u}$, and the zero input response $\bar{A}x(0)$.

Then the true unique solution to the inverse problem is given as

$$\underline{u}^* = P^{-1}\left[\underline{y}^* - \bar{A}x_0\right] \tag{4.7}$$

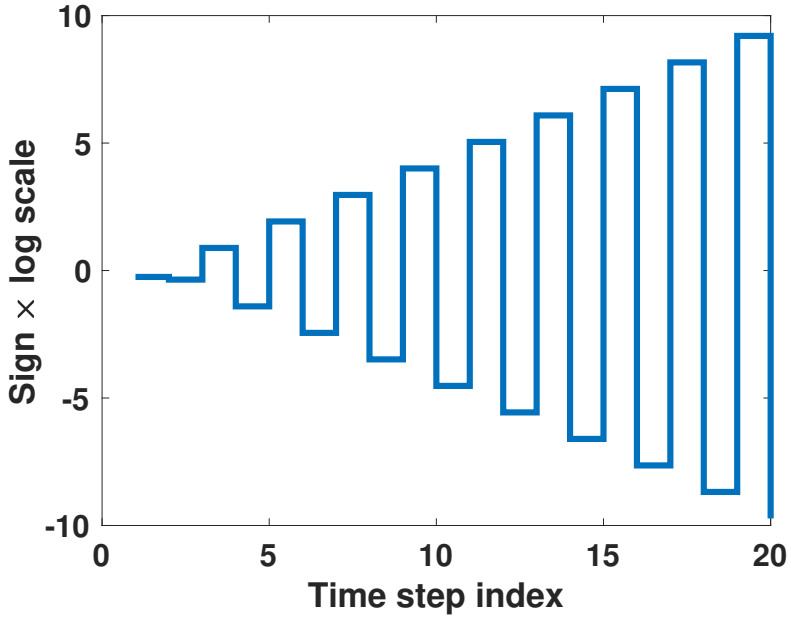

Figure 4.1: Unstable inverse control action

where $\underline{y}^*$ denotes the desired output trajectory. The inverse P^{-1} is guaranteed to exist since all the eigenvalues of matrix P are $CB > 0$. This is nonzero as a result of the assumption of one time-step delay through the system.

For the purpose of illustration, consider a third order continuous time system $G(s)$ with pole excess of three, and it is a reasonable model of the input-output relationship for all axes of a robot at NASA Langley Research Center. Two zeros are introduced during the discretization process. References [1] and [2] tell that one zero is inside and unit circle and the other one is outside, with asymptotical location -3.7321 as the sample time interval tends to zero. Thus, when aim to solve the inverse problem, there is one term in the solution of the corresponding homogeneous equation as $c(-3.7321)^k$, and could be interpreted as that the needed control action grows exponentially alternating sign each time step. This is shown in Figure 4.1 computed for 100Hz sample rate and a simple desired trajectory. The

instability presented here, must lie in P^{-1} matrix. Equation 4.5 gives

$$\|\underline{u}^*\| \leq \left\|P^{-1}\right\| \left\|\underline{y}^*\right\| + \left\|P^{-1}\right\| \left\|\bar{A}x_0\right\| \tag{4.8}$$

where $\|\cdot\|$ denotes the vector norm and the corresponding matrix norm. We know that as the dimension p increases, $\underline{u}^*$ grows exponentially, and this growth must be the result of this inverse matrix. Suppose the system matrix A is diagonalizable as: $A = M\Lambda M^{-1}$, where M is the the square matrix whose columns are eigenvectors of A and Λ is the diagonal matrix whose diagonal elements are the corresponding eigenvalues, i.e. $\Lambda_{ii} = \lambda_i$, $i = 1, 2, ..., n$ and $|\lambda_{\max}| < 1$ because we consider only asymptotically stable systems. Then

$$\begin{aligned}
\left\|\bar{A}x_0\right\| &\leq \sum_{k=1}^{p} \left|CA^k x_0\right| \\
&\leq \sum_{k=1}^{p} \left|CM\Lambda^k M^{-1} x_0\right| = \sum_{k=1}^{p} \left|\bar{C}\Lambda^k x_0\right| \\
&\leq \left(n \max_i \left|\bar{C}_i\right| \cdot \max_i \left|\bar{x}_{0_i}\right|\right) \cdot \sum_{k=1}^{p} \left|\lambda_{\max}\right|^k = \mu \left|\lambda_{\max}\right| \frac{1-|\lambda_{\max}|^p}{1-|\lambda_{\max}|}
\end{aligned} \tag{4.9}$$

It is clear that $\bar{A}x_0$ has an upper bound independent of dimension p. The $\underline{y}^*$ is the norm of the desired output trajectory which we are free to pick so that it avoids any exponential growth with time. If A is not diagonalizable, the use of Jordan form gives us a similar result. We conclude that the instability of the solution in Equation 4.7, i.e. the exponential growth in the required control action producing the desired output trajectory must the matrix inverse norm. We are free to pick the matrix norm as that induced by the Euclidian vector norm, i.e. $\|P^{-1}\| = \sigma_{\min}^{-1}$, where $\sigma_{\min}$ is the least singular value of P matrix. In other words, as the dimension of matrix P increases, $\sigma_{\min}$ decays exponentially.

Properties of Matrix P

The singular value decomposition of matrix $P = USV^T$ has various special properties.

- Reference [42] presents the relationship between singular values and singular vectors of matrix P and the frequency response of the system whose transfer function is denoted by $G(z)$. Its frequency response version is $G(e^{i\omega T}) = M(\omega)e^{i\theta(\omega)}$. For p step long signal $\underline{u}$, the Discrete Fourier Transform (DFT) can be used to find its frequency content

$$\begin{aligned} U\left(e^{i\omega_0 n}\right) &= \sum_{k=1}^{p-1} u(k)\left(e^{i\omega_0 n}\right)^{-k} \\ \omega &= (2\pi/p)\, n = \omega_0 n, \quad n = 0, 1, \ldots, p-1 \end{aligned} \tag{4.10}$$

Define $z_0 = e^{i\omega_0}$, $U_n = U\left(e^{i\omega_0 n}\right) = U\left(z_0^n\right)$, and the DFT matrix is

$$H = \begin{bmatrix} \left(z_o^0\right)^0 & \left(z_o^0\right)^{-1} & \cdots & \left(z_o^0\right)^{-(p-1)} \\ \left(z_o^1\right)^0 & \left(z_o^1\right)^{-1} & \cdots & \left(z_o^1\right)^{-(p-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \left(z_o^{p-1}\right)^0 & \left(z_o^{p-1}\right)^{-1} & \cdots & \left(z_o^{p-1}\right)^{-(p-1)} \end{bmatrix} \tag{4.11}$$

The Inverse DFT is generated using $H^{-1} = (1/p)(H^*)^T$, where the superscript asterisk denotes the complex conjugate. Now, assuming zero initial conditions, $\underline{y} = P\underline{u}$. Multiplying on the left by H on both sides, and inserting $H^{-1}H$ in front of $\underline{u}$ produces

$$\underline{Y} = E\underline{U}; \quad E = (1/p)HP(H^*)^T = \hat{H}P(H^*)^T \tag{4.12}$$

Then E gives the relationship between the frequency components of the input and the frequency components of the output. The H^* represents H with the complex columns and rows normalized to unit length: $\hat{H} = (1/p)H$, $\hat{H}^{-1} = \left(\hat{H}^*\right)^T$. As trajectory gets long, the E converges to the systems frequency response given by

$$E = diag\begin{pmatrix} M_0 e^{i\theta_0} & M_1 e^{i\theta_1} & \dots & M_{p-1} e^{i\theta_{p-1}} \end{pmatrix} \tag{4.13}$$

Since all of the attenuation or amplification information is contained in M_n and Σ, by deleting both, the phase information can be determined as

$$diag\begin{pmatrix} e^{i\theta_0} & e^{i\theta_1} & \dots & e^{i\theta_{p-1}} \end{pmatrix} = \hat{H}UV^T\left(\hat{H}^*\right)^T \tag{4.14}$$

One observes that these singular vectors are close to being sinusoids, and one can identify the frequency by taking the DFT of the vector. Because these vectors also have to handle transients from the zero initial conditions, there are some end effects.

- Steady state frequency response would require two identical singular values for each frequency. Often, the DFT of the singular vectors has a pea with only one frequency in it, every other vector, and the DFT of the next vector has two frequencies in the peak mixing the last frequency and the next frequency.
- Reference [43] shows that the i^{th} input singular vector is equal to plus or minus the i^{th} output singular vector in reverse time order. So the phase of these two vectors has to be chosen so that this process produces the correct phase change for the associated frequency.

- As the number of time steps p is finite, the singular values and singular vectors need not perfectly represent the frequency response, bu the magnitude response remains very good to relatively small values of p. One can create a circulant form of matrix P, by using the first column shifted downward with entries that leave the matrix at the bottom, entering at the top, filling up the upper triangular part of the matrix. The singular value decomposition of this matrix precisely gives the steady state frequency response. It has eliminated the influence of starting from zero initial conditions associated with the original P.

The Signature of Instability in the System Matrix

Figure 4.2 shows the smallest singular value and the next to the smallest singular value of P as a function of the size p of this matrix, i.e. the number of time steps in the desired trajectory. The next to the smallest singular value after the dimension gets above 10 becomes the steady state magnitude frequency response for the highest frequency visible in p time steps of data. The last singular value is not related to frequency response. Instead it has the slope shown by the solid line, given by the reciprocal of the absolute value of the outside zero location to power p. Note that after dimension about 35, Matlab is no longer able to compute the singular value due to word length limitations. One can use the slope to extrapolate to find the true value.

Figure 4.3 shows all singular values of matrix P for the 3^{rd} order problem sampled at 100Hz. Note that there is a singular value that is very small near the bottom right corner of the plot. This corresponds to the zero location outside the unit circle. The remaining

singular values that use the same symbol are shown and compared to the solid at the discrete frequencies one can see in p time steps. Notice that there is very little difference between theses, so that one can recognize the singular value related to the zero location.

Figure 4.4 shows the output singular vector in U associated with the zero. The plot shows the absolute value of the vector components versus time step, on a dB scale. This vector is associated with early time steps in the output. Note that the decay of these components with time, match the decay of the reciprocal of the absolute value of the zero location to the power of the time step. Figure 4.5 shows the corresponding input singular vector which grows with the slope of the absolute value of the zero location to the power of the time step. This means that to fix an error component on the output singular vector that is near the beginning of the trajectory, one needs to apply a control that can be small at the start but grows exponentially with time. This is the unstable inverse, seen in singular value space. Again, notice that Matlab has numerical error from the finite word length that prevents accurate computation of these components when the values reach a numerical zero. These plots give clean examples. For a 7^{th} order pole excess there will be 3 zeros outside, and Matlab can distinguish the individual slopes only up to a matrix size of about 10 by 10, and after that strange distortions occur.

Conclution: The signature in the P matrix of one zero outside the unit circle, with corresponding instability of the inverse system, is (1) A linear decay on a log scale of a singular value as a function of matrix size p. (2) A corresponding pair of input and output singular vectors which have opposite slopes, with the input singular vector growing linearly with time step on a log scale. If we find a system "inverse" with the property that it produces zero error at the time steps addressed, and has no singular values and singular vector pair of

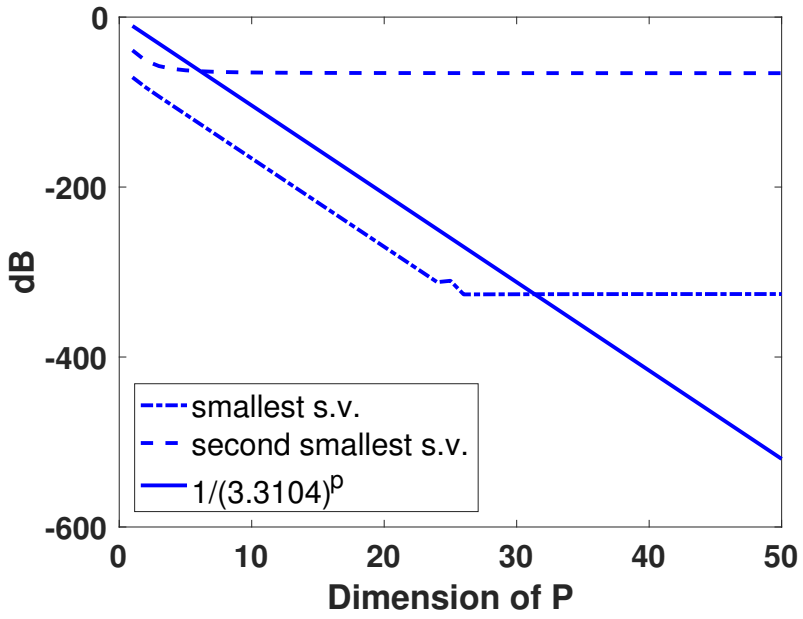

Figure 4.2: The smallest singular value and the nest to smallest singular value as a function of the dimension of p

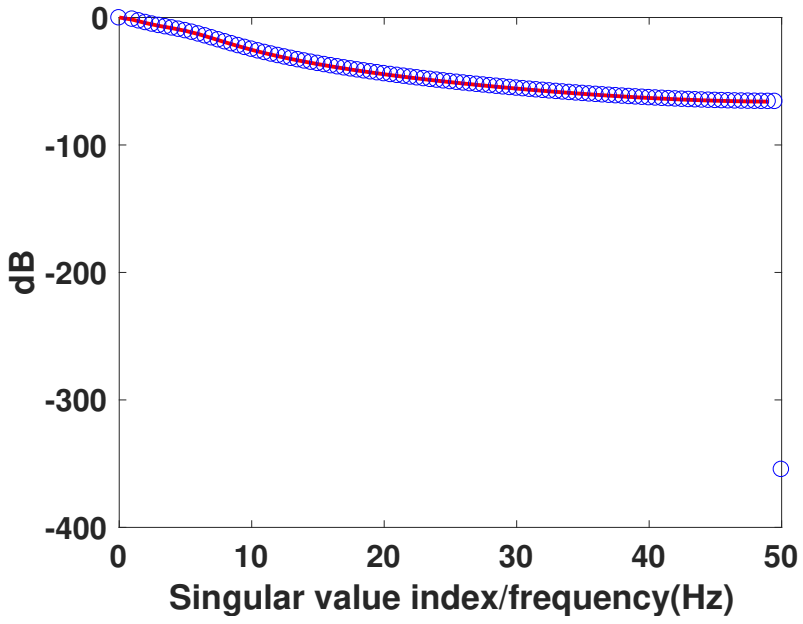

Figure 4.3: The singular values of matrix P compared to discrete magnitude frequency response

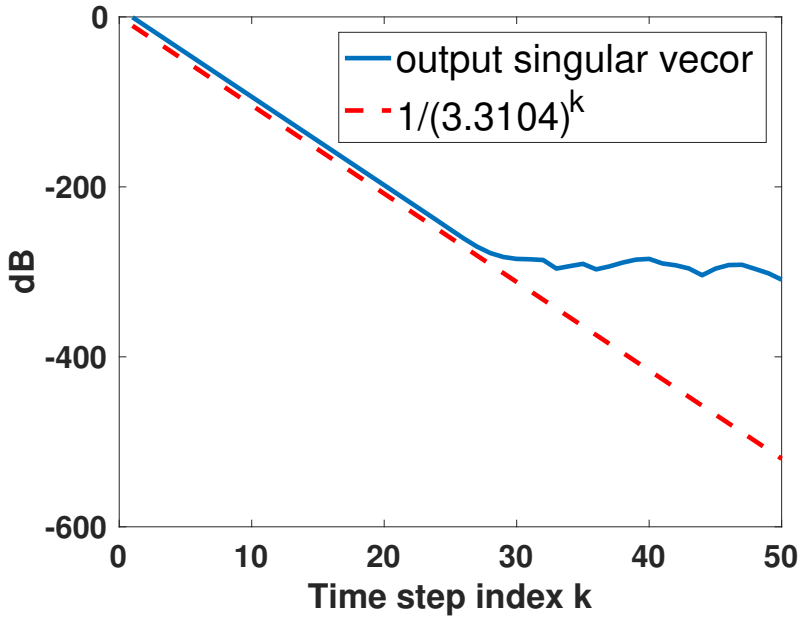

Figure 4.4: Magnitudes of the components of last output singular vector of P, also shown magnitudes of reciprocal of zero and zero location to the k^{th} power with pole excess 3

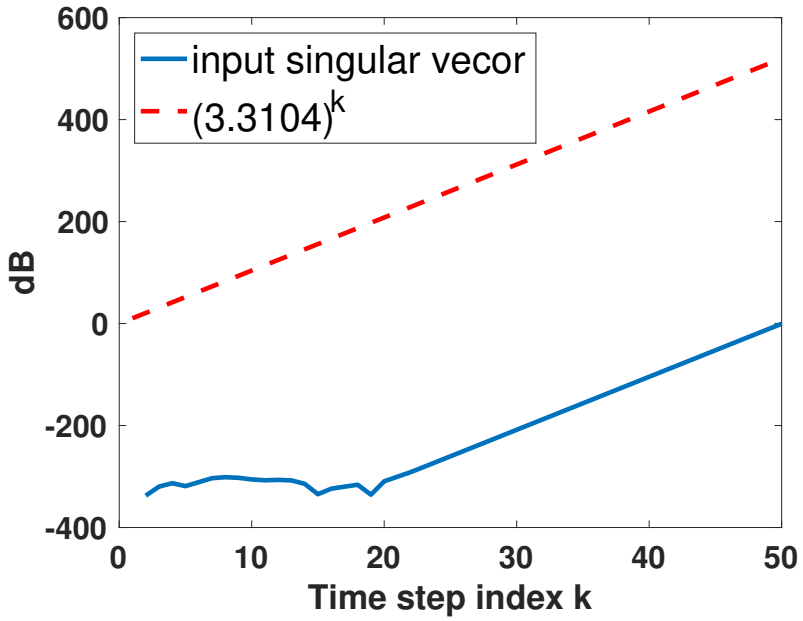

Figure 4.5: Magnitudes of the components of last input singular vector of P, also shown magnitudes of reciprocal of zero and zero location to the k^{th} power with pole excess 3

the above kind, then we have generated a zero error stable inverse to the unstable inverse problem.

4.3 New Stable Inverses

We can consider two kinds of inverse problems. (1) Given a desired p time-step trajectory starting from a given initial condition, find the needed input p time-step history to produce this output. This is a batch computation made before running the control system. (2) The second version tries to address the problem of making a control system give perfect tracking in real time to whatever command one wants to make in the next time step.

Since the actual solution to (1) for large classes of systems gives an unstable control, something must be relaxed about the statement of the problem in order to produce a stable inverse. What aspects are relaxed is different for the different stable inverses discussed here. Regarding (2), some discussion is given of different approaches trying to make steps toward somehow using the solution for (1) to address (2).

The new stable inverse results described here, define the stable inverse problem differently. All problems posed in discrete time. The emphasis is on addressing the non-minimum phase zeros introduced by discretization, but the methods developed work on intrinsic zeros as well (with some differences in the properties of the solutions regarding relationship to frequency response).

The Problem Statement

- **Initial Problem Statement.** Given a SISO discrete time system, given the initial condition $x(0)$, and given a desired p time-step output history $y^*(k)$ for $k = 1, 2, ..., p$, find the input history $u(k)$ for $k = 0, 1, ..., p-1$ that will produce this output. The time delay through the system is assumed to be one time step, simple modifications treat other delay values.
- **Modified Problem Statement No.1 (Initial Delete).** Find the input history $u(k)$ for $k = 0, 1, ..., p-1$ that produces desired output history for $k = 1 + n_o, 2, ..., p$. The number n_o is the number of zeros of the discrete time transfer function outside the unit circle. The first modified problem statement is called initial deletion method, and the second below is called the skip step method. One way to consider the second method is: given a desired trajectory p steps long, in the case of a single zero outside the unit circle, double the sample rate but only ask for zero error at the original sample times. Control updates are made every time step at the doubled rate. If the number of zeros outside is 2, then introduce two sample times between each of the original sample times. One can think of this as creating a generalized hold that involves one or more extra zero order holds each times step.
- **Modified Problem Statement No.2 (Skip Step).** Let $p = (n_o + 1)p^*$ where p^* is the number of original time steps, and p is the number of time steps after introducing n_o steps between each of the original time steps. Find input history at all time steps $u(k)$ for $k = 0, 1, ..., p-1$, to produce zero error at the original time steps, $y^*(k(n_o + 1))$ for $k = 1, 2, ..., p^*$.

A Factorization of the System Maitrx P

In order to develop the stable inverses, we first create a factorization of the system matrix

$$P = P_O P_I \tag{4.15}$$

where the P_I matrix models all dynamics of zeros inside the unit circle in and all of the poles of Equation 4.1, since it is assumed asymptotically stable. Matrix P_O models all zeros outside, which can be sampling zeros, intrinsic zeros, or both.

The Simplest Case: One Zero Outside the Unit Circle To understand the nature of these matrices, consider a 3^{rd} order discrete time system, which is the simplest system for which one encounters the problem of an unstable inverse due to sampling zeros,

$$y(k+3) + a_1 y(k+2) + a_2 y(k+1) + a_3 y(k) = b_1 u(k+2) + b_2 u(k+1) + b_3 u(k) \tag{4.16}$$

On the right-hand side, two sampling zeros have been introduced during the discretization process. [1],[2] indicates that one zero is inside the unit circle and the other one is outside for reasonable sample rates, with asymptotic locations -3.7321 and its inverse as the sample time interval tends to zero. One can convert Equation 4.16 to its equivalent state realization, and find the corresponding P matrix where

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_3 & -a_2 & -a_1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} b_3 & b_2 & b_1 \end{bmatrix} \tag{4.17}$$

Write Equation 4.16 in the above operator form so that the zero's polynomial can be factored

$$\begin{aligned}[1 + a_1 z^{-1} + a_2 z^{-2} + a_3 z^{-3}]\{y(k+3)\} &= [b_1 + b_2 z^{-1} + b_3 z^{-2}]\{u(k+2)\} \\ &= b_1[(1 - z_O z^{-1})(1 - z_I z^{-1})]\{u(k+2)\}\end{aligned} \tag{4.18}$$

where z_O denotes the zero outside the unit circle, i.e. with magnitude greater than one, and z_I denotes the zero outside. Note that $b_2 = -b_1(z_I + z_O)$ and $b_3 = b_1 z_O z_I$.

To produce the P_I matrix that models the zero inside and all poles, while matching the coefficients of $u(k)$, we define difference equation

$$\left[z^3 + a_1 z^2 + a_2 z^1 + a_3\right] \left\{\underline{y}_I(k)\right\} = b_1(-z_O)(z - z_I)\left\{u(k)\right\} \tag{4.19}$$

where $\underline{y}_I$ is an intermediate artificial output. To convert this difference equation to state-space form, again define an intermediate variable $\bar{y}_I(k)$ which is the solution of

$$\left[z^3 + a_1 z^2 + a_2 z^1 + a_3\right] \left\{\bar{y}_I(k)\right\} = u(k) \tag{4.20}$$

This is converted to state space form using the state definition

$$x_I(k) = \begin{bmatrix} \bar{y}_I(k) \\ \bar{y}_I(k+1) \\ \bar{y}_I(k+2) \end{bmatrix} \tag{4.21}$$

Then the solution to difference Equation 4.19 is written in terms of these state variables by

superposition

$$y_I(k) = b_1(-z_O)\bar{y}_I(k+1) + b_3\bar{y}_I(k) \tag{4.22}$$

Note for the future use, that the actual output is

$$y(k) = b_1\bar{y}_I(k+2) + b_2\bar{y}_I(k+1) + b_3\bar{y}_I(k) \tag{4.23}$$

Hence, the state space realization of Equation 4.19 is

$$A_I = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a_3 & -a_2 & -a_1 \end{bmatrix}, \quad B_I = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad C_I = \begin{bmatrix} b_3 & b_1(-z_O) & 0 \end{bmatrix} \tag{4.24}$$

This differs from the original P matrix for the system which has one time-step delay with diagonal terms CB nonzero. From Equation 4.16 the P_I matrix must have two time steps delay meaning that C_IB_I is zero (which can be checked using Equation 4.23). Hence,

$$P_I = \begin{bmatrix} C_IA_IB_I & 0 & \dots & 0 \\ C_IA_I{}^2B_I & C_IA_IB_I & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ C_IA_I{}^pB_I & \dots & C_IA_I{}^2B_I & C_IA_IB_I \end{bmatrix} \tag{4.25}$$

and the inverse of P_I is guaranteed to exist.

Compare Equation 4.18 and Equation 4.19, by superposition one has

$$y(k) = -(1/z_O)(z - z_O)\left\{y_I(k)\right\}, \quad y(k) = y_I(k) - (1/z_O)y_I(k+1) \tag{4.26}$$

In matrix form,

$$\underline{y} = P_O \underline{y}_I, \quad P_O = \begin{bmatrix} -1/z_O & & 0 \\ 1 & \ddots & \\ 0 & \ddots & -1/z_O \end{bmatrix} \tag{4.27}$$

where P_O is a bi-diagonal Toeplitz matrix, $\underline{y}$ and $\underline{y}_I$ are full time histories of sequences. Before proceeding, observe the relationship between initial conditions $x_O = \begin{bmatrix} y(-2) & y(-1) & y(0) & u(-2) & u(-1) \end{bmatrix}^T$ of $y(k)$ for Equation 4.18 that are prescribed for the original physical system, and the initial conditions $x_{IO} = \begin{bmatrix} y_I(-2) & y_I(-1) & y_I(0) & u(-2) \end{bmatrix}^T$ of $y_I(k)$ in Equation 4.19 as related to x_O, and $\bar{x}_{IO} = \begin{bmatrix} \bar{y}_I(-2) & \bar{y}_I(-1) & \bar{y}_I(0) \end{bmatrix}^T$ of $\bar{y}_I(k)$ in Equation 4.20 also related to x_O. Combining Equation 4.21 and Equation 4.23, setting $k = 0, 1, 2, ...$ and solving recursively

for $\bar{y}_I(1)$ and $\bar{y}_I(2)$ in Equation 4.20, produced the relationship between $\bar{x}_{IO}$ and x_O as

$$
\begin{aligned}
&\bar{x}_{I0} = {}_{\bar{X}_I}T_X \cdot x_0 \\
&{}_{\bar{X}_I}T_X = \tilde{T}_1^{-1}\tilde{T}_2 \\
&\tilde{T}_1 = \begin{bmatrix} b_3 & b_2 & b_1 \\ -a_3b_1 & b_3 - a_2b_1 & b_2 - a_1b_1 \\ b_1a_1a_3 - a_3b_2 & b_1a_1a_2 - a_3b_1 - a_2b_2 & b_1a_1^2 - a_2b_1 - a_1b_2 + b_3 \end{bmatrix} \\
&\tilde{T}_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -b_1 & 0 \\ 0 & 0 & 1 & a_1b_1 - b_2 & -b_1 \end{bmatrix}
\end{aligned}
\tag{4.28}
$$

Note from the right hand side of this equation, that the initial conditions $y(-1)$ and $u(-2)$ are interchangeable, as well as the initial conditions $y(0)$, $u(-2)$ and $u(-1)$. Similarly, one could relate $\bar{x}_{IO}$ and x_{IO} as

$$
\begin{aligned}
&\bar{x}_{IO} = {}_{\bar{X}_I}T_{x_I} \cdot x_{I0} \\
&{}_{\bar{X}_I}T_{x_I} = \begin{bmatrix} b_3 & b_1(-z_O) & 0 \\ 0 & b_3 & b_1(-z_O) \\ b_1(-z_O)a_3 & b_1(-z_O)a_2 & b_3 + b_1z_Oa_1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & b_1z_O \\ 0 & 0 & 1 & 0 \end{bmatrix}
\end{aligned}
\tag{4.29}
$$

With the relations above, one could always set the corresponding initial conditions, given the initial conditions prescribed by the original physical problem.

General Case: More Than One Zero Outside the Unit Circle Consider a general n^{th} order system as in Equation 4.1 and Equation 4.2. Among the $n-1$ zeros of the right

hand side polynomial, let the numbder of zeros inside the unit circle be m, then the number of zeros outside is $n-1-m$, including sampling zeros, intrinsic zeros, or both. We do not consider the case of zeros on the unit circle. We now generalize the factorization of matrix P to handle multiple zeros outside the unit circle.

- Factor the polynomials in the right hand side of Equation 4.2 grouped into factors with zeros outside, and factors with zeros inside

$$\left[z^n + a_1 z^{n-1} + \ldots + a_n\right]\{y(k)\} = b_1 \prod_{i=1}^{m}(z - z_{O\_i}) \prod_{i=1}^{n-1-m}(z - z_{I\_i})\{u(k)\} \tag{4.30}$$

- Define an artificial intermediate output y_I of the system matrix P_I modeling dynamics of all zeros inside the unit circle and all poles. And keep the coefficients of $u(k)$ unchanged,

$$\begin{aligned} & z^n + a_1 z^{n-1} + \ldots + a_n]\{y_I(k)\} \\ & = b_1 \prod_{i=1}^{m}(-z_{O\_i}) \prod_{i=1}^{n-1-m}(z - z_{I\_i})\{u(k)\} \\ & = c_{m+1}u[k + (n-m-1)] + \ldots + c_{n-1}u[k+1] + b_n u[k] \end{aligned} \tag{4.31}$$

This is equivalently expressed in difference equation from as

$$\begin{aligned} & y[k+n] + a_1 y[k + (n-1)] + \ldots + a_n y[k] \\ & = c_{m+1}u[k + (n-m-1)] + \ldots + c_{n-1}u[k+1] + b_n u[k] \end{aligned} \tag{4.32}$$

The P_I matrix relating $\underline{y}_I = P_I \underline{u}$ is

$$P_I = \begin{bmatrix} C_I A_I^m B_I & 0 & \dots & 0 \\ C_I A_I^{m+1} B_I & C_I A_I^m B_I & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ C_I A_I^{p+m-1} B_I & \dots & C_I A_I^{m+1} B_I & C_I A_I^m B_I \end{bmatrix} \tag{4.33}$$

where

$$A_I = \begin{bmatrix} 0 & 1 & 0 & \cdots \\ \vdots & 0 & \ddots & \vdots \\ 0 & \vdots & \cdots & 1 \\ -a_n & -a_{n-1} & \cdots & -a_1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}, \tag{4.34}$$

$$C_I = \begin{bmatrix} b_n & \cdots & c_{m+1} & 0 & \cdots \end{bmatrix}$$

- To develop P_O, from Equation 4.30 and Equation 4.31, write

$$y(k) = \frac{\prod_{i=1}^{m} (z - z_{O\_i})}{\prod_{i=1}^{m} (-z_{O\_i})} \{y_I(k)\} = \prod_{i=1}^{m} (-\frac{1}{z_{O\_i}} z + 1) \{y_I(k)\} \tag{4.35}$$

In matrix form,

$$P_O = \prod_{i=1}^{m} P_{O\_i}$$

$$= \prod_{i=1}^{m} \begin{bmatrix} -\frac{1}{z_{O\_i}} & & 0 \\ 1 & \ddots & \\ 0 & \ddots & -\frac{1}{z_{O\_i}} \end{bmatrix} = \begin{bmatrix} d_1 & & & & & 0 \\ \vdots & \ddots & & & & \\ d_m & \ddots & \ddots & & & \\ 1 & \ddots & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \ddots & \\ 0 & & 1 & d_m & \cdots & d_1 \end{bmatrix} \in C^{p \times p} \tag{4.36}$$

As a result, P_O is the matrix product of a number of bi-diagonal Toeplitz matrices, one for each of the m zeros outside the unit circle, each P_{Oi} models the dynamics of the corresponding i^{th} zero, which could be complex or real.

Of course, the inverse $P_{\bar{I}}^{-1}$ exists and does not have any instability associated with it. This then can cancel everything inside the unit circle, and it is the treatment of the zeros outside that must handle the issue of creating a stable inverse. Note that this section established the existence of the factorization. Note that one may compute the factor for the zeros outside from the known P and P_I, using $P_O = P P_{\bar{I}}^{-1}$.

The Stable Inverse Law for Initial Delete using Factored Form (Longman-JiLLL FI)

This stable inverse address the following class of problems. The discrete time system of the form of Equation 4.1, Equation 4.2 is an asymptotically stable single input, single output system, with $n-1$ zeros, $m>0$ of them are outside the unit circle and there is no zero on the unit circle. This means that the inverse problem for zero error at all time steps produces an unstable control action. The zeros outside can be sampling zeros or intrinsic zeros, or both, and can be either real or complex.

Instability of the control action is immediately known from the right hand side of Equation 4.2 when there is a zero outside the unit circle. The solution of the homogeneous equation that must be solved for the control contains a term that is a constant times the zero location to the power of the time step. In the inverse of the Toeplitz matrix the instability can be observed in several ways as discussed in Reference [38], but the most obvious and fundamental is the change in the minimum singular value of the matrix P as the dimension of P increases by one. As discussed in relation to Equation 4.8, the inverse must have a singular value that increases every time the matrix dimension is increase. If this increase in singular value with dimension of P is dliminated, then the instability of the inverse has been addressed.

For a system with m zeros outside the unit circle, the stable inverse method presented this section will produce a stable inverse that produced zero error at all time steps except for the first m time step. The error at these time steps is determined by a pseudo-inverse of an underspecified set of equations, with corresponding minimum norm properties of the

appropriate variable history.

Computation Produce the factorization $P = \prod_{i=1}^{m} P_{O\_i} P_I$ where the index number i starts from 1 at the right. Delete the first i initial rows and $i-1$ initial columns in P_{Oi} matrix to produce P_{Od_i}, and delete the same number m rows from the beginning of the desired trajectory to form $\underline{y}_d^*$. The the stable inverse is given by

$$\underline{u} = P_I^{-1} \prod_{i=1}^{m} P_{Od\_i}^{\dagger} \left[\underline{y}_d^* - \left(\bar{A} x(0) \right)_d \right] \tag{4.37}$$

where the index number i from 1 at the left and $P_{Od\_i}^{\dagger}$ is the Moore-Penrose pseudo-inverse of P_{Od_i}.

- **Case** $m = 1$**:** For $m = 1$, P_O is of the form as in Equation 4.35. Then

$$P_{Od} = \begin{bmatrix} 1 & -1/z_O & & 0 \\ & \ddots & \ddots & \\ 0 & & 1 & -1/z_O \end{bmatrix} \in C^{(p-1)\times p} \tag{4.38}$$

Take the product of P_{Od} with its conjugate transpose P_{Od}^H to form a tri-diagonal Toeplitz matrix P_T

$$P_T = P_{Od} P_{Od}^H = \begin{bmatrix} 1 + 1/z_O z_O^* & -1/z_O^* & & 0 \\ -1/z_O^* & \ddots & \ddots & \\ & \ddots & \ddots & -1/z_O^* \\ 0 & & -1/z_O^* & 1 + 1/z_O z_O^* \end{bmatrix} \in C^{(p-1)\times(p-1)} \tag{4.39}$$

Reference [44] presents the eigenvalues and associated eigenvectors for such a tri-diagonal matrix, and has the potential to be generalized by [45], [46], and [47]. Here we sketch the calculation. Let λ represent an eigenvalue of P_T and q the corresponding eigenvector with components $q = \begin{bmatrix} q_1 & q_2 & \dots & q_{p-1} \end{bmatrix}^T$. From the definition of the eigenvalue problem $P_T q = \lambda q$, we could form a system of equations and combine into one single difference equation by setting $q_0 = q_p = 0$.

$$-\frac{1}{z_O^*}q_{k-1} + \left(1 + \frac{1}{z_O z_O^*} - \lambda\right) q_k + \left(-\frac{1}{z_O}\right) q_{k+1} = 0, \quad k = 1, ..., p-1 \quad (4.40)$$

The solution to Equation 4.40 is $q_k = C_1 m_1^k + C_2 m_2^k$ where m_1, m_2 are roots of $\left(-\frac{1}{z_O}\right) m^2 + \left(1 + \frac{1}{z_O z_O^*} - \lambda\right) m + \left(-\frac{1}{z_O^*}\right) = 0$ and it can be proven that these two cannot be equal. C_1 and C_2 are arbitrary constants satisfying the boundary conditions $C_1 + C_2 = 0$ and $C_1 m_1^p + C_2 m_2^p = 0$. Use these equalities to get the relation $\left(\frac{m_1}{m_2}\right)^p = 1$ and knowing the product $m_1 m_2 = 1$, we solve for roots

$$m_1 = e^{ij\pi/p}, \quad m_2 = e^{-ij\pi/p}, \quad j = 1, 2, ..., p-1 \quad (4.41)$$

The relation between roots m_1, m_2 and λ seen from Equation 4.40 produces the closed from of $p-1$ eigenvalues

$$\lambda_j = \left(1 + |z_O|^{-2}\right) + 2|z_O|^{-1} \cos\frac{j\pi}{p}, \quad p = 1, 2, ..., p-1 \quad (4.42)$$

and it is clear that all eigenvalues of P_T are bounded by

$$\left(1-|z_O|^{-1}\right)^2 < \lambda_j < \left(1+|z_O|^{-1}\right)^2, \quad j=1,2,...,p-1 \tag{4.43}$$

It is well known that the singular values σ_j of P_{Od} and the eigenvalues λ_j of P_T (the product of P_{Od} and its conjugate transpose) are related by $\sigma_j = \sqrt{\lambda_j}$. Hence, all the singular values of P_{Od} are bounded as

$$\left(1-|z_O|^{-1}\right) < \sigma_j < \left(1+|z_O|^{-1}\right), \quad j=1,2,...,p-1 \tag{4.44}$$

and both lower and upper bounds are independent of the dimension p of matrix P. Rearrange Equation 4.34 and $\underline{\hat{u}} = P_I u$, to produce

$$\underline{\hat{u}} = P_{Od}^{\dagger}\left[\underline{y}_d^* - \left(\bar{A}x(0)\right)_d\right] \tag{4.45}$$

Then $\underline{\hat{u}}$ is bounded according to

$$\begin{aligned}
\|\underline{\hat{u}}\| &= \left\| P_{Od}^{\dagger}\left[\underline{y}^* - \left(\bar{A}x(0)\right)_d\right]\right\| \\
&\leq \left\|P_{Od}^{\dagger}\right\| \left\|\underline{y}_d^*\right\| + \left\|P_{Od}^{\dagger}\right\| \left\|\left(\bar{A}x(0)\right)_d\right\| = \left|\sigma_{\min}^{-1}\right| \left(\left\|\underline{y}_d^*\right\| + \left\|\left(\bar{A}x(0)\right)_d\right\|\right) \\
&< \left(1-\tfrac{1}{|z_O|}\right)\left(\left\|\underline{y}_d^*\right\| + \left\|\left(\bar{A}x(0)\right)_d\right\|\right)
\end{aligned} \tag{4.46}$$

Then Bounded-Input-Bounded-Output stability theorem for asymptotically stable time invariant systems shows that $\|\underline{u}\|$ is bounded as well. There is no exponential growth with dimension.

- **Case: for any** m To generalize the above analysis, simply use

$$\|\hat{\underline{u}}\| \leq \prod_{i=1}^{m} \left\| P^{\dagger}_{Od\_i} \right\| \left(\left\| \underline{y}^{*}_{d} \right\| + \left\| \left(\bar{A}x(0) \right)_{d} \right\| \right) \tag{4.47}$$

  where the norms $\left\| P^{\dagger}_{Od\_i} \right\|$ for the matrices with deleted entries as described above, all have upper bounds. Hence, $\|\underline{u}\|$ is bounded, and the stable inverse presented in Equation 4.37 is proven to work in general.

- **Examine the result of overestimation NMP zeros:** For each P_{O_i}, if the number of initial rows and columns deleted are greater than i and $i-1$ respectively, $\|\hat{\underline{u}}\|$ decreases.

The Stable Inverse for Skip Step using Factored Form (Longman-JiLLL FS)

The stable inverse method presented in this section considers systems starting from continuous time and fed by a zero order hold. The approach addresses the same class of problems as described above having m zeros outside the unit circle in discrete time. This inverse suggests that you start with a given sample rate for which you want to have zero error following a desired trajectory at each time step. Then increase the sample rate by introducing m extra time steps between each of the original time steps. One makes use of the continuous to discrete conversion at the higher sample rate. For the control action there are m new sample times between the initial condition time step, and the first of the original output sample times. These m extra time steps with their extra control inputs between each time step for which one seeks zero tracking error, can be thought of as a form of generalized

hold.

Computation Using the state space model of the system at the faster sample rate, form the Toeplitz matrix P. Then create the factorization $P = P_O P_I$. Remove the rows of P_O matrix so that only the rows associated with the original sample times remain and denote the result with subscript d. And remove the same rows of the desired trajectory to form $\underline{y}^*_d$. Then the stable inverse is given as

$$\underline{u} = P_I^{-1} P_{Oa}^{\dagger} \left[\underline{y}^*_a - \left(\bar{A} x(0) \right)_a \right] \tag{4.48}$$

where $P_{Oa}^{\dagger}$ is the Moore-Penrose pseudo-inverse of P_{Od}. Note that for JiLLL FS one uses the pseudo-inverse of the product of matrices given by P_{Od}, in contrast the what was done in JiLLL FI that used a product of pseudo-inverses of matrices for each zero separately.

Proof The deletion of the rows of P_O using the expression in Equation 4.36 gives

$$P_{Oa} = \begin{bmatrix} 1 & d_m & \dots & d_1 & & & & & & 0 \\ & & & & 1 & d_m & \dots & d_1 & & \\ & & & & & & \ddots & \ddots & \ddots & \ddots \\ 0 & & & & & & & 1 & d_m & \dots & d_1 \end{bmatrix} \in C^{\frac{p}{(m+1)} \times p} \tag{4.49}$$

Define P_B as the product of P_{Oa} and its conjugate transpose

$$P_B = P_{Oa} P_{Oa}^H = \left(1 + \sum_{i=1}^{m} |d_i|^2 \right) I \in R^{\frac{p}{(m+1)} \times \frac{p}{(m+1)}} \tag{4.50}$$

where $1+\sum_{i=1}^{m}|d_i|^2$ are the eigenvalues of P_B. Then $\left(1+\sum_{i=1}^{m}|d_i|^2\right)^{1/2} > 1$ are the singular values of P_{Oa}, i.e. $\sigma_{\min} > 1$. Using $\hat{\underline{u}}$ in the form of Equation 4.45 with new upper bound of $\left\|P_{Od}^{\dagger}\right\| = \sigma_{\min}^{-1} < 1$, established that $\|\hat{\underline{u}}\|$ is bounded, indicating that the control action $\|\underline{u}\|$ is bounded.

The Stable Inverse Law for Initial Delete and Skip Step using Non-Factored Form (Longman-JiLLL NI, NS)

Stable inverse FI given by Equation 4.37 was obtained using the pseudoinverse, and analogously for stable inverse FS with Equation 4.48. The resulting control action minimizes $\|P_I\underline{u}\|_2$ for the corresponding deletions. Again consider one zero outside for simplicity.

Now consider Equation 4.15 for the two different kinds of deletions. The NI and NS stable inversion laws are given by

$$\underline{u} = P_d^{\dagger}\left[\underline{y}_d^* - \left(\bar{A}x(0)\right)_d\right] \tag{4.51}$$

For NI only one initial row is deleted from P to form $P_d^{\dagger}$, and for NS all odd rows are deleted.

Given the extra control variable(s) available in each case compared to the number of equations to be satisfied in Equation 4.15, the pseudoinverse finds that solution that minimizes $\|\underline{u}\|_2$. This is enough to make the needed conclusion, but to make it evident consider the following.

Write the SVD of P_d

$$P_d = U \begin{bmatrix} S & 0 \end{bmatrix} V^T = U \begin{bmatrix} S & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_1^T \end{bmatrix} = USV_1^T, \quad P = V_1 S^{-1} U^T \tag{4.52}$$

using locally defined U, S, and V. Define $\underline{c} = \left[\underline{y}_d^* - \left(\bar{A}x(0)\right)_d\right]$ so the problem considered is $P_d\underline{u} = \underline{c}$. Write this using the decomposition above

$$U \begin{bmatrix} S & 0 \end{bmatrix} \begin{bmatrix} V_1^T \\ V_2^T \end{bmatrix} u = \underline{c}, \quad \begin{bmatrix} S & 0 \end{bmatrix} \begin{bmatrix} V_1^T\underline{u} \\ V_2^T\underline{u} \end{bmatrix} = U^T\underline{c} \tag{4.53}$$

or $V_1^T\underline{u} = S^{-1}U^T\underline{c}$, $V_2^T\underline{u} = \underline{\gamma}$, where $\underline{\gamma}$ can be any vector. Then

$$\underline{u} = \begin{bmatrix} V_1 & V_2 \end{bmatrix} \begin{bmatrix} S^{-1}U^Tc \\ \underline{\gamma} \end{bmatrix} = V_1S^{-1}U^T\underline{c} + V_2\underline{\gamma} \tag{4.54}$$

or

$$\|\underline{u}\|_2^2 = \left[V_1S^{-1}U^T\underline{c} + V_2\underline{\gamma}\right]^T \left[V_1S^{-1}U^T\underline{c} + V_2\underline{\gamma}\right] = \underline{c}^T\left(P_d^\dagger\right)^T P_d^\dagger\underline{c} + \underline{\gamma}^T\underline{\gamma} \tag{4.55}$$

By considering all possible $\underline{\gamma}$, one obtains all possible solutions of the underdetermined set of equations. The $\underline{\gamma}'$s are components on V_2 while the first term are components on V_1. The columns vectors in each are mutually orthogonal. Therefore the minimum norm of $\underline{u}$ is obtained when $\underline{\gamma}$ is the zero vector, i.e. when using the Moore Penrose pseudoinverse in Equation 4.52.

Our objective is to have a stable inverse solution for NI and NS. Equation 4.51 says $\underline{\gamma}$ equal to the zero vector gives the minimum possible Euclidean norm of control $\underline{u}$ to satisfy for both initial and skip step deletion. Solutions FI and FS from factorization therefore, have a nonzero $\underline{\gamma}$ while NI and NS have a zero value of $\underline{\gamma}$. The stable inverse control action for FI and FS were bounded independent of the value of p and satisfied the stable inverse conditions for the matrix model. Since the NI and NS control actions are smaller than FI and FS, the NI and NS must also create stable inverses.

Motivation and Comparison of the Factored and Non-Factored Forms: Consider the Initial Delete method. Conside the desired trajectory $y^*(t) = 0.25[1 - \cos(\pi t)]^2$, one second long, sampled at 50 Hz. Since FI and NI produce zero error for all time steps except the first (for $n_0 = 1$), the first equation one might ask is, how different is the control action and the output at this time step. The answer is, they are the same to computational accuracy, 16 digits. The difference between the two results $\underline{\gamma}$ which is the scalar $\gamma = -5 \times 10^{-4}$ for $G(s) = \left(\frac{a}{s+a}\right)\left(\frac{\omega_n^2}{s^2+2\xi\omega_n+\omega_n^2}\right)$ sampled at 50Hz with $a = 1.4$, $\xi = 0.5$, $\omega_n = 27$. This γ multiplies column vector V_2 of P_d. This is the same except for the sign as the corresponding column vector for P. Figure 4.4 plots the absolute value of each time step k of V_2. The initial straight line plotted 2.9^{p-k}. Clearly Matlab is unable to compute initial steps, but they canbe approximated by linear extrapolation of the straight line after aligning the plots. This gives the first time step value as 2.09×10^{-23}. Hence, it sould require having a γ of similar size in order to influence the output at the first time step. Vector V_2 is a unit vector, and the last time step component is -0.94 at $k = 50$. This implies that $V_2\underline{\gamma}$ can have a small influence on the control action of FI compared to NI toward the end of the trajectory.

Of course this influence is undesirable because it increases the control action without any benefit. The conclusion is that one should use NI. The FI development can be though of as step in the proof of the useful stable incerse NI.

Consider the Skip Step method, and again use the 3^{rd} order system for the purpose of illustration. Figure 4.6 gives the ouput produced by FS introducing skip steps between the original steps. Clearly, there is undesirable behavior at the newly introduced time steps. Figure 4.7 shows the corresponding control input history. Not only are the big changes from time step to time step objectionable, but the magnitude of the control action is particularly large reaching a maximum value of 446. Figure 4.8 presents the control action when using NS, which appears well behaved and reaches a maximum of 1.185. Figure 4.9 shows show the corresponding output error history using NS, with the error at the addressed time steps being at the level of numerical zeros 10^{-15} or 10^{-16} while the error at the unaddressed time steps being around 10^{-6} or 10^{-7}. One is tempted to try to compare this with the error one would have if one used only one zero order hold input from addressed step to addressed step as control people normally do, instead of two such holds. Of course this comparison is not possible, because the solution is unstable, and between the time steps the maximum error is growing exponentially.

It is possible to create a formula explaining the behavior in Figure 4.6, telling how the output at the inctroduced intermediate time steps is related to the two neighboring addressed time steps. Consider

$$\underline{y}_d^* = P_{Od} P_I \underline{u}, \quad \underline{u} = P_I^{-1} P_{Od}^{\dagger} \underline{y}_d^* \tag{4.56}$$

deleting all odd numbered time steps. Let $p = 6$ and use this control to find what happens

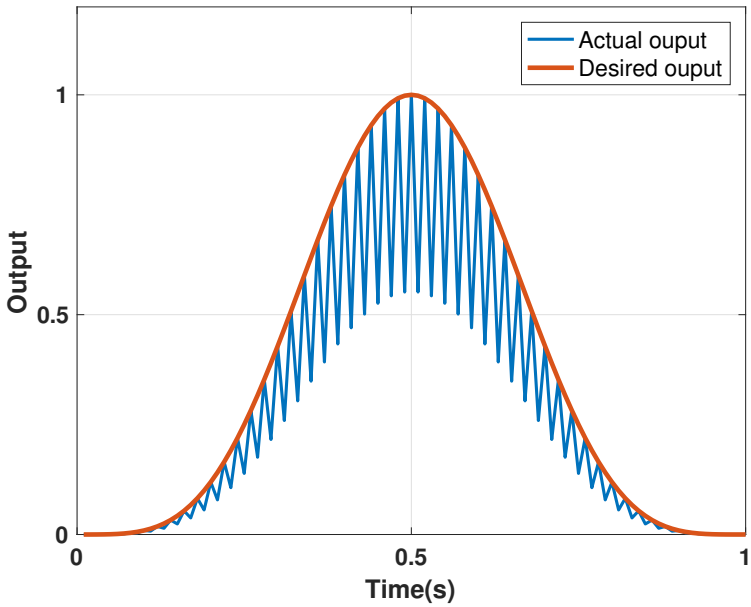

Figure 4.6: The actual output using Longman-JiLLL FS on $y^*(t) = 0.25 * [1 - \cos(2\pi t)]^2$

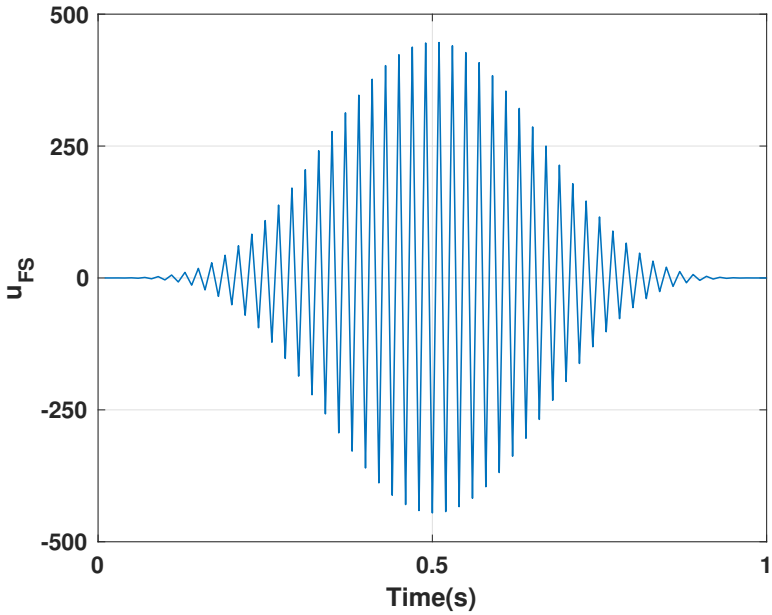

Figure 4.7: The control input producing output in Figure 4.6

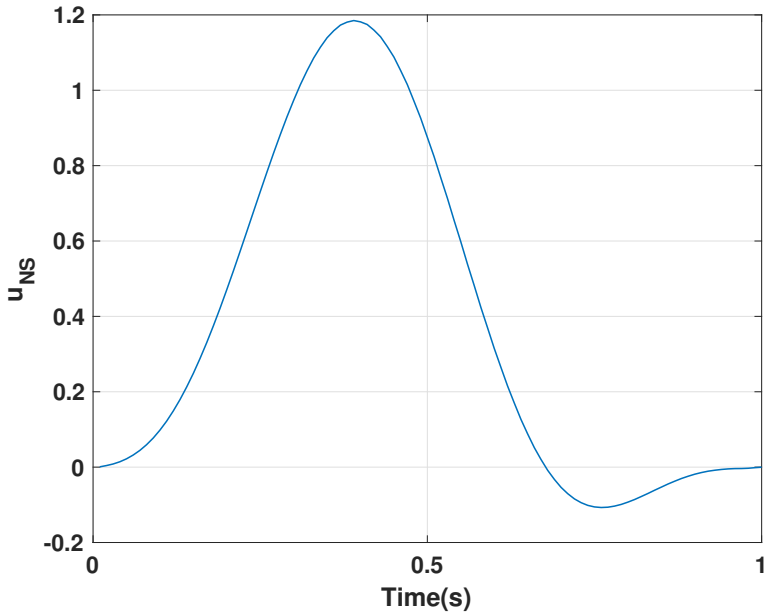

Figure 4.8: The control input using Longman-JiLLL NS on $y^*(t) = 0.25 * [1 - \cos(2\pi t)]^2$

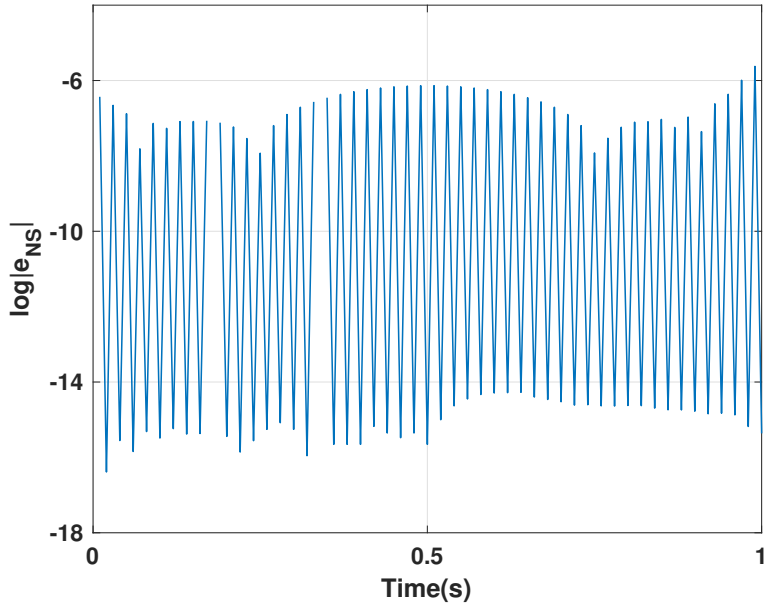

Figure 4.9: Logrithm of error magnitude at all time steps using Longman-JiLLL NS

at the between time steps, by replacing P_{Od} by the odd numbered rows instead of the even numbered rows P_{OE} to find the ouput at the even numbered rows $\underline{y}_E^*$. Then

$$\begin{aligned} \underline{y}_E^* &= P_{OE} P_{Od}^{\dagger} \underline{u}_d^* \\ \alpha &= \tfrac{-z_O}{\left(z_O^2+1\right)}, \quad \beta = \tfrac{1}{\left(z_O^2+1\right)} \end{aligned} \tag{4.57}$$

where

$$\begin{aligned} P_{OE} &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -z_O & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -z_O & 1 & 0 \end{bmatrix} \\ P_{Od}^{\dagger} &= \begin{bmatrix} \alpha & \beta & & & & \\ & & \alpha & \beta & & \\ & & & & \alpha & \beta \end{bmatrix}^T, \quad \underline{y}_E^* = \begin{bmatrix} \alpha & & \\ \alpha & \alpha & \\ & \alpha & \alpha \end{bmatrix} \underline{y}_d^* \end{aligned} \tag{4.58}$$

The conclusion is that the output at the even time steps is the following function of the even time steps just before and just after

$$y\left(2k+1\right) = \alpha\left[y\left(2k\right) + y\left(2k+2\right)\right] \tag{4.59}$$

For $z_O = -2.90$, $\alpha = 0.307$, at the peak in Figure 4.9 both addressed sample times have values near unity, at the between sample times the output is predicted to be 0.6 instead of unity. Note that if $\alpha = 1/2$, then this would correspond to linear interpolation. However, for a zero outside the unit circle $\alpha > 1$, and the further outside the unit circle, the larger the error. This produces the sawtooth behavior of the output tracking using the FS stable inverse.

The ”Clean” Stable Inverse Solutions

This section presents the ”clean” version of stable inverse solutions, and the reason being referred as ”clean” will be explained later. Consider the same 3^{rd} order system governed by Equation 4.16 with two zeros,z_1 and z_2. Given the case of two sampling zeros, one of them is outside the unit circle, and the other one is inside. However, the approach presented in this chapter could handle more general cases. This time we start from an autoregressive exogenoues **ARX** model representing the same system. To be more precise, we study the model as arecursive equation, and the first output observed is $y(1)$ controlled by $u(0)$, we need to know the 5 initial conditions $y(-2)$, $y(-1)$, $y(0)$, and $u(-2)$, $u(-1)$ to start the

recursion.

$$\left[\begin{array}{ccc|ccccc} a_3 & a_2 & a_1 & 1 & & & & \\ & \ddots & \ddots & a_1 & \ddots & & & \\ & & \ddots & a_2 & a_1 & 1 & & \\ & & & a_3 & \ddots & \ddots & \ddots & \\ & & & & \ddots & \ddots & \ddots & \ddots \end{array}\right] \left[\begin{array}{c} \\ \\ \\ y(-2) \\ y(-1) \\ y(0) \\ \hline y(1) \\ y(2) \\ y(3) \\ \vdots \\ y(p) \end{array}\right]$$

$$= \left[\begin{array}{ccc|ccccc} b_3 & b_2 & & b_1 & & & & \\ & \ddots & & b_2 & \ddots & & & \\ & & & b_3 & b_2 & b_1 & & \\ & & & & \ddots & \ddots & \ddots & \\ & & & & & \ddots & \ddots & \ddots \end{array}\right] \left[\begin{array}{c} \\ \\ \\ u(-2) \\ u(-1) \\ \hline u(0) \\ u(1) \\ u(2) \\ \vdots \\ u(p-1) \end{array}\right] \tag{4.60}$$

Denote locally as

$$\begin{bmatrix} A_{IC} & A \end{bmatrix} \begin{bmatrix} \underline{y}_{IC} \\ \underline{y} \end{bmatrix} = \begin{bmatrix} B_{IC} & B \end{bmatrix} \begin{bmatrix} \underline{u}_{IC} \\ \underline{u} \end{bmatrix} \tag{4.61}$$

As a result, we model the same system equivalent to P matrix model and Equation 4.16 as

$$\begin{aligned} A_{IC}\underline{y}_{IC} + A\underline{y} &= B_{IC}\underline{u}_{IC} + B\underline{u} \\ A\underline{y} &= B\underline{u} + \left(A_{IC}\underline{y}_{IC} - B_{IC}\underline{u}_{IC}\right) \\ \underline{y} &= (A^{-1}B)\,\underline{u} + \underline{y}_{\underline{u}=0} \end{aligned}$$

with the connection to P matrix model as $P = A^{-1}B$, and $\bar{A}x(0) = \left(A^{-1}A_{IC}\underline{y}_{IC} - A^{-1}B_{IC}\underline{u}_{IC}\right)$. Note that we decompose the response of the system into two separate parts, where $P\underline{u} = A^{-1}B\underline{u}$ models the zero initial condition response, and the rest part $\underline{y}_{\underline{u}=0}$ models the intial input response, i.e. excited only by the nonzero initial conditions. Given the community property of lower triangular Toeplitz matrices, reorder as

$$\underline{y} = BA^{-1}\underline{u} + \underline{y}_{\underline{u}=0} \tag{4.63}$$

Before proceeding, let's review the instability hidden in the inverse solution. Reference [41] initiated it in the form of

$$u(k) = u_p(k) + c_1(z_1)^k + c_2(z_2)^k \tag{4.64}$$

where $u_p(k)$ is a particular solution, c_1 and c_2 are arbitrary constants determined by initial conditions. References [1], [2] point that if these are two sampling zeros, z_1 is aymptotially approaching -3.7321 as sample time T approaching zero, where z_2 is approaching the reciprocal of z_1. For the issue of inverse stability, what one really wants it $c_1 = 0$ such that the exponentially growing term is gone. Generally speaking, one is intereted in finding a systematic way to find a solution with $c_1 = 0$, or $c_2 = 0$, or both to be zero.

Driven by the idea above, we delete the first 2 rows (elements) aiming to free 2 degrees of freedom of initial time steps to pick those 2 coefficients c_1 and c_2

$$\underline{y}_{dd} = B_{dd}A^{-1}\underline{u} + \underline{y}_{\underline{u}=0,dd} \tag{4.65}$$

For the output tracking problem, substitute the prespecified desired output trajectory $\underline{y}^*$, the stable inverse solution we claim is

$$\underline{u}^* = AB_{dd}^{\dagger}\left(\underline{y}_{dd}^* - \underline{y}_{\underline{u}=0,dd}\right) \tag{4.66}$$

Statement This "clean" stable inverse has no components on the solutions of the corresponding homogeneous equation, i.e. $c_1 = c_2 = 0$.

Justification

- **Case 1:** $z_1 \neq z_2$ In this case, B_{dd} as a $p-2$ by p matrix modeling an underdetermined system of linear equations, having z_1 and z_2 as the same roots of the characteristic polynomial from the starting time step to the end, i.e. with no ending effects in the

finite time problem. And the two linearly independet solutions of the homogeneous equation, $c_1(z_1)^k$ and $c_2(z_2)^k$, $k = 1, 2, \ldots, p$ span the 2-dimensional null space of matrix B_{dd}. Hence, any items involving these 2 transient terms in the input space map zero components in the ouput space. Simple checks could be done by (1) pre-multiplying B_{dd} on the solution gives the component-wise zero vector; (2) project $(z_1)^k$ or $(z_2)^k$ on the 2 basis singular vectors in the null space resulting the orthogonal projection.

- **Case 2:** $z_1 = z_2$ In this case, the solution to the homegeneous equation is in the form of $c_1(z_1)^k + c_2 k(z_1)^k$. Similarly, these 2 linearly independent solutions are guaranteed to span the 2-dimensional null space of B_{dd}.

Comments

- There is an infinite number of solutions satisfying this undertermined **ARX** system of linear equations, and the Penrose-Moore pseudo-inverse solution picks the minimum Euclidean norm of $A\underline{u}$, and gives zero tracking error for the rest $(p - 2)$ time steps of the desired trajectory $y^*(k)$, $k = 1, 2, \ldots, p$. Figure 4.10 and Figure 4.11 illustrates the required control solution trying to follow the desired output $y^*(t) = 0.25 * [1 - \cos(2\pi t)]^2$ and the resulting actual output error.
- This stable inverse solution contains only the particular solution part with no transient terms satisfying the homogeneous equation, which is illustrated in 4.12.
- The algorithm is independent of the source of NMP zeros, could handle both sampling or intrinsic zeros. Since we start from an **ARX** model with no prior knowledge where NMP zeros come from. And this makes the algorithm easy to calculate without the

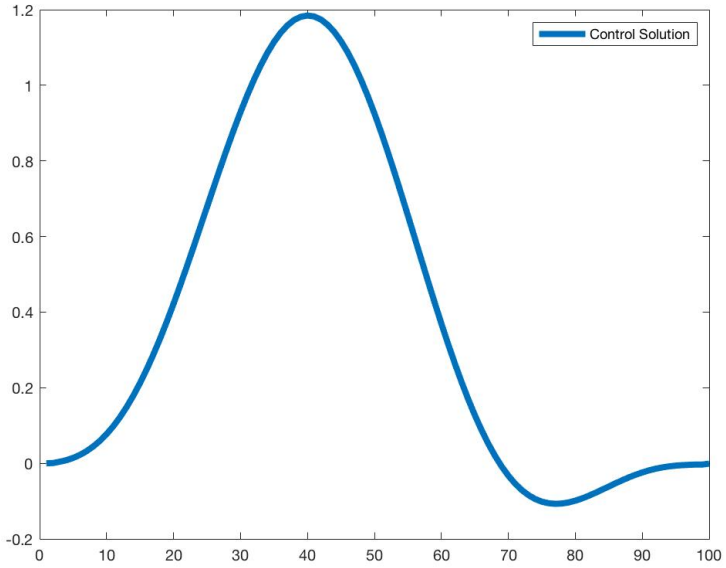

Figure 4.10: The "clean" control inverse solution on $y^*(t) = 0.25 * [1 - \cos(2\pi t)]^2$

necessities developing a state space model.

- Initial deleing the number of initial steps to be deleted is equal to the number of zeros is enough to get the "clean" solution, however, over-deletion might be due to the over-estimate of the order of the system or over-deletion on the upper bound of th zeros is also permitted.

Generalization The algorithm presented in Equation 4.66 could be easily generated to achieve different objectives. Note that we remove all of the transient terms satisfying the homogeneous equation by initial deleting the first time steps euql to the number of zeros freeing the corresponding number of null space. This might over beat the issue of inverse stability, since consider Equation 4.16 is the discretized model containing only sampling zeros, i.e. one NMP zero $z_1 = z_O$ and one MP zero $z_2 = z_I$. We could achieve the goal of removing only the exponentially growing term $c_1(z_O)^k$ without asking the removal of

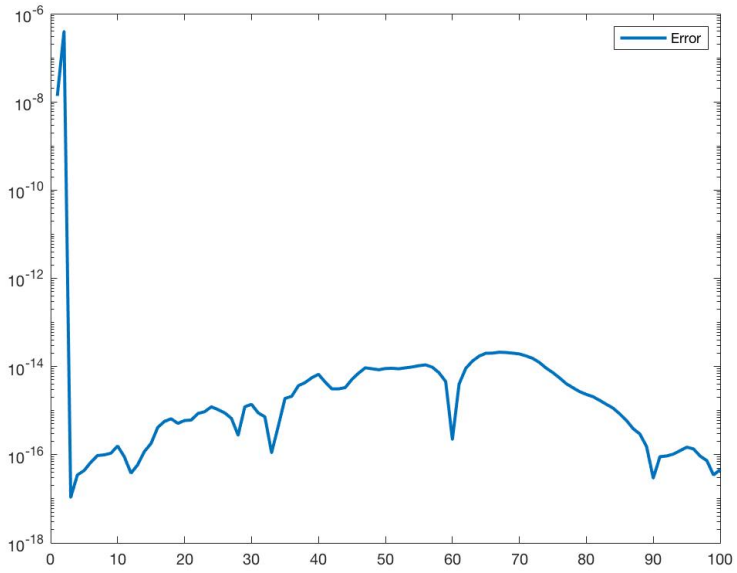

Figure 4.11: The actual output error using Figure 4.10

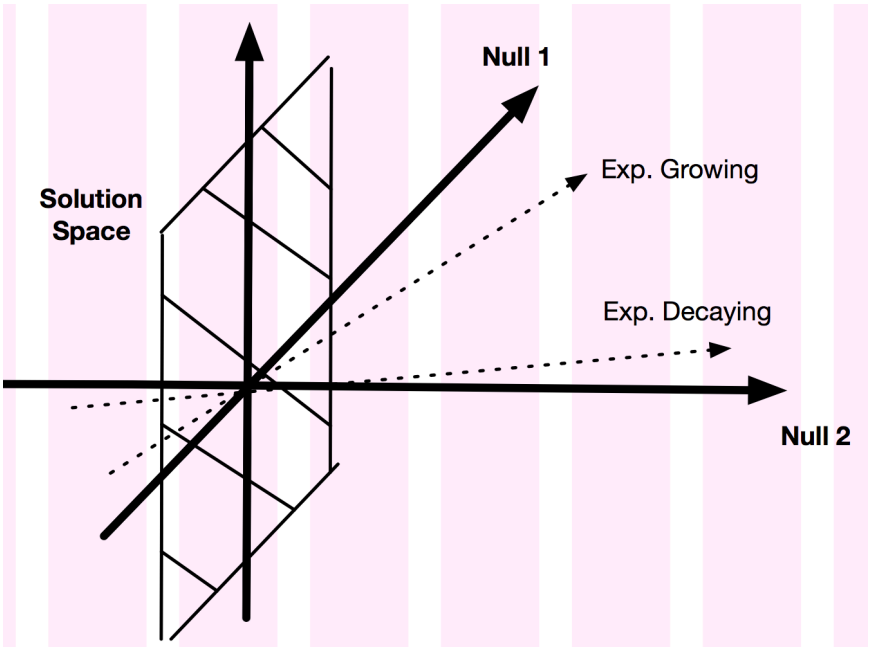

Figure 4.12: The solution space of the ”clean” stable inverse

$c_2(z_I)^k$, by initial deleting the 1^{st} row (element) in the model. We factorize $B = B_O B_I$ where the bi-diagonal matrix B_O models the NMP the finite-time dynamics of z_O, and B_I models z_I. Then the solution is

$$\underline{u}^* = AB_I^{-1}B_{Od}^{\dagger}\left(\underline{y}_d^* - \underline{y}_{\underline{u}=0,d}\right) \tag{4.67}$$

Note that this solution is same as the one by FI, and we have another interpretaiton of the solution.

4.4 Apply Stable Inverse Theorem in a Linear Discrete Time System

It is of the author's interest and for the purpose of comparison, to appy the existing stable inverse theorem on the same 3^{rd} order discrete time system as Equation 4.16. Given the problem setup, we are solving, if possible, a state space model with the reference triplet input-state-output trajectory $\begin{pmatrix} x^*(k) & u^*(k) & y^*(k) \end{pmatrix}$ satisfying

$$\begin{cases} x^*(k+1) = Ax^*(k) + Bu^*(k) \\ y^*(k) = Cx^*(k) \end{cases} \quad k \in Z \tag{4.68}$$

Note that we assume a onte time-step delay from input to output, however, more general cases could be easily inferred, the corresponding inverse system is

$$\begin{cases} u^*(k) = (CB)^{-1}\left[y^*(k+1) - CAx^*(k)\right] \\ y^*(k+1) = y^*(k+1) \end{cases} \qquad k \in Z \tag{4.69}$$

and exact tracking is maintained. References [48] and [49] introduced a transformation T_1 such that

$$\begin{bmatrix} \xi(k) \\ \eta(k) \end{bmatrix} = T_1 x(k) \tag{4.70}$$

where

$$T_1 = \begin{bmatrix} b_3 & b_2 & b_1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{4.71}$$

by appropriately defining

$$\xi(k) = y(k), \quad \eta(k) = \begin{bmatrix} \bar{y}(k) \\ \bar{y}(k+1) \end{bmatrix} \tag{4.72}$$

and $\bar{y}(k)$ and $\bar{y}(k+1)$ defined in Equation 4.21. Then follow the procedure as described in the Chapter 3, state dynamics could be rewritten in these new coordinates as

$$\begin{bmatrix} \xi(k+1) \\ \eta_{2\times1}(k+1) \end{bmatrix} = \begin{bmatrix} \hat{A}_1 & \hat{A}_2 \\ \hat{A}_3 & \hat{A}_4 \end{bmatrix} \begin{bmatrix} \xi(k) \\ \eta_{2\times1}(k) \end{bmatrix} + \begin{bmatrix} \hat{B}_1 \\ \hat{B}_2 \end{bmatrix} u(k) \tag{4.73}$$

where

$$\hat{A}_1 = \frac{b_2 - a_1 b_1}{b_1}, \quad \hat{A}_2 = \begin{bmatrix} -a_3 b_1 - \frac{b_3(b_2 - a_1 b_1)}{b_1} & b_3 - a_2 b_1 - \frac{b_2(b_2 - a_1 b_1)}{b_1} \end{bmatrix},$$
$$\hat{A}_3 = \begin{bmatrix} 0 \\ \\ \frac{1}{b_1} \end{bmatrix}, \quad \hat{A}_4 = \begin{bmatrix} 0 & 1 \\ \\ -\frac{b_3}{b_1} & -\frac{b_2}{b_1} \end{bmatrix}, \quad \hat{B}_1 = b_1, \quad \hat{B}_2 = \begin{bmatrix} 0 \\ \\ 0 \end{bmatrix} \tag{4.74}$$

For the inverse problem, to find the reference control input written in these new states, if possibly bounded

$$u(k) = (CB)^{-1} \left[y(k+1) - CAT_1^{-1} \begin{bmatrix} \xi(k+1) \\ \\ \eta_{2\times 1}(k+1) \end{bmatrix} \right] \tag{4.75}$$

The statement now is that, given the desired p time-step trajectory, finding the required input-state trajectory is equivalent to finding bounded solutions to the system's internal dynacmis which is

$$\eta_{2\times 1}(k+1) = A_\eta \eta_{2\times 1}(k) + B_\eta Y^*_{2\times 1}(k) \tag{4.76}$$

where

$$A_\eta = \hat{A}_4, \quad B_\eta = \begin{bmatrix} 0 & 0 \\ \\ 0 & \frac{1}{b_1} \end{bmatrix}, \quad Y^*_{2\times 1}(k) = \begin{bmatrix} y^*(k+1) \\ \\ y^*(k) \end{bmatrix} \tag{4.77}$$

Apply the eigen-decomposition $A_\eta = U_\eta \Lambda_\eta U_\eta^{-1}$ on the internal dynacmis system whose poles codincide with the zeros of the original 3^{th} order system to decouple the minimum

and non-minimum phase zeros, i.e. the stable and unstable internal dynamics as

$$\begin{bmatrix} \hat{\eta}_u(k+1) \\ \hat{\eta}_s(k+1) \end{bmatrix} = \begin{bmatrix} \lambda_O & 0 \\ 0 & \lambda_I \end{bmatrix} \begin{bmatrix} \hat{\eta}_u(k) \\ \hat{\eta}_s(k) \end{bmatrix} + \hat{B}_\eta \begin{bmatrix} y^*(k+1) \\ y^*(k) \end{bmatrix} \tag{4.78}$$

where

$$\begin{aligned} &\hat{\eta}_{2\times 1}(k) = U_\eta^{-1}\eta_{2\times 1}(k), \quad \hat{B}_\eta = U_\eta^{-1}B_\eta = \begin{bmatrix} \hat{B}_{\eta,u} \\ \hat{B}_{\eta,s} \end{bmatrix}, \quad \lambda_O = z_O, \quad \lambda_I = z_I \\ &U_\eta = \begin{bmatrix} \frac{b_2+\sqrt{b_2^2-4b_1b_3}}{2b_3} - \frac{b_2}{b_3} & \frac{b_2-\sqrt{b_2^2-4b_1b_3}}{2b_3} - \frac{b_2}{b_3} \\ 1 & 1 \end{bmatrix} \end{aligned} \tag{4.79}$$

It is obvious that the stable part of internal dynamics is evolved forward in time, while its unstable part backward in time, and could be calculated as

$$\begin{cases} \hat{\eta}_s^*(k) = \sum_{j=0}^{\infty} z_I^j \hat{B}_{\eta,s} \begin{bmatrix} y^*(k-j) \\ y^*(k-(j+1)) \end{bmatrix} \\ \hat{\eta}_u^*(k) = -\sum_{j=1}^{\infty} \frac{1}{z_O^j} \hat{B}_{\eta,u} \begin{bmatrix} y^*(k+j) \\ y^*(k+(j-1)) \end{bmatrix} \end{cases} \tag{4.80}$$

Note that as shown above, the desired internal state can be computed by stable inverse of the system by substituting the desired trajectory to maintain exact tracking. The input to the system required is then computed using Equation 4.75.

4.5 Conclusions

The skip step stable inverse is important for many potential extensions of the inverse idea to various control methods. The stable inverse results allow one to preplan a desired trajectory and then perform it. The majority of control theory aims for some form of a real time control law. Skip step forms a basic ingredient toward bridging from batch to real time. If one can incorporate batch stable inverse ideas, then the first approach is to ask if it is possible to do short batch updates sequentially, in which case the initial delete approach is not natural. Some investigations are in progress aiming to extend the use of the stable inverse concepts in References [18], [50], [51], [52], and [53].

- **Iterative Learning Control.** ILC was the original motivation for the development of the stable inverse concepts presented here. ILC attempts to converge to the full inverse control action. Hence, it is very often trying to converge to an unstable control action. Often in applications people do not realize. The instability may not be evident for many iterations, and could also be stabilized by the analog to digital converters. For a slow sample rate it can become evident earlier, and then people wonder what is wrong. We created NI to address this problem. Reference [18] explains why ILC, without actually applying the NI inverse, without making any use of the prescribed pseudoinverse, nevertheless converges to the right answer when using initial deletion. Reference [51] considers ILC to perform local learning in a trajectory, by using a quadratic cost control in general, but phasing into the skip step stable inverse for some portion of the trajectory that needs high precision tracking.
- **p-Step Ahead Control, Linear Model Predictive Control, and Indirect Adaptive**

Control. One step ahead control uses a model to compute the control action at the current time step to produce the output desired at the next time step. Before it can be useful, it must be phased in to honor actuator saturation limits, and being a true inverse it requires that the system have a stable inverse. Referene [52] generalizes this to p-step ahead control, updating the control action every p steps instead of every one step. It determines how small p can be to give a stable implementation using skip step, and it can be quite small. So it only requires knowledge of future desired control for a few steps. Note that his can be reformulated as Linear Model Predictive Control as Reference [54] that updates ever p steps instead of every step. This offers the ability to converge to zero tracking error at every time step of the skip step inverse, instead of the usual aim to converge to a quadratic cost solution. Indirect discrete time adaptive control [55] combines one step ahead control with the projection algorithm to perform real time identification updates. It has limited applications, because it requires a stable inverse. Reference [53] presents a first pass at developing indirect discrete time adaptive control that does have this requirement.

Chapter 5

Iterative Learning Control for Linear Discrete Time Non-Minimum Phase Systems

5.1 Introduction

In routine feedback control the input command is the desired output. The actual output is a convolution integral of the forcing function – essentially never equal to the command. If it were equal, the control system designers would be solving an inverse problem. Iterative Learning Control (ILC) addresses this in discrete time by iterating in the real world to converge to that command producing the desired output, adjusting it each run based on previous run error. For a majority or real world systems, this asks to solve an ill-conditioned inverse problem, one whose exact solution is unstable and completely undesirable. For a

simple robot example performing a one second maneuver, the condition number of the matrix to be inverted is 10^{52}, although the matrix is guaranteed full rank. This is typical of discrete time inverse problems in digital control. Numerical methods such as Tikhonov regularization, fail to get zero tracking error at any time step. The author and co-workers have developed a stable inverse discussed in the previous chapter, i.e. the Initial Delete, that produces zero tracking error at all steps except the first step for the robot problem, or more generally, the first time steps. This can be thought of a new kind of regularization for lower triangular Toeplitz matrices of Markov parameters. This chapter examines how to apply the new stable inverse las to ILC. Three main ILC laws are shown to converge to a solution completely determined by the control applied in the initial run that starts the iteration. This dependence is very small, so one can reasonably use any initial run. But by picking an initial input that goes to zero approaching the final time step, this influence becomes particularly small. And simply commanding zero in the first run, gives an optimal inverse minimizing the Euclidean norm control action associated with zero tracking error at all time steps but the first. The error in the initial time step is studied and shown to be well behaved.

Feedback control systems aim to execute whatever command is given to them. Thus, given the differential equation for the output as a function of the input command, the control law is aiming to solve an inverse problem. There is always considerable error which can be characterized by the control system bandwidth. Plotting the response to sinusoidal inputs, when the amplitude of the output sinusoid has decayed to 70

ILC stores data from the previous run, so that it is a digital control method solving a discrete-time inverse problem. The real world for digital control systems is governed by

ordinary differential equations, but the digital controller creates the forcing function applied to this equation, updating it each sample time. Each update is continuously applied to the differential equation until a new update arrives from the controller - called a zero-order hold. If one looks at the solution to the differential equation at the sample times, one can make a linear difference equation that has identical solution to the differential equation. Reference [1], [2] prove that the process of converting to a difference equation model introduces the forcing function at additional sample times, enough to make the most recent output time step in the equation be one step ahead of the most recent forcing function input time step. When the discretization introduces three or more additional terms, and the sample rate is reasonable, the characteristic polynomial of the forcing function side of the equation will contain a root or roots that are larger than one in magnitude. This makes the discrete-time inverse problem unstable for a majority of digital control systems in the world.

The implication is, if one wants to have perfect tracking of a desired discrete-time trajectory at all time steps, the control action needed is unstable, and grows exponentially with time steps. The inverse problem error must be zero at the sample times, but between sample times the solution of the differential equation (after some initial time steps) is growing in magnitude exponentially, and alternating in sign each time step. Of course, this exponential error growth when perfectly following the discrete-time desired trajectory does not address the initial intended problem of finding the input to accurately follow the desired continuous time output.

Mathematically, the discrete-time inverse problem is asking to invert an ill-conditioned matrix. We note that the lower triangular Toeplitz matrix of Markov parameters is guaranteed full rank analytically, indicating that the inverse exists. Numerical methods that

aim to address this kind of problem include Tikhonov regularization. This is unapplealing, since it will not produce zero error at any of the sample times. We comment that the stable inverse solution presented here could be of use in numerical methods as an alternative to Tikhonov regularization. Another numerical approach is to set the small singular value(s) producing the ill-conditioning to zero, then use a pseudo-inverse. This is unapplealing for the same reason. In this paper we wish to do better.

There is a theory established for stable inverses in References [3], [4], [5], [6], [8], [9], [10], [11] and [13]. Reference [13] outlines the scheme of stable inverse theorem based ILC, and relates the gradient based algorithm searching the optimal solution to ILC law in iteration domain, but it makes use of pre- and post- extension of the original finite time desired output, as simlarly done in Reference [56]. Reference [9] introduces a band filter sovling for NMP systems, and justifies its better performance than a low pass filter, by constructing the optimization problem in frequency domain. Reference [57] studies the unsatisfyng results done by gradient-based optimization algorithm caused by zero dynamics of system. Reference [8] shows that there is no direct relation between ILC law using adjoint system and stable inverse. Reference [11] proposed a non-causal ILC law based on the structure of error and control action observed in previous iteration.

We wish to take advantage of the JiLLL NI result when using ILC, but it is not reasonable to perform the pseudoinverse step with ILC. Making use of this step would simply find an inverse of the model, and ILC seeks to iterate with the world and solve the inverse problem iteratively in the real world, not a model of the world. ILC can be very effective. In experiments performed on the robot, the tracking error was decreased by a factor of 1000 in 12 iterations with the world, and this factor if far below the error level

of our model of the world and it approaches the reproducibility level of the hardware. The ILC laws make use of a model, but must be sufficiently robust to model error that they still produce convergence.

To start an ILC iteration, one first applies some input to the system and observes the output. Since we are concerned with control systems the most logical input is the desired output. To match ILC to the JiLLL NI inverse, the ILC updates will not consider the output error at the first time step or first few time steps, the number being the number of non-minimum phase zeros in the transfer function. Then the output at these time steps becomes a function of the command given during this first run. This chapter studies this in detail. It is concluded that three main classes of ILC laws converge to the same control action, given the same initial run. The influence of the initial run, introduces a small component on the unstable solution, but it is so small that it does not exhibit any instability in the p time steps of the given trajectory. It is also shown that if one decides to command zero during the first run, then the converged solution is the same as the JiLLL NI solution, giving the minimum norm control action for the first time step.

The authors and co-workers developed a set of stable inverses as described in the previous chapter. We consider JiLLL NI here. In this paper we illustrate with a simple 3^{rd} order system that models the input-ouput relationship of each axis of a Robotics Research Corporation robot. Then, given a p-time step desired output, JiLLL NI gives a p-time step input that produces zero error at all time steps except the first step. The first step is determined by a minimum Euclidean norm input action. The unstable behavior produced by the ill-conditioning is eliminated. When we apply the approach to the ILC problem, the first step instead is the result of the command given in the initial run. We comment that

for a one-second desired trajectory for the robot link, sampled at 100Hz sample rate, the condition number of the matrix to be inverted is of the order of magnitude of 10^{52}. We also comment that Matlab is not able to compute this condition number. We need other techniques to estimate this number. We reiterate, the matrix is analytically guaranteed to be full rank, the inverse is guaranteed to exist.

We wish to take advantage of the JiLLL NI result when using ILC, but it is not reasonable to perform the pseudoinverse step with ILC. Making use of this step would simply find an inverse of the model, and ILC seeks to iterate with the world and solve the inverse problem iteratively in the real world, not a model of the world. ILC can be very effective. In experiments performed on the robot, the tracking error was decreased by a factor of 1000 in 12 iterations with the world, and this factor if far below the error level of our model of the world and it approaches the reproducibility level of the hardware. The ILC laws make use of a model, but must be sufficiently robust to model error that they still produce convergence.

To start an ILC iteration, one first applies some input to the system and observes the output. Since we are concerned with control systems the most logical input is the desired output. To match ILC to the JiLLL NI inverse, the ILC updates will not consider the output error at the first time step or first few time steps, the number being the number of non-minimum phase zeros in the transfer function. Then the ouput at these time steps becomes a function of the command given during this first run. This paper studies this in detal. It is concluded that three main classes of ILC laws converge to the same control action, given the same initial run. The influence of the initial run, introduces a small component on the unstable solution, but it is so small that it does not exhibit any instability in the p time steps

of the given trajectory. It is also shown that if one decides to command zero during the first run, then the converged solution is the same as the JiLLL NI solution, giving the minimum norm control action for the first time step.

5.2 Iterative Learning Control Laws

ILC is a rather new type of control that adjusts the command to a feedback control system repeatedly performing a desired task under a repeating disturbance. The command is adjusted after each run, based on the error observed in the previous run, and the aim is to achieve zero error $\underline{e}_j(k) = \underline{y}^*(k) - \underline{y}_j(k)$ tracking the repeated desired trajectory as the repetition number j tends to infinity. There have been many ILC approaches developed, and References [33], [28], [29] and [58] give good perspective on the ILC field that developed. Each repetition starts from the same initial condition. A general linear learning control law is given by

$$\underline{u}_{j+1} = \underline{u}_j + L\underline{e}_j \tag{5.1}$$

where L is a matrix of learning gains. By the use of a difference operator, one can write the error propagation equation

$$\underline{e}_{j+1} = (I - PL)\underline{e}_j \tag{5.2}$$

where I is the identity matrix.

The simplest control law implements the following concept, if the output as a given time step in the previous run was two units too small, add two units to the command in this run, at the appropriate time step. Note that we assume a one time-step delay from input to

output, we add two units to the command one step before the error considered. Also, in the spirit of classical control, we can insert a scalar learning gain ϕ so that instead of asking for 2 units more, we ask for 2ϕ. This is sometimes referred to as P type ILC as defined by Reference [20]. Perhaps being a bit less aggressive and asking for less change in the next iteration can have some benefit. As a result, the learning gain matrix is a p by p identity matrix multiplied by this scalar learning gain. Many are not aware of the extreme behavior that this ILC can exhibit, and this phenomenon is studies by References [59], [60]. Here, we consider three other main laws. And refer to the first law investigated in Reference [61] as the P Transpose Law (or the Contraction Mappling Law),

$$L = \phi P^T \tag{5.3}$$

and it is a contraction mapping in the sense of the Euclidean norm of the tracking error from iteration to iteration. The second law investigated in Reference [62] is the partial isometry law formed from the singular value decomposition of the $P = USV^T$ matrix according to

$$L = \phi V U^T \tag{5.4}$$

Here U and V are unitary matrices whose columns (and rows) represent unit vectors in p dimensional space and these vectors are orthogonal. And people also choose to pick the learning gain matrix in such a way as to minimize a quadratic cost each iteration that controls the learning that controls the learning transients in References [63] and [57]. We comment that by picking the quadratic cost weights appropriately, all three laws can be

presented by the quadratic cost law as in Reference [64]. The quadratic cost function that uses a single scalar weight r, results in the following learning law L

$$J_{j+1} = \underline{e}_{j+1}^T \underline{e}_{j+1} + r\delta_{j+1}\underline{u}^T \delta_{j+1}\underline{u} \quad ; \quad L = \left(P^T P + rI\right)^{-1} P^T \tag{5.5}$$

where the difference operator $\delta_j \xi = \xi_j - \xi_{j-1}$ holds for any quantity ξ_j. Using any of these laws in Equation 5.2, one concludes that the error history converges to zero tracking error as j tends to infinity, for all possible intial runs, if and only if all eigenvalues of $(I - PL)$ are less than one in magnitude. If all singular values of $(I - PL)$ are less than one, the error converges monotonically with iterations in the sense of the Euclidean norm. For sufficiently small gain ϕ or large r, each of these laws is guaranteed to converge monotonically to zero tracking error at all time steps, which means it converges to the unstable solution in Equation 4.7.

5.3 A New Stable Inverse Based Iterative Learning Control

Behavior of ILC When Solving the Ill-Conditioned Problem

The mathematics of ILC says it converges to the P inverse solution. Properties of this solution are that it has zero error at every time step, control action alternates in sign every time step and grows in matnitude exponentially, and produces errors in the solution of the differential equation that grow exponentially in magnitude between successive time steps.

Such a solution defeats the purpose of asking for zero error. References [65], [66], [67], [68], [69], [70], [71] and [72] study the conversions of adaptive control to the learning control problems. It is expected that if two major goals are achived, i.e. the system model converges to the true model and meantime zero tracking error is achieved at all time steps, the control converges to the inverse model solution. Reference [73] gives a unifying understanding of the stability boundary for convergence to zero tracking error, and of a stability condition by using frequency response allowing dynamic and inverse dynamic control laws.

Sometimes these bad properties are not observed when ILC is applied to the real world. Often, the error decreases reasonbly fast, but it seems to have finished converging at a disappointing error level far from the zero error promised by the mathematical analysis. References [40], [39], [74] explain this phenomenon, saying that the iterations have not yet converged, and that with enough iterations one will see the instability appearing, and the error at sample times decaying further. This means that one may have improved the error level, but the iteration process is poised to become unstable. In practice, this phase of the convergence process may not be observed because the iterations are terminated before actually reaching convergence, or because the finite word length in the analog to digital and digital to analog converters does not allow accumulation of the learning signal.

Modifying ILC Laws to Aim for A Stable Inverse

Reference [39] applied the idea presented in References [36] and [41] to ILC. Given a third order disrete time system, one deletes the first initial row to form P_d, picking $L = \phi P_d^T$

for the modified P Transpose Law, $L = \phi V_d {U_d}^T$ for the partial isometry law where V_d and U_d are the singular vector matrices of P_d, and $L = \left({P_d}^T P_d + rI\right)^{-1} {P_d}^T$ for the quadratic cost law. These ILC laws are updating the control action for all p time steps, but ask for zero error at only the addressed time steps remaining after deleting the initial row (or rows whose number is equal to the number of non-minimum phase zeros). Reference [41] asks to pick the extra freedom by picking a minimum norm solution, but the ILC approach simply applies whatever command one wishes for the first iteration, and then starts using any of the above control laws. The questions addressed here are: How well will this work? What final error level is produced? Does it make a difference which law we use? How significantly is the final error level affected by our choice of the initial run, etc?

Extensive numerical experience shows that no matter how one chooses the control action in the initial iteration, one could always achieve zero tracking error on the time steps remaining after deleting the chosen initial steps. After using the modified ILC laws, the final level of the control action and also the unaddressed error at the first time step, are insensitive to the choice of the control action in the initial run. This may appear counter-intuitive, and we seek to explain this phenomenon.

5.4 Analytical and Numerical Results

We established without asking for zero error for the first (or first few) time steps allows one to create solutions to the inverse problem that are stable. There are of course an infinite number of solutions to the underspecified set of equations, one of which is the original inverse solution which we do not want. The Moore-Penrose pseudoinverse was used in the

stable inverse chosen above, and seen to eliminate the unstable inverse issue. In section 5.4.1 we obtain an expression for the set of all possible inverse solutions, all expressed in terms of a parameter γ that indicates the difference between any given solution and the Moore-Penrose solution.

ILC cannot make use of the Moore-Penrose pseudoinverse as a learning law during its iterations. ILC aims to get to zero tracking error in the real world, not in our imperfect model of the world, by iterative adjustments of the system input using world response data. ILC starts with an initial run, applying whatever one chooses, for example, as the input to the control system, one logically would ask for the desired output. The authours' group has investigated three main classes of ILC laws in References [61], [62] and [75] and term these as P transpose law, partial isometry law and quadratic cost law. Reference [64] shows that all the above laws can be unified in one general formulation. In section 5.4.2 we show that the converged solution for the ILC laws is dependent of the choice of the input for the initial run. And we show that all three ILC laws converge to the same solution when using the same initial run, i.e. the same value of γ.

Section 5.4.3 shows the dependence of the final converged control action $u(k)$ on the choice of the input for the initial run is very small. Hence, ILC easily converges to a very well behaved solution after deleting the requisite number of initial rows. It is also shown that if the input in the initial run is zero, or if it is made zero near the end of the initial run, then ILC essentially converges to the Moore-Penrose pseudoinverse solution.

One might expect to need very precise choice of the initial conditions to ensure the value of C_2 is sufficiently small to avoid unstable behavior. From this thinking the results here are counter-intuitive. It becomes evident that the choice of initial run needed to produce

anything close to the unstable behavior of the system inverse is something that one would never think of using. Section 5.4.4 examines that the influence of the initial run on the converged error on the unaddressed time step is very small.

The γ Parameter Set of All Possible Solutions to the Underspecified Equations

First, consider some properties of a generalized inverse of a rectangular matrix P_d modeling an underdetermined system after the deletion of the first row(s) in P, i.e. for the 3^{rd} order system, there is one more control action than the number of errors being addressed. Partition the SVD of matrix P_d using the system $\underline{y}_d^* = P_d \underline{u}$ for simplicity and considering one zero outside the unit circle

$$\underline{y}_d^* = U_d \begin{bmatrix} S_d & 0 \end{bmatrix} \begin{bmatrix} V_{d1}^T \\ V_{d2}^T \end{bmatrix} \underline{u} \tag{5.6}$$

Denote $V_d^T \underline{u} = \underline{\hat{u}}$, then $\underline{\hat{u}}_1 = V_{d1}^T \underline{u} = {S_d}^{-1} {U_d}^T \underline{y}_d^*$,

$$\underline{\hat{u}}_2 = V_{d2}^T \underline{u} = \gamma$$

, and γ could be any scalar as it lies in the null space. Then converting to the original $\underline{u}$ space, one gets

$$\begin{aligned} \underline{u} &= \begin{bmatrix} V_{d1} & V_{d2} \end{bmatrix} \begin{bmatrix} {S_d}^{-1} {U_d}^T \underline{y}_d^* \\ \gamma \end{bmatrix} \\ &= P_d^\dagger \underline{y}_d^* + V_{d2} \gamma \end{aligned} \tag{5.7}$$

The first term represents the Moore-Penrose pseudo-inverse result minimizing the Euclidean norm of the control action. The second term gives all possible solutions by choice of all possible values of γ. The pseudo-inverse solution given by JiLLL NI, is a particularly attractive solution since it produces the smallest possible Euclidean norm of the control to accomplish the zero error at the time steps addressed. Of course there also exists a γ producing zero error at all time steps, i.e. producing the true inverse solution of Equation 4.7, which contains the exponentially growing unstable control action.

The γ Values for Each ILC Law as a Function of the Initial ILC Run

Since we use modified ILC laws to conerge to one of these solutions that give zero error at addressed steps, it is of interest to know what γ value is produced upon convergence as a function of $\underline{u}_0$ for each choice of ILC law.

P Transpose Law: Plug $L = \phi V_d U_d^T$, then the control action updates according to $\underline{u}_{j+1} = \underline{u}_j + \phi P_d^T \underline{e}_{j,d} = (I_p - \phi P_d^T P_d)\underline{u}_j + \phi P_d^T (\underline{y}_d{}^* - \underline{d}_d)$, where the subscript d denotes deleting the first row or entry in a matrix or a vector. Apply the SVD of matrix P_d and partition $V_d^T \underline{e}_{j+1}$ into the part that learns and the part not being updated in the iteration process, and one calculates the decoupled solution for the control action in the new space

$$\begin{bmatrix} V_{d1}{}^T \\ V_{d2}{}^T \end{bmatrix} \underline{u}_{j+1} = \begin{bmatrix} I_{p-1} - \phi S_d^2 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_{d1}{}^T \\ V_{d2}{}^T \end{bmatrix} \underline{u}_j + \phi \begin{bmatrix} S_d \\ 0 \end{bmatrix} U_d^T \left(\underline{y}_d^* - \underline{d}_d \right) \quad (5.8)$$

Denote ${V_{d1}}^T \underline{u}_{j+1} = \underline{\tilde{u}}_{d,j+1}$, $M = I_{p-1} - \phi S_d^2$ and $W_d = \phi S_d U_d^T (y_d^* - d_d)$, then the learned control action in the new space is governed by

$$\underline{\tilde{u}}_{d,j+1} = M \underline{\tilde{u}}_{d,j} + W_d \tag{5.9}$$

whose solution is

$$\underline{\tilde{u}}_{d,j} = M^j \underline{\tilde{u}}_{d,0} + (I_{p-1} - M)^{-1} \left(I - M^j\right) W_d \tag{5.10}$$

Since one is free to pick the learning gain ϕ in matrix M, we consider $\phi = 1/\sigma_{\max}$ is a reasonable choice where $\sigma_{\max}$ denotes the maximum singular value of the P matrix. As the iteration number j goes to infinity, the learned part of the control action expressed in the converted space converges to

$$\underline{\tilde{u}}_{d,j} = S_d^{-1} U_d^T \left(\underline{y}_d^* - \underline{d}_d\right) \tag{5.11}$$

Meanwhile, solving for the unlearned part and convert it back to the original space produces

$$\underline{u}_\infty = P_d^\dagger \left(\underline{y}_d^* - \underline{d}_d\right) + V_{d2} V_{d2}^T \underline{u}_0 \tag{5.12}$$

Comparing Equation 5.12 to Equation 5.7, we conclude that $\gamma = V_{d2}^T u_0$ for this learning law.

Partial Isometry Law: The control updates according to $\underline{u}_{j+1} = \underline{u}_j + \phi V_d U_d^T \underline{e}_{j,d}$. Perform the same operation as above, i.e. partitioning the converted control $\tilde{\underline{u}}_j = V_d^T \underline{u}_j$ into the learned and the unlearned parts as $\tilde{\underline{u}}_{d,\infty} = V_{d1}^T \underline{u}_j$ and $\tilde{\underline{u}}_{f,\infty} = V_{d2}^T \underline{u}_j$. Then one obtains 5.8 with minor changes from S_d^2, S_d to S_d, 0 respectively. After performing the same calculation for the learned and the unlearned parts and converting back to the original space, one concludes that the value of γ is identical for this ILC law as for the P Transpose Law, and the converted control history is therefore also identical.

Quadratic Cost Law: The control updates as $\underline{u}_{j+1} = \underline{u}_j + \phi(P_d^T P_d + rI)^{-1} P_d^T \underline{e}_{j,d}$. Given the identity $(X+Y)^{-1} = X^{-1} - X^{-1}Y(I + X^{-1}Y)^{-1}X^{-1}$ and the use of SVD of $P_d = U_d \begin{bmatrix} S_d & 0 \end{bmatrix} V_d^T$, the inverse term in the middle can be expressed as $\left(P_d^T P_d + rI\right)^{-1} = V_d diag \begin{pmatrix} (s_1^2 + r)^{-1} & \dots & (s_{p-1}^2 + r)^{-1} & r^{-1} \end{pmatrix} V_d^T$, where s_i denotes the i^{th} singular value in matrix S_d. Then plug into the control updating equation and again convert to the $\tilde{\underline{u}}_j$ space, and partition to the learned $\tilde{\underline{u}}_{d,\infty}$ and unlearned $\tilde{\underline{u}}_{f,\infty}$ by premultiplying $\underline{u}_j$ with V_{d1}^T and V_{d2}^T. One gets a modified version of 5.8 with ϕS_d^2 changed to $diag \begin{pmatrix} s_1^2(s_1^2 + r)^{-1} & \dots & s_{p-1}^2(s_{p-1}^2 + r)^{-1} & 0 \end{pmatrix}$, and ϕS_d changed to $diag \begin{pmatrix} s_1(s_1^2 + r)^{-1} & \dots & s_{p-1}(s_{p-1}^2 + r)^{-1} & 0 \end{pmatrix}$. Again, one gets the same control action as 5.11 and the same value of γ.

It is interesting to note that when the iteration number goes to infinity, the control actions produced by all three ILC laws are identical, given by JiLLL NI pseudoinverse solution, plus the same term as a function of the initial iteration's choice of command.

The Influence of the Initial Run on the Converged Final Control History

For the 3^{rd} order discrete-time system, V_{d2} is a unit vetor with magnitudes of the components growing linearly on a log scale as shown in Figure 4.4. Consider a one-second length trajectory, and 100 Hz sample rate, then the magnitude of the first component has an approximate magnitude of 10^{-50} based on linear extrapolation, since Matlab is not able to compute this number. Then it grows exponentially up to the magnitude of 10^{-1} for the last component. Therefore the latter part of $\underline{u}_0$ contributes more to the value of $\gamma = V_{d2}^T \underline{u}_0$. For a reasonable choice of the control action (the initial command) $\underline{u}_0$, pre-multiplying it by a unit vector with such a property, one sees no reason to think that the resulting value of γ could be large.

Also note, that one multiplies γ by V_{d2} to produce the influence on the converged control action. Hence, the influence of u_0 on the control action is given by multiplying this initial control history by the outer product $V_{d2}V_{d2}^T$. Figure 5.1) gives a carpet plot of the matrix entries vs. row and column number, on a linear scale. Obviously, only the last five entries in the rows and columns have much influence on the converged control action. Figure 5.2 further illustrates this where the magnitudes of the matrix entries given on a log scale. The planar surface in the back corner of the plot ending at the 100 by 100 entry, represents correct matrix entries. As the row entries are decreased the carpet plot leaves the planar surface when the computed entries get below approcimately 10^{-16} to 10^{-17}, and the same happens for the column entries. This corresponds to the fact that Matlab is unable to compute these entries accurately. When both the rows and the columns are too far from the back corner

of the plot, Matlab has trouble in two ways, the computed matrix entries are all stuck in the range of 10^{-33}.

Consider some of the implications:

- If one wants γ to precision of 4 significant digits, the last 5 components in V_{d2} is enough. For purposes of illustration, consider that $\underline{u}_0$ has all components equal to unity. Figure 5.3 shows the accumulation of γ, adding the terms of the inner product $\gamma = V_{d2}^T \underline{u}_0$ together, progressing from step one to the final step. When entry p is reached one has the actual value of γ. Note that only the last few time steps matter. Because the entries in V_{d2} alternate in sign, the figure also gives the corresponding result if one makes all entries in $\underline{u}_0$ have unit magnitude and alternate sign. But this does not produce a significantly different result.
- Therefore, one could use any desired input during the initial run, e.g. aiming to get close to zero error at all addressed time steps from the start, but then make the command decay to near zero for the last 5 entries. Then the final converged control action would be very close to the Moore-Penrose pseudoinverse.
- Alternatively, if possible, one could simply use the initial command as identically zero, in which case any of the ILC laws will converge to the Moore-Penrose pseudoinverse, corresponding to the minimum Euclidean norm control action to produce zero error at the addressed time steps.

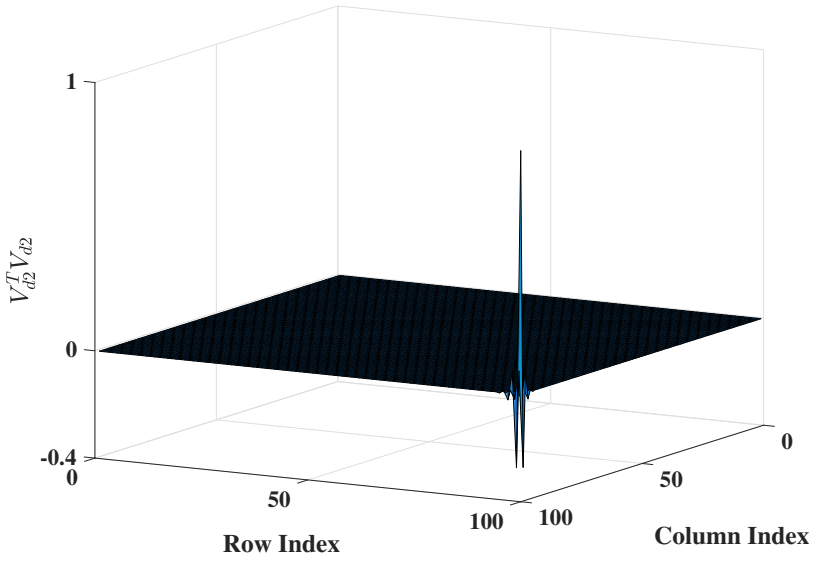

Figure 5.1: Matrix entries of $V_{d2}V_{d2}^T$ showing the influence of initial input components of $\underline{u}_0$ on control action

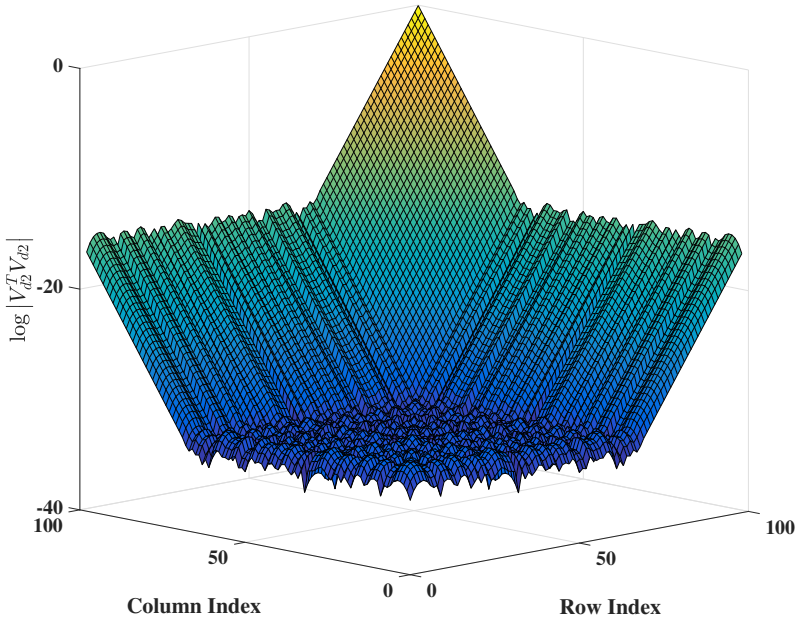

Figure 5.2: Logrithm of Magnitude of matrix entries $V_{d2}V_{d2}^T$

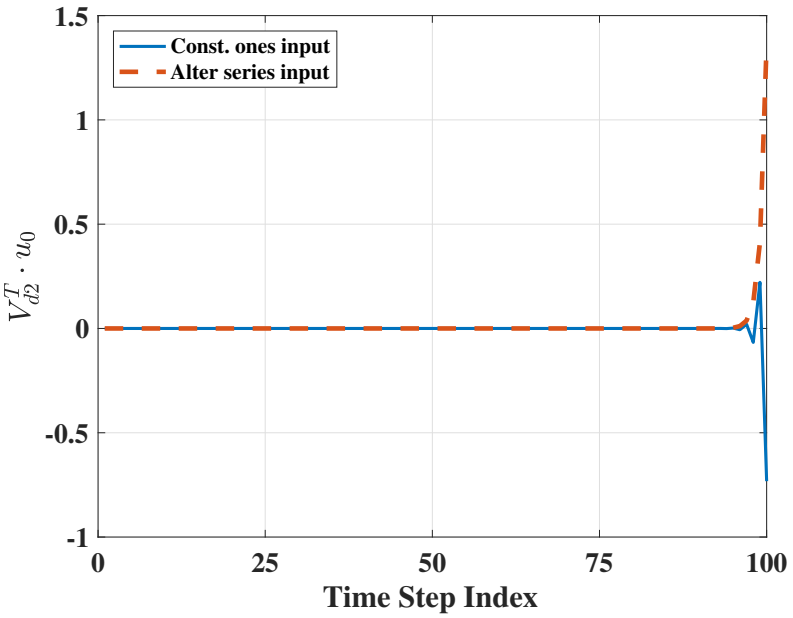

Figure 5.3: Illustration of how the value of γ accumulates as time steps progress

The Influence of the Initial Run on the Converged Error of the Unaddressed First Time Step

The methods used here have avoided the difficulty of inverting an ill-conditioned matrix and avoided the use of an unstable control action. But this was accomplished at the expense of not being able to get zero error in the first time step. It is of interest to ask what happens to the error for this time step(s). The final level of error is given by $\underline{e}_\infty = \underline{y}^* - P\underline{u}_\infty - \underline{d}$, which produces zero at all time steps but the first which gives

$$\begin{aligned} \underline{e}_\infty(1) &= \underline{y}^*(1) - P_f\underline{u}_\infty - \underline{d}(1) \\ &= \underline{y}^*(1) - P_f P_d^\dagger \left(\underline{y}_d^* - \underline{d}_d\right) - \underline{d}(1) - P_f V_{d2} V_{d2}^T \underline{u}_0 \end{aligned} \quad (5.13)$$

where $P_f = \begin{bmatrix} CB & 0 & \dots & 0 \end{bmatrix}$ is the first row in the P matrix. The first three terms on the right hand side are pre-determined by system dynamics, the desired command

trajectory, and the repeated disturbance during the iteration process. The only free choice is $\underline{u}_0$. Again it is premultiplied by the matrix whose entries are studied. But this time it is further premultiplied by the P_f matrix of all zeros except for CB. Recall that the discrete-time B is roughly equal to the continuous time B times the sample time interval, in this case 0.01 sec. So CB should not be a large number. It is very difficult to move this initial value of the error, but it also seems true that the error is not likely to be a large number.

Previously we discussed the unstable inverse from the point of view of the initial conditions determining the coefficient of C_2 of the unstable solution $(-3.104)^k$. From this point of view one might think that one would have to be very careful with the initial condition in order to make the unstable term be near zero. Let us investigate if there is any need to be careful.

- First, observe how hard it is to influence the error at the first time step, and still maintain zero error at later steps. The additional initial control action Δu_0 necessary to make a change $\Delta e_\infty(1)$ satisfies

$$\Delta \underline{e}_\infty(1) = -P_f V_{d2} V_{d2}^T (\Delta \underline{u}_0) = -(CB) \cdot V_{d2}(1) \cdot \Delta\gamma \tag{5.14}$$

  Numerically, Equation 5.13 says that in order to make one unit change $\Delta \underline{e}_\infty(1)$ in the first time step error, $\Delta\gamma$ must have a magnitude of approximately 10^{57} with a negative sign in front.

- To produce a given desired change in $\Delta \underline{e}_\infty(1)$ there needs to be a correspondingly

large change in the control action given by

$$\underline{u}_\infty{}' = P_d^\dagger \left(\underline{y}_d^* - \underline{d}_d \right) + (\gamma + \Delta\gamma) V_{d2} \tag{5.15}$$

.

- It is conceivable that the Moore-Penrose pseudoinverse contains a nonzero value of C_2 when minimizing the Euclidean norm of the control action, but the value must be of the order of $1 \Big/ (z_2)^k$ so that no large control is accumulated from this term in p time steps. One might prefer to have C_2 identically zeros, if one could find a way to do this.
- The instability of the control input producing zero error at the addressed time steps is then included in the $V_{d2}\gamma$ term in Equation 5.12. Since V_{d2} is a unit vector, in order for the unstable history in the V_{d2} vector to have substantial influence on the control history, one needs a substantial value for γ, i.e. one needs a substantial component of $\underline{u}_0$ on V_{d2}.
- For a given magnitude $\underline{u}_0$ the maximizing choice is to pick the initial input to be equal to V_{d2}, i.e. pick an unstable control input. One is not likely to do this, but in addition this only produces a value of $V_{d2}^T V_{d2} = 1$. Therefore, one also needs to make the control u_0 to be a large number multiplying V_{d2}, in order to generate an initial input that has substantial instability observed within the given p time steps of the problem.

5.5 Stable Inverse Theorem Based Iterative Learning Control

In this section, the stable inverse theorem based ILC approach is sketched according to [13], for the purpose of comparison of differences and similarties. Because the stable inverse problem is defined on the infinite time interval $(-\infty, +\infty)$, the authors pre- and post-extended the original finite time desired output trajectory $y_d(t), \quad t \in [0, T]$ as

$$y_r(t) = \begin{cases} 0, & t \in (-\infty, t_0] \\ r_1(t), & t \in [t_0, 0] \\ y_d(t), & t \in [0, T] \\ r_2(t), & t \in [T, t_f] \\ 0, & t \in [t_f, +\infty] \end{cases} \tag{5.16}$$

where

$$r_1(t_0) = 0, \quad r_1(0) = y_d(0), \quad r_1(T) = y_d(T), \quad r_1(t_f) = 0 \tag{5.17}$$

$r_1(t)$, $r_2(t)$ are continuously differentiable. Then the problem of finding the stable inverse solution maintaining the exact output trajectory which satisfies the system

$$\begin{cases} \dot{x}_j(t) = Ax_j(t) + Bu_j(t) \\ y_j(t) = Cx_j(t) \end{cases} \tag{5.18}$$

where $x_j(t) \in \Re^n$, $u_j(t) \in \Re$, $y_j(t) \in \Re$, could be converted to an optimal tracking problem

as

$$\begin{aligned} &\min_{u} \ J(u) = \tfrac{1}{2}\int_{-\infty}^{+\infty} e^T(t)e(t)dt \\ &s.t. \quad e(t) = y_r(t) - y(t) \\ &\qquad y(t) = Pu(t) \\ &\qquad u(\pm\infty) = 0, \quad x(\pm\infty) = 0, \quad y(\pm\infty) = 0 \end{aligned} \tag{5.19}$$

Now using continuous time *Lagrange* multipliers $\lambda(t)$ and gradient-based optimization algorithm to solve Equation 5.18. Define the *Hamiltonian* $H(x, u, \lambda, t)$ as

$$H(x, u, \lambda, t) = \frac{1}{2}(y_r - Cx)^T(y_r - Cx) + \lambda^T(Ax + Bu) \tag{5.20}$$

The first-order necessary condition gives the costate equation as

$$\dot{\lambda}(t) = -\partial_{x_j} H[x_j, u_j, \lambda, t] = -A^T\lambda(t) + C^T(y_r(t) - y_j(t)) \tag{5.21}$$

To make the performance index $J(u)$ sliding towards in the decreasing direction, the control action based on the counterpart in the previous repetition updates along the direction

$$\delta u_j = -\partial_{u_j} H[x_j, u_j, \lambda, t] = -B^T\lambda(t) \tag{5.22}$$

Then the control action when the iteration number $j \to \infty$ would converge to the stable inverse theorem based solution as modeled

$$u_{j+1}(t) = u_j(t) + \phi z_j(t) \tag{5.23}$$

where ϕ is learning rate, and $z_j(t)$ could be solved by

$$\begin{cases} \dot{\lambda}_j(t) = -A^T\lambda_j(t) + C^T(y_r(t) - y_j(t)) \\ z_j(t) = -B^T\lambda_j(t) \end{cases} \tag{5.24}$$

constraint by the boundary conditions $\lambda_j(\pm\infty) = 0$. Again, as mentioned previously in the stable inverse theorem section, this calculation needs two integrals from both $-\infty$ and $+\infty$, or by an infinite summation in discrete time scenario. Reference [8] points out that there is no direct relation between ILC law using adjoint system as outlined above, and stable inverse theorem. However, they both converge to the same solution when one chooses the control action of the initial run as zero, which coincides with the solution done by JiLLL NI. The author here finds this is the most interesting part of this comparison.

5.6 Conclusions

The JiLLL NI solution to an inverse discrete-time control problem produces zero error at all time steps except the first few (one step for example the 3^{rd} order problem (or pole excess 3), 2 time steps if the problem were 5^{th} order, 3 if it were 7^{th} order). The inverse problem is inverting the associated difference equaiton, which has a zero(s) outside the unit circle that becomes an unstable pole(s) outside the unit circle. The resulting difference equation that must be satisfied has a solution of the associated homogeneous equation that is an arbitrary constant(s), determined by initial conditions, times the unstable solution(s). In order to have a stable inverse, it would seem necesary to set the initial conditions very precisely so that there would be a zero coefficient multiplying the unstable solution. We show that this is

not necessary, and in fact it is very hard to set initial conditions that exhibit the instability.

JiLLL NI gives a p time step control history producing the desired output at all time steps but the first (or first few). Then the control action at the first time step is determined by the Moore-Penrose pseudoinverse. ILC wants to similarly converge to zero error, but cannot use this pseudoinverse solution because it seeks zero error in the unknown world model, instead of our model of the world. The set of all possible pseudoinverses is established. It is determined that all 3 ILC laws converge to the same pseudoinverse solution, when given the same initial input used in the first ILC iteration. It is determined what choice of the initial run is required to produce the actual unstable inverse for all time steps, and also it is determined what kind of initial run is needed to result in any significant unstable control action. It is clear that one would never pick such an initial input history.

If one wants to reduce the very small influence of a reasonable initial run on the ILC converged zero error control history, one can make the initial control action decay to zero for the last few time steps. If one wants to go further, and have ILC converge to the Moore-Penrose minimum Euclidean norm solution, then one can give a zero command to the control system for the entire first run. In this case, the ILC converges to the minimum Euclidean norm control solution of the Longman Stable Inverse JiLLL NI.

Since the ILC does not consider the error in the first time step, the error at this time step is studied. One might worry that something wild must be done at this time step in order to produce the stable inverse for the remaining steps. It appears that this not the case. Furthermore, requesting any significant change in the error at this first time step while keeping the remaining errors zero, requires introducing unstable control time histories.

The final conclution is that all three ILC laws will converge to well behaved and very

useful solutions to the inverse control problem using any reasonable intial runs, and that the ill-conditioning and the instability of the inverse model are eliminated.

This page intentionally left blank.

Conclusion

This work develops a series of new stable inverses of linear discrete time systems. Having a stable inverse to make use of, this addresses a basic problem and has the potential to address difficulties in many control design problems. The work was motivated by solving the inverse instability issue in Iterative Learning Control (ILC) problems.

Typical feedback control systems do not do what you ask them to do. The concept of bandwidth is created to describe up to what frequency such a system will do something reasonably close to the command. Various control approaches aim to fix this problem, and produce zero tracking error following the commanded trajectory. These include Iterative Learning Control (ILC), Repetitive Control (RC), one step ahead control, indirect adaptive control, etc. Each aims to produce that input command that produces the desired output, i.e. solve the inverse problem. To implement such control laws, one must use discrete time models which represent the continuous world with inputs coming through a zero-order-hold. The exact discrete time equivalent, giving the same output at sample times as the differential equation, very often has Non-Minimum Phase (NMP) zeros, including sampling zeros, or intrinsic zeros, or both. This means that the inverse problem is unstable, i.e.

finding the input necessary to produce the desired output results in a control action that grows exponentially in magnitude with time.

With the knowledge of asymptotic locations of sampling NMP zeros, this work picks a representative discrete time model of a robot at NASA Langley Research Center. The analysis starts from detecting the instability in a discrete time difference equation model and a finite time matrices model, including a state space model and an ARX model. Then creates a factorization of systems matrices, modeling finite time dynamics of the input-output linear mapping of the discrete time system as the analogy to a transfer function. Based on two versions of the modified problem statements, a series of new stable inverses are developed. Then apply the new stable inverse ideas to solve the instability issue in ILC problems, justifying that all major learning laws converge to the same well behaved and useful solutions as well as being insensitive to the choice of control action in the initial iteration to start the learning procedure. This work also studies the commonalities and differences between the existing stable inverse theorem, and its application on ILC problems.

Applications of new stable inverses include. (1) ILC was the original motivation for the development of the stable inverse concepts presented in this work. ILC attemps to converge to the full inverse control action. Hence, it is very often trying to converge to an unstable control action. Often in applications people do not realize. The instbiality may noe be evident for many iterations, and could also be stabilized by the analog to digital converters. For a slow sample rate it can become evident earlier, and then perople wonder what is wrong. This work creates stable inverse methods to address this problem. This work also explains why ILC, without actually applying the stable inverse, without making any use of

the prescribed pseudoinverse, nevertheless converges to the right answer when using initial deletion for all major ILC laws, at meantime insensitive to the choice of control action at the intial iteration. (2) ILC design with local learning: Imagine a factory robot repeatedly starts from a home position, going to a newly arrived object where it performs a high precision task, and then returns home. High accuracy tracking is only needed for the task part of the trajectory. One could consider using a quadratic cost control in general, but phasing into the skip step stable inverse for some portion of the trajectory that needs high precision tracking. (3) One step ahead control is a digital feedback control law that aims to produce zero error every time step, and one of the major limitation is the inverse instability for any system with the relative degree of three or more. One could apply the stable inverse idea by increasing the sample rate to produce the equivalent of a generalized hold between the original sample times, and study the tracking error between the original sample times. (4) Indirect discrete time adaptive control creates a law promising to converge to zero tracking error in real time. The basic indirect adaptive control relies on the one-step ahead control law followed by the projection algorithm to update the model based on the current inverse. The asymptotic stability of the inverse of the system required by convergence condition is not satisfied in a majority of systems one would like to control. Preliminary study has been done to solve this issue by applying stable inverse ideas. (5) Note that one could reformulate the above control design problems as Linear Model Predictive Control that updates a batch of time steps instead of every step. This offers the ability to converge to zero tracking error at every time step of the skip step inverse, instead of the usual aim to converge to a quadratic cost solution.

There are still more things to explore in the future and some of them are ongoing. (1)

The author intends to generalize stable inverses to multi-input, multi-output (MIMO), linear time varying or nonlinear discrete time systems. (2) Examine in a more detailed level on errors in between sample times to evaluate overall performance, via the study on the degree of freedom at initial time steps driving the system onto the desired state trajectory. (3) Make the current study on indirect adaptive control on real-time via the enhancement on stable inverses. (4) Generalize the stable inverses on systems with partial prior knowledge of system dynamics.

References

[1] K.Astrom, Hagander, P., and Strenby, J., 1980. "Zeros of sample systems". *Proceedings of the 19th IEEE Conference on Decision and Control*, pp. 1077–1081.

[2] Blachuta, M. J., 1997. "On zeros of sampled systems". In *American Control Conference* (Albuquerque, NM, 07), Vol. 5.

[3] Devasia, S., Chen, D., and Paden, B., 1996. "Nonlinear inversion-based output tracking". *IEEE Transactions on Automatic Control,* ***41***(7), 07, pp. 930–930.

[4] Hunt, L. R., Meyer, G., and Su, R., 1996. "Noncausal inverses for linear systems". *IEEE Transactions on Automatic Control,* ***41***(4), 04.

[5] Hunt, L. R., and Meyer, G., 1997. "Stable inversion for nonlinear systems". *Automatica,* ***33***(8), pp. 1549–1554.

[6] Zou, Q., and Devasia, S., 1999. "Preview-based stable-inversion for output tracking of linear systems". *ASME Journal of Dynamic Systems, Measurement and Control,* ***12***, pp. 625–630.

[7] Qingze Zou, 2007. "Optimal preview-based stable-inversion for output tracking of nonminimum-phase linear systems". In *2007 46th IEEE Conference on Decision and Control* (Dec), pp. 5258–5263.

[8] Kinosita, K., Sogo, T., and Adachi, N., 2002. "Iterative learning control using adjoint systems and stable inversion". *Asian Journal of Control,* ***4***(1), pp. 60–67.

[9] Cai, Z., Freeman, C., Rogers, E., and Lewin, P., 2007. "Reference shift iterative learning control for a non-minimum phase plant". In *American Control Conference* (New York, NY, 07), pp. 558–563.

[10] Koshy, G., Verhaegen, M., and Scherpen, J., 1999. "Stable inversion of mimo linear discrete time non-minimum phase systems". In *Proceedings of the 7th Mediterranean Conference on Control and Automation* (07), pp. 267–281.

[11] Jeong, G.-M., and Choi, C.-H., 2002. "Iterative learning control with advanced output data". *Asian Journal of Control,* ***4***(1), 03, pp. 30–37.

[12] T.Sogo, 2002. "Stable inversion for nonminimum phase sampled-data systems and its relation with the continuous-time counterpart". In *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.* (Dec), Vol. 4, pp. 3730–3735 vol.4.

[13] S.Liu, and T.Wu, 2003. "Stable-inversion based iterative learning control for non-minimum phase systems". *Control Theory and Applications,* ***20***, 12.

[14] Ye, L., Zong, Q., and Wang, F., 2017. "Tracking control of nonminimum phase systems: an overview". *Control Theory and Applications,* ***34***, 02.

[15] Zhang, Y., and Liu, S., 2016. "Non-causal stable inversion based on optimal state to state transition". *Control Theory and Applications,* ***33***, 12.

[16] Ji, X., and Longman, R. W., 2019. "New results for stable inverses of discrete time systems". In *Proceedings of the 19th Yale Workshop on Adaptive and Learning Systems* (New Haven, CN, 06), Narendra, ed., pp. 558–563.

[17] Ji, X., and Longman, R. W., 2017. "Proof of two new stable inverses of discrete time systems". *AIAA/AAS Astrodynamics Specialist Conference*, August.

[18] Ji, X., and Longman, R., 2018. "The insensitivity of the iterative learning control inverse problem to initial run when stabilized by a new stable inverse". In *Modeling, Simulation and Optimization of Complex Processes: The 7-th International Conference on High Performance Scientific Computing* (Hanoi, Vietnam, 03), H. Bock, E. Kostina, X. Phu, and R. Rannacher, eds., Springer.

[19] UCHIYAMA, M., 1978. "Formulation of high-speed motion pattern of a mechanical arm by trial". *Transactions of the Society for Instrumentation and Control Engineers,* ***14***, pp. 706–712.

[20] Arimoto, S., Kawamura, S., and Miyazaki, F., 1984. "Bettering operation of robots by learning". *Journal of Robotic Systems,* ***1***(2), pp. 123–140.

[21] CRAIG, J., 1984. "Adaptive control of manipulators through repeated trials". In *Proceedings of the American Control Conference* (06), pp. 1566–1573.

[22] MOORE, K., 1999. "Iterative learning control - an expository overview". *Applied and Computational Controls, Signal Processing, and Circuits,* ***1***(1), pp. 151–241.

[23] XU, J., 2002. "The frontiers of iterative learning control - part i". *Journal of Systems, Control and Information,* ***46***(2), pp. 63–73.

[24] XU, J.-X., 2002. "The frontiers of iterative learning control - part ii". *Journal of Systems, Control and Information,* ***46***(5), pp. 233–243.

[25] BRISTOW, D., THARAYIL, M., and ALLEYNE, A., 2006. "A survey of iterative learning control". *IEEE Control Systems Magazine,* ***26***(3), pp. 96–114.

[26] MOORE, K., 1993. "Iterative learning control for deterministic systems". In *Advances in Industrial Control* (London, U.K, 07), Springer, ed.

[27] MOORE, K., and XU, J.-X., 2003. *Linear and Nonlinear Iterative Learning Control*. Springer, Berlin.

[28] MOORE, K., and XU, J., 2000. "Editorial: Iterative learning control". *International Journal of Control,* ***73***(10).

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

[30] Longman, R. W., 2000. "Iterative learning control and repetitive control for engineering practice". *International Journal of Control,Special Issue on Iterative Learning Control,* ***73***, pp. 930–954.

[31] Longman, R. W., 2010. "On the theory and design of linear repetitive control systems". *European Journal of Control, Special Section on Iterative Learning Control,* ***16***, pp. 447–496.

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

[33] AHN, H.-S., and BRISTOW, D., 2011. "Special issue on iterative learning control". *Asian Journal of Control,* ***13***(1).

[34] Zhou, W., and Longman, R. W., 2018. "Root locus of zeros of discrete time systems as a function of sample rate". In *Advances in the Astronautical Sciences* (07), Vol. 162.

[35] Zhang, T., and Longman, R. W., 2018. "Repetitive control design for the possible digital feedback control configurations". In *Advances in the Astronautical Sciences* (07), Vol. 129.

[36] Ji, X., and Longman, R., 2018. "Proof of two stable inverses of discrete time systems". In *Advances in the Astronautical Sciences* (Columbia River Gorge, Stevenson, WA, 07), Vol. 162, pp. 123–136.

[37] Ji, X., and Longman, R. W., 2019. "New stable inverses of discrete time systems". In *Advances in the Astronautical Sciences* (Portland, Maine, 07).

[38] Longman, R., and Li, T., 2017. "On a new approach to producing a stable inverse of discrete time systems". In *Proceedings of the 18th Yale Workshop on Adaptive and Learning Systems* (New Haven, CN, 07), Narendra, ed.

[39] Li, Y., and Longman, R. W., 2008. "Addressing problems of instability in intersample error in iterative learning control". In *Advances in the Astronautical Sciences* (07), Vol. 129.

[40] Li, Y., and Longman, R. W., 2010. "Using underspecification to eliminate the usual instability of digital system inverse models". In *Advances in the Astronautical Sciences* (07), Vol. 135.

[41] LeVoci, P., and Longman, R., 2004. "Intersample error in discrete time learning and repetitive control". In *Proceedings of the 2004 AIAA/AAS Astrodynamics Specialist Conference* (Providence, RI, 08), pp. 1967–1985.

[42] Chen, K., and Longman, R., 2003. "Creating short time equivalents of frequency cutoff for robustness in learning control". In *Advances in the Astronautical Sciences* (07), Vol. 114, pp. 95–114.

[43] Song, B., and Longman, R. "Circulant zero-phase low pass filter design for improved robustification of iverative learning control". In *Advances in the Astronautical Sciences*, Vol. 156, pp. 2161–2180.

[44] Smith, G., 1978. *Numerical Solution of Partial Differential Equations*. Clarendon Press, Oxford.

[45] Longman, R., and Juang, J., 1988. "A variance based confidence criterion for era identified modal parameters". In *Advances in the Astronautical Sciences* (), Vol. 65, pp. 581–601.

[46] WEN-CHYUAN, Y., and SUN, C. S., 2008. "Explicit eigenvalues and inverses of tridiagonal toeplitz matrices with four perturbed corners". *The ANZIAM Journal,* ***49***(3), p. 361–387.

[47] Bristow, D. A., and Singler, J. R., 2009. "Analysis of transient growth in iterative learning control using pseudospectra". In *Proceedings of the Symposium on Learning Control at IEEE CDC 2009* (Shanghai, China, 08), Vol. 136.

[48] Isidori, A., and Byrnes, C. "Output regulation of nonlinear systems". *IEEE Transactions on Automatic Control,* ***35***(2), pp. 131–140.

[49] Isidori, A., 1985. *Nonlinear Control Systems*. Springer, London.

[50] Zhu, J., and Longman, R., 2017. "Repetitive model predictive control based on markov parameters". In *Proceedings of the 2017 AAS/AIAA Spaceflight Mechanics Conference* (San Antonio, TX, 02), pp. 267–281.

[51] Zhu, J., and Longman, R., 2018. "Iterative learning control design with local learning". In *Advances in the Astronautical Sciences* (08), Vol. 162, pp. 47–61.

[52] Wang, B., and Longman, R., 2018. "Generalized one step ahead control made practical by new stable inverses". In *Proceedings of the AIAA/AAS Space Flight Mechanics Conference* (Kissimmee, FL, 01), pp. 123–136.

[53] Wang, B., and Longman, R., 2018. "On the development of indirect adaptive p-step ahead control of systems with unstable discrete time inverse". In *Modeling, Simulation and Optimization of Complex Processes: Proceedings of the International Conference on High Performance Scientific Computing* (Hanoi, Vietnam, 03), H. Bock, E. Kostina, X. Phu, and R. Rannacher, eds., Springer, pp. 123–136.

[54] Chen, K., Longman, R., and Phan, M., 2006. *On the Relationship Between Repetitive Control and Model Predictive Control*. Keystone, CO, Aug.

[55] Goodwin, G. C., Ramadge, P. J., and Caines, P. E., 1980. "Discrete-time multivariable adaptive control". *IEEE Transactions on Automatic Control,* ***AC-25***, pp. 449–456.

[56] Gao, F., and Lonamgn, R., 2013. "Examing the learning rate in iterative learning control near the end of the desired trajectory". In *Proceedings of the AAS/AIAA Astrodynamics* (07).

[57] Amann, N., Owens, D., and Rogers, E., 1995. "Robustness of norm-optimal iterative learning control". In *Proceedings of International Conference on Control* (Exeter, UK, 07), pp. 1119–1124.

[58] AHN, H.-S., CHEN, Y., and MOORE, K., 2007. "Iterative learning control: brief survey and categorization". *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews,* ***37***(6), pp. 1099–1122.

[59] Longman, R., Alnajjar, K., and Ji, X., 2014. "Comments on how a new engineering field develops: A case study from iterative and repetitive control". In *Proceedings of the 2nd International Conference on Intelligent Technologies and Engineering Systems* (12), J. Juang, C. Chen, and C. Yang, eds., Vol. 293, Springer, pp. 1273–1279.

[60] Walker, M., Ji, X., and Longman, R., 2018. "The extreme behavior of the simplest form of iterative learning control". In *Proceedings of the 2018 AAS/AIAA Spaceflight Mechanics Conference* (07).

[61] Jang, H., and Longman, R. W., 1994. "A new learning control law with monotonic decay of the tracking error norm". In *Proceedings of the Thirty-Second Annual Allerton Conference on Communication, Control and Computing* (Monticello, IL, 09), pp. 314–323.

[62] Jang, H., and Longman, R. W., 1996. "Design of digital learning controllers using a partial isometry". In *Proceedings of the Thirty-Second Annual Allerton Conference on Communication, Control and Computing* (07), Vol. 93, pp. 137–152.

[63] Frueh, J. A., and Phan, M. Q., 1988. "Linear quadratic optimal learning control (lql)". In *Proceedings of the 37th IEEE Conference on Decision and Control* (Tampa, FL, 12), pp. 678–683.

[64] J. Bao, R. L., 2010. "Unification and robustification of iterative learning control laws". *Advances in the Astronautical Sciences,* ***136***, pp. 727–745.

[65] Beigi, H., 1997. "Adaptive and learning-adaptive control techniques based on an extension of the generalized secant method". *Intelligent Automation and Soft Computing Journal,* ***3***(2), pp. 171–184.

[66] Avrachenkov, K. E., Beigi, H., and Longman, R. W., 2002. "Updating procedures for iterative learning control in hilbert space". *Intelligent Automation and Soft Computing Journal (Special Issue on Learning and Repetitive Control),* ***8***(2).

[67] Beigi, H., Li, C., and Longman, R. W., 1991. "Learning control based on generalize secant methods and other numerical optimization methods". In *Sensors, Controls and Quality Issus in Manufacturing* (12), Vol. 55, pp. 163–175.

[68] Beigi, H., 1992. "A parallel network implementation of the generalized secant learning-adaptive controller". In *Proc. of Canadian Conference on Electrical and Computer Engineering* (Toronto, Canada, 9), Vol. 2, pp. 1–4.

[69] Avrachenkov, K., Beigi, H., and Longman, R., 1999. "Operator-updating procedures for quasi-newton iterative learning control in hilbert space". In *IEEE CDC* (Phoenix, AZ, 12).

[70] Beigi, H., 1992. "An adaptive control scheme using the generalized secant method". In *Proc. of Canadian Conference on Electrical and Computer Engineering* (Toronto, Canada, 9), Vol. 2, pp. 1–4.

[71] Avrachenkov, K. E., and Longman, R. W., 2003. "Iterative learning control for over-determined, under-determined, and ill-conditioned systems". *International Journal of Applied Mathematics and Computer Science,* ***13***, pp. 113–122.

[72] Longman, R., Peng, Y., Kwon, T., Lus, H., Betti, R., and Juang, J., 2011. "Adaptive inverse iterative learning control". *Journal of the Chinese Society of Mechanical Engineers,* ***32***(6), pp. 493–506.

[73] Elci, H., Longman, R. W., Minh Phan, Jer-Nan Juang, and Ugoletti, R., 1994. "Discrete frequency based learning control for precision motion control". In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics* (Oct), Vol. 3, pp. 2767–2773 vol. 3.

[74] Li, Y., and Longman, R., 2010. "Characterizing and addressing the instability of the control action in iterative learning control". In *Advances in the Astronautical Sciences* (07), Vol. 136, pp. 1967–1985.

[75] S. J. Oh, R. L., and Phan, M. Q., 1997. "Use of decoupling basis functions in learning control for local learning and improved transients". *Advances in the Astronautical Sciences,* ***96***, pp. 651–670.

[76] Yao, H., and Longman, R. W., 2010. "Frequency response based repetitive control design for linear systems with periodic coefficients". In *Advances in the Astronautical Sciences* (), Vol. 136, pp. 727–745.

This page intentionally left blank.

Appendix A

Numerical Results on ILC of Time Varying Systems

A.1 Introduction

Systems for which one might want to apply ILC to obtain high prevision tracking, can have time varying coefficients. Two cases can apply. One is that the system model is simply time varying. A second situation applies when one wants to use ILC designed for linear systems to apply to nonlinear systems. One can linearize the nonlinear model around the desired trajectory that is a function of time, and this produces time varying coefficients. The linear range around the desired trajectory can be large enough to make this approach effective. If not, one may seek to repeatedly linearize about trajectories during the iterations.

For constant coefficient LTI systems, the Toeplitz matrix of Markov parameters is well studies in the previous chapters. The purpose of this Appendix is to examine how these properties are modified as a time variation is introduced into the coefficients. We study the various classes of time variation. And for the basic laws of contraction mapping, partial isometry, and quadratic cost learning updates can in theory be used directly with time varying systems but no one has addressed the issue of what happens to the troublesome singular values associated with zeros of linear transfer functions outside the unit circle for constant coefficient systems are transformed by time variation. The investigation here seeks whether the methods obtained still address the issue of instability of the inverse.

A.2 On ILC of Linear Time Varying Systems

The Repetition Domain Model

Consider a general linear time varying state variable difference equation model

$$\begin{cases} x_j(k+1) = A(k)x_j(k) + B(k)u_j(k) \\ y_j(k+1) = C(k+1)x_j(k+1) + v(k+1) \end{cases} \tag{A.1}$$

Similarly, by repeatedly writing Equation A.1 for successive values of time step k, and substituting previous equations into the current equation, one can get the convolution sum solution to the difference equation

$$\begin{cases} x_j(k+1) = \prod_{i=0}^{k} A(i)x(0) + \sum_{i=0}^{k} \left[\prod_{l=i+1}^{k} A(l) \right] B(i)u_j(i) \\ y_j(k+1) = C(k+1)x_j(k+1) \end{cases} \tag{A.2}$$

equivalently in matrix as

$$P = \begin{bmatrix} C(1)B(0) & 0 & \cdots & 0 \\ C(2)A(1)B(0) & C(2)B(1) & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ C(p)\left[\prod_{i=1}^{p-1} A(i)\right] B(0) & C(p)\left[\prod_{i=2}^{p-1} A(i)\right] B(1) & \cdots & C(p)B(p-1) \end{bmatrix} \tag{A.3}$$

$$A_O = \begin{bmatrix} C(1)A(0) \\ C(2)A(1)A(0) \\ \vdots \\ C(p)\prod_{i=1}^{p-1} A(i) \end{bmatrix}$$

Relating Nonlinear Ordinary Differential Equations to P Matrix

Consider a general nonlinear ordinary differential equation of the form

$$\begin{cases} \dot{\bar{x}}(t) = f\left(\bar{x}(t), \quad \bar{u}(t)\right) \\ \bar{y}(t) = C\bar{x}(t) \end{cases} \tag{A.4}$$

Suppose that we have a set of functions $u^*(t)$, $y^*(t)$, $x^*(t)$ satisfying these equations, and we would like to have Equation A.4 linearized about these functions. Define the deviations from the desired solution by

$$u(t) = \bar{u}(t) - u^*(t), \quad x(t) = \bar{x}(t) - x^*(t), \quad y(t) = \bar{y}(t) - y^*(t) \tag{A.5}$$

Note that we will be integrating these differential equations across a sample time, and that if the input comes through a zero order hold, both $u(t)$ and $\bar{u}(t)$ would be constants during such an interval. Linearzing the right hand of Equation A.4 about the desired trajectory

$$labeleq : a6 \begin{cases} f\left(\bar{x}(t), \quad \bar{u}(t)\right) \approx f\left(x^*(t), \quad u^*(t)\right) + \frac{\partial f}{\partial \bar{x}(t)}\Big|_* x(t) + \frac{\partial f}{\partial \bar{u}(t)}\Big|_* u(t) \\ \dot{\bar{x}}(t) - \dot{x}^*(t) = f\left(\bar{x}(t), \quad \bar{u}(t)\right) - f\left(x^*(t), \quad u^*(t)\right) \end{cases} \tag{A.6}$$

Therefore, the linear variational differential equations become

$$\begin{cases} \dot{x}(t) = A_C\left(x^*(t), \quad u^*(t)\right) x(t) + B_C\left(x^*(t), \quad u^*(t)\right) u(t) \\ y(t) = Cx(t) \end{cases} \tag{A.7}$$

where

$$A_C\left(x^*(t), \quad u^*(t)\right) = \left.\frac{\partial f}{\partial \bar{x}(t)}\right|_* \quad B_C\left(x^*(t), \quad u^*(t)\right) = \left.\frac{\partial f}{\partial \bar{u}(t)}\right|_* \tag{A.8}$$

Now convert the liner time varying state space differential equation to a difference equation, giving the solution to Equation A.7 at sample times $t = kT$ where k is an integer and T ia the sample time interval. The state transition matrix $\Phi\left(t,\ kT\right)$ is the square matrix solution of

$$\dot{\Phi}\left(t,\ kT\right) = A_C\left(x^*(t), \quad u^*(t)\right)\Phi\left(t,\ kT\right) \tag{A.9}$$

on the interval $kT - \tau - (k+1)T$ starting from the initial condition $\Phi\left(kT,\ kT\right) = 1$. Then the solution of Equation A.7 is given as

$$x(t) = \Phi\left(t,\ kT\right)x(kT) + \int_{kT}^{t} \Phi\left(t,\ \tau\right)B_C\left(x^*(\tau), \quad u^*(\tau)\right)u(\tau)d\tau \tag{A.10}$$

Setting $t = (k+1)T$ and using the fact that the input is constant over a sample time interval produces the desired time varying difference equation

$$\begin{cases} x_j(k+1) = A(k)x_j(k) + B(k)u_j(k) \\ y_j(k+1) = C(k+1)x_j(k+1) + v(k+1) \\ A(k) = \Phi\left((k+1)T, \quad kT\right) \\ B(k) = \int_{kT}^{(k+1)T} \Phi\left((k+1)T, \quad \tau\right) B_C\left(x^*(\tau), \quad u^*(\tau)\right) d\tau \end{cases} \tag{A.11}$$

Of course in the linearized time varying difference equation model, the resultant coefficients are varying with time, for simplicity in this paper, we choose coefficients $A(k)$, $B(k)$ as piecewise constants with reasonable sampling rate.

A Linearization Example

As an example of linearization, consider a first order nonlinear system in continuous time domain

$$\dot{y}(t) + k_1 y^3(t) + k_2 y(t) = u(t) \tag{A.12}$$

Suppose we know the command $u^*(t)$ that produces the output $y^*(t)$ and we wish to linearize the equation about this input-output pair. Then the nonlinear term can be written linearized for small deviations from $y^*(t)$, as $y^3(t) = \left(y^*(t)\right)^3 + 3\left(y^*(t)\right)^2\left(y(t) - y^*(t)\right)$. The linearized equation becomes

$$\dot{y}(t) + \left[3k_1\left(y^*(t)\right)^2 + k_2\right] y(t) = u(t) - k_1\left(y^*(t)\right)^3 + 3k_1\left(y^*(t)\right)^3 \tag{A.13}$$

This is a linear differential equation with time varying coefficient, and it has a repeating forcing term, and the function about which it was linearized. If the command input comes in through a zero order hold, it would be held constant from one time step to the next, and one could recomputed the linearized equation, and convert to a difference equation, but with the repeating forcing term on the input to the equation rather than on the output. This forcing function produces a particular solution to the linear equation which could be computed as a convolution sum. One can then delete the forcing term to the difference

equation, and substitute the particular solution for the output disturbance $v(k)$, to complete getting the equation into the form used in Equation A.11. Or equivalently, we could write $y(t) = y^*(t) + \Delta y(t)$ and $u(t) = u^*(t) + \Delta u(t)$ and convert the equations to use the deviations $\Delta y(t)$, $\Delta u(t)$ as variables. In this case the linearized equation takes the form

$$\Delta \dot{y}(t) + \left[3k_1(y^*(t))^2 + k_2\right] \Delta y(t) = \Delta u(t) \tag{A.14}$$

This form has the same time varying coefficient on the left hand side, but the "disturbance" forcing term is no longer present on the right. Then use the linearization method described in the last section, and combine the results of Equation A.11 and Equation A.3 producing P matrix. The purpose of this chapter is to investigate the property of this resultant P matrix presenting the dynamics of time varying systems. Note that, the coefficients of the system itself could be also time varying, here we consider the situation where the coefficients are dependent at time since linearized system of the original nonlinear system, and the nominal output trajectory is function of time. And it might cause some trouble for some systems for some desired trajectory, like the eigenvalues of $A(k)$ are greater than one, which makes the system P matrix unstbale.

A.3 Investigation of P Matrix of LTV Systems

In this section, the authors investigate the "distortion" to the properties of linear time invariant P matrix listed above in the last section. Systems for which one might want to apply ILC to obtain high precision tracking, can have time varying coefficients. Two cases can apply. One is that the system model is simply time varying. A second situation applies when one wants to use ILC designed for linear systems to apply to nonlinear system. One can linearize the nonlinear model around the desired trajectory that is a function of time, and this produces time varying coefficients. The purpose of the paper is to investigate the second situation. The linear range around the desired trajectory can be large enough to make this approach effective. If not, one may seek to repeatedly linearize about trajectories during the iterations.

The Distortion of Input-Output Singular Vector Pairs

The expectation of the input-output singular vector pairs of the LTI P matrix, as discussed in Chapter 3 and shown in Figure A.1 and Figure A.2. The author is in interested in investigating the systematic pattern of distortion on these singular vector pairs due to the variation on different part of desired $y^*(t)$. Use the linearized time varying system modeled as Equation A.14, and we choose a first set of different desired output trajectories $y_1^*(t)$, $y_2^*(t)$, $y_3^*(t)$ as

$$y_i^*(t) = e^{\left(\frac{(t-m_i)^2}{2\sigma^2}\right)} \tag{A.15}$$

where $m_1 = 0.3$, $m_2 = 1.0$, $m_3 = 1.7$, and $\sigma = 0.1$. Basically, this set of the resired $y^*(t)$ is a Gaussian distribution function without the normalizing factor and with different means

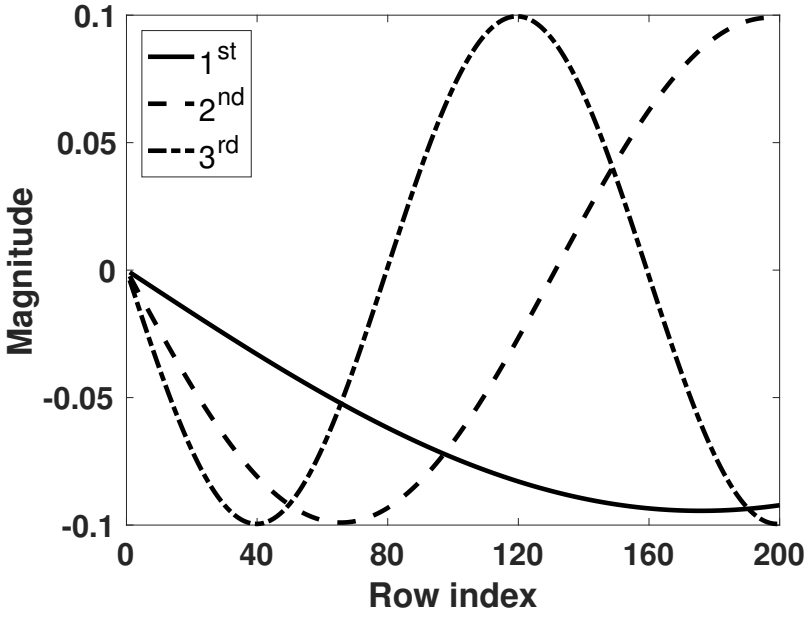

Figure A.1: Magnitude of components of first three output singular vectors of LTI P matrix

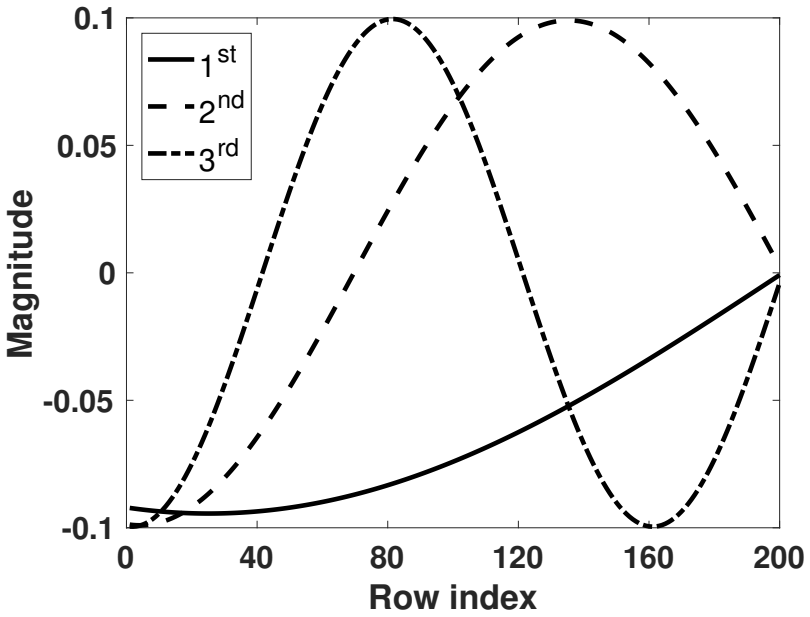

Figure A.2: Magnitude of components of first three input singular vectors of LTI P matrix

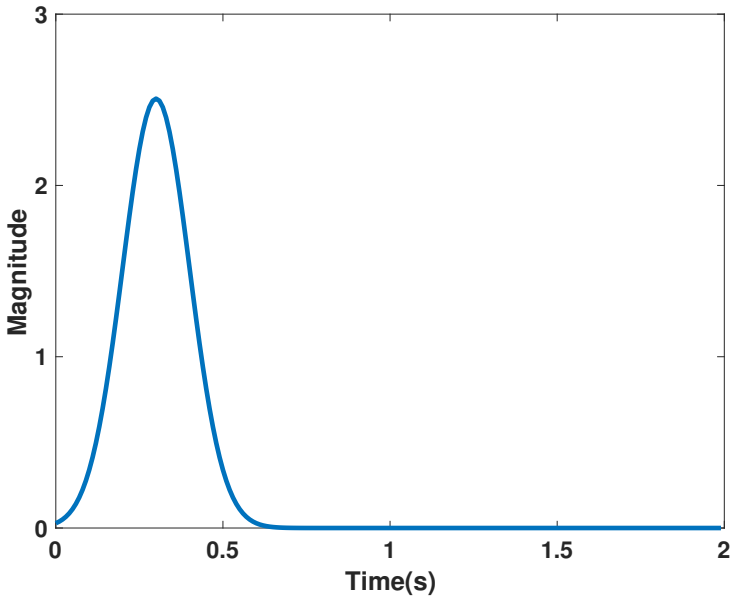

Figure A.3: $y_1^*(t) = e^{\left(\frac{(t-m_1)^2}{2\sigma^2}\right)}$

and the same standard deviation, which are depicted in Figure A.3, Figure A.4, Figure A.5. We choose each desired $y^*(t)$ as two seconds long with 100 sampling rate.

Observation 1: Figure A.6, Figure A.7, Figure A.8, Figure A.9, Figure A.10, Figure A.11, Figure A.12, Figure A.13, Figure A.14 show the first set of the desired output trajectories $y_1^*(t)$, $y_2^*(t)$, $y_3^*(t)$, and their corresponding linearized time varying system coefficients, $A_d(k)$ and $B_d(k)$, since we want to investigate properties of P matrix, the coefficients are in discrete time, although $B_d(k)$ here is not easily visible. And also present the first three singular vectors U and V. It could be seen that for the desired output in the same "shape" with concentration on different parts of the trajectories, i.e. the "bump" is at the start, in the middle and in the end. . For some systems, might lead to the time variation on the system coefficients on their corresponding part of the period. Compared to the first three singular vectors of linear time invariant P matrix, we might also expect that there should be some variation on the input-output singular vector pairs on their corresponding part, respectively. And if the time variation is reasonably small, the rest time histories of the singular vectors remain similar shape to the undistorted ones. At the same, the input-output singular vector pairs no longer remain the reversed time order pattern.

Observation 2: Compared to the desired output $y_1^*(t)$, Figure A.15, Figure A.16, Figure A.17 linearized about the desired trajectory $y_4^*(t) = \sin(7\pi t)e^{\left(\frac{(t-m_1)^2}{2\sigma^2}\right)}$. The authors want to mimic the wavelet basis functions with the mother basis with the Gaussian distribution function coupling with some other sinusoidal functions representing different localized frequencies. We have reasons to believe that after linearizing the desired output

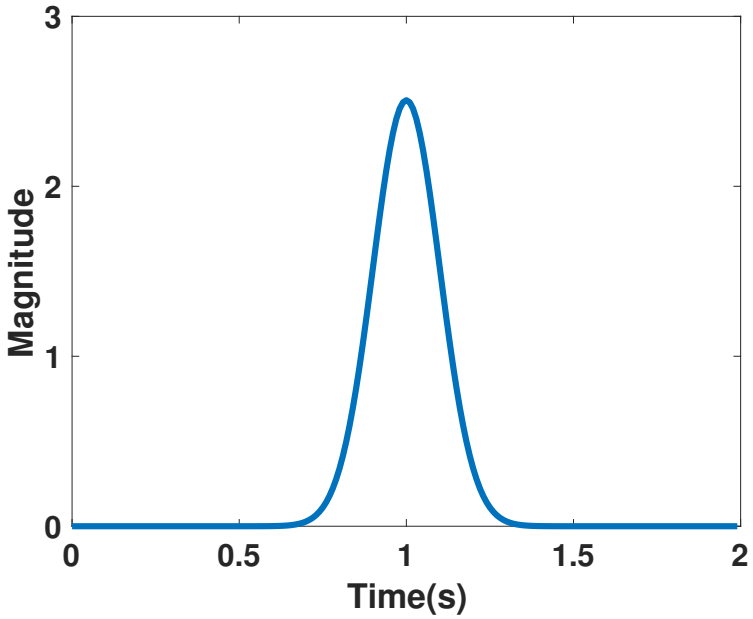

Figure A.4: $y_2^*(t) = e^{\left(\frac{(t-m_2)^2}{2\sigma^2}\right)}$

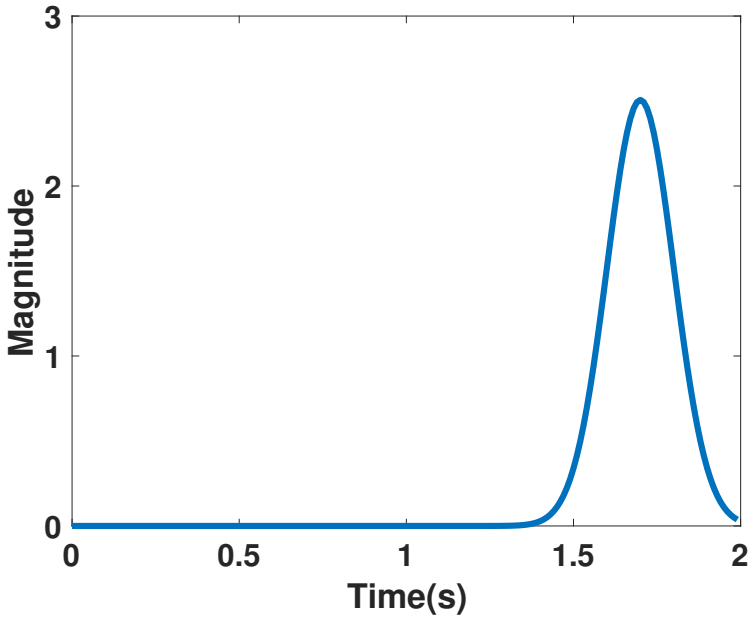

Figure A.5: $y_3^*(t) = e^{\left(\frac{(t-m_3)^2}{2\sigma^2}\right)}$

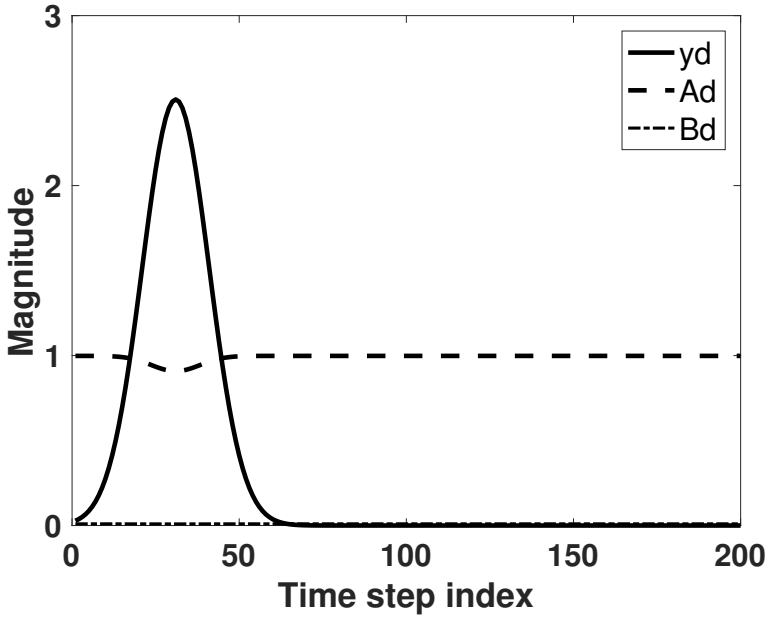

Figure A.6: The desired output $y_1^*(t)$ and time varying coefficients

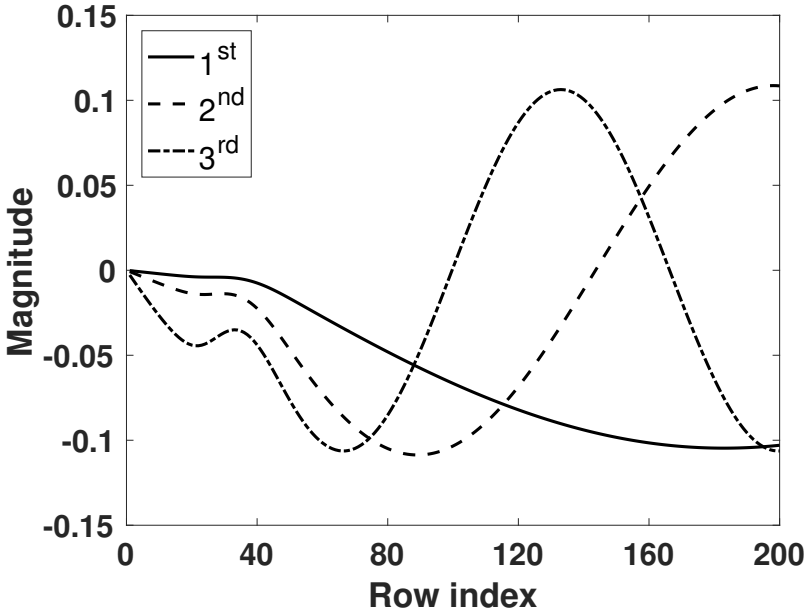

Figure A.7: The resulted first three input singular vectors linearized about $y_1^*(t)$

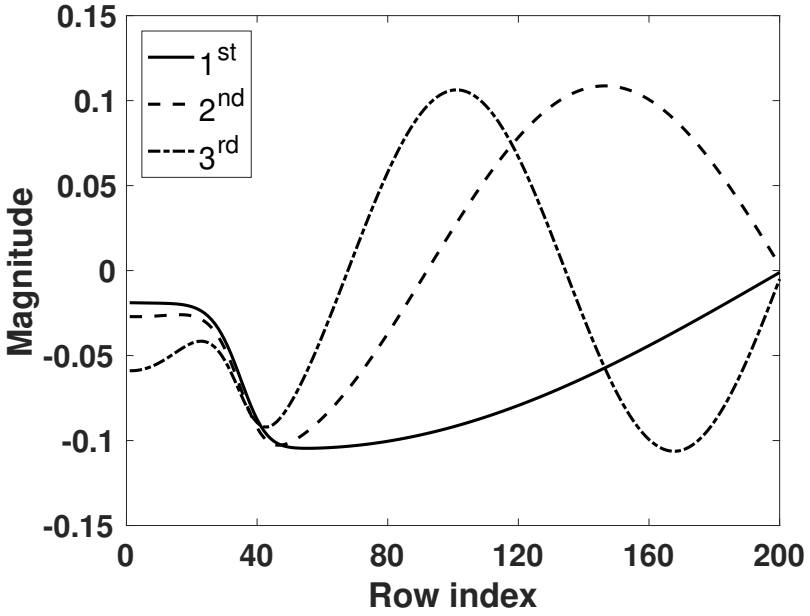

Figure A.8: The resulted first three output singular vectors linearized about $y_1^*(t)$

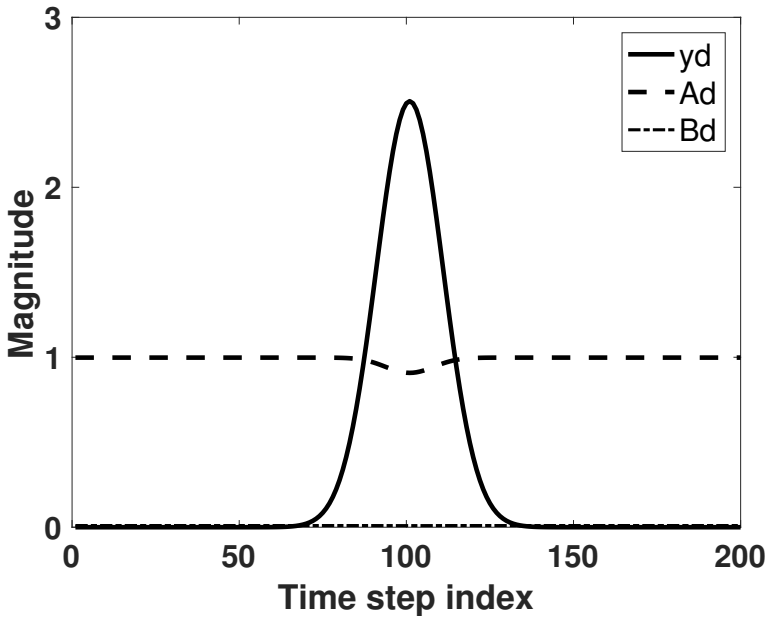

Figure A.9: The desired output $y_2^*(t)$ and time varying coefficients

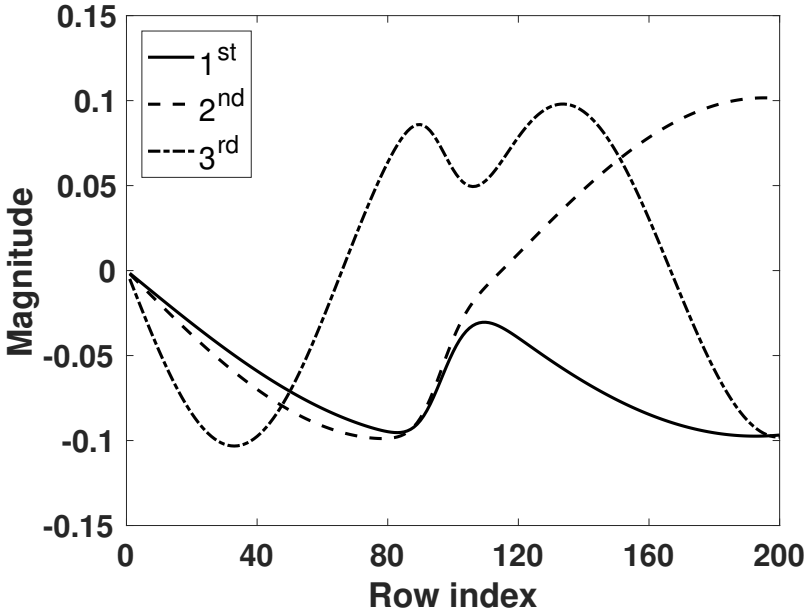

Figure A.10: The resulted first three input singular vectors linearized about $y_2^*(t)$

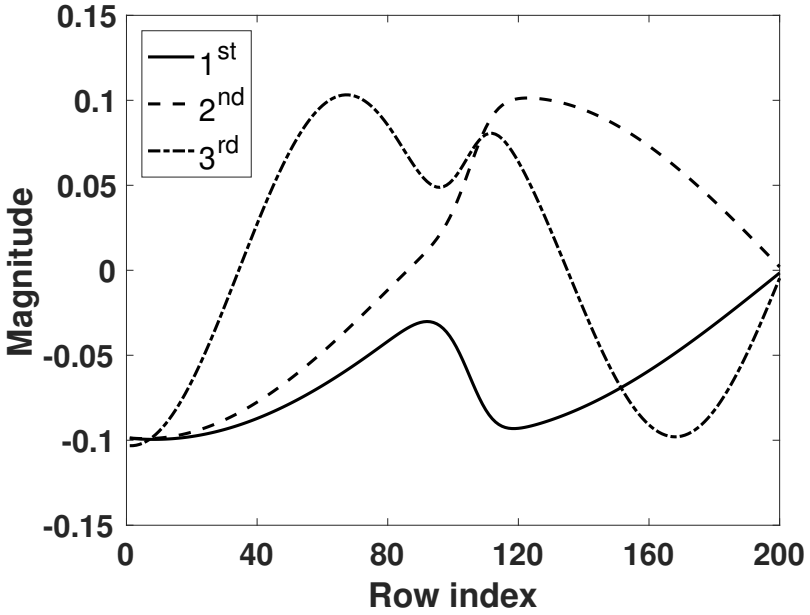

Figure A.11: The resulted first three output singular vectors linearized about $y_2^*(t)$

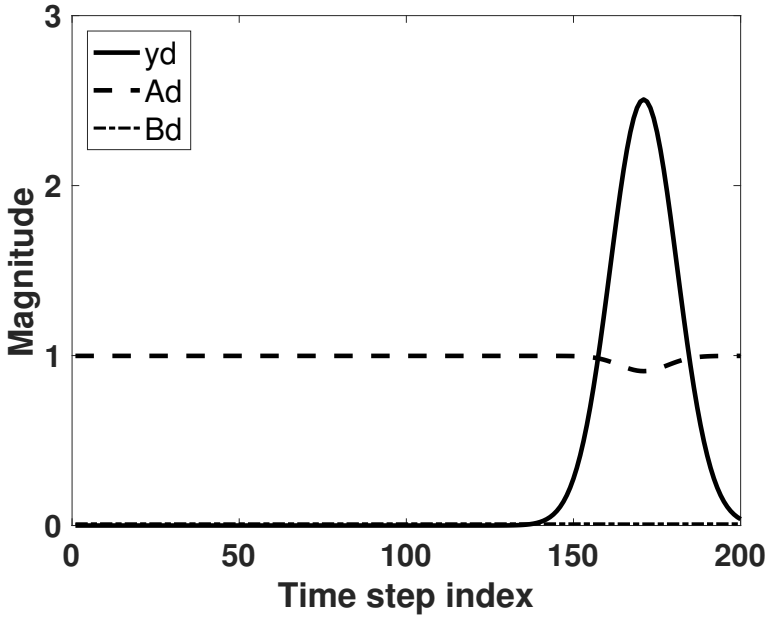

Figure A.12: The desired output $y_3^*(t)$ and time varying coefficients

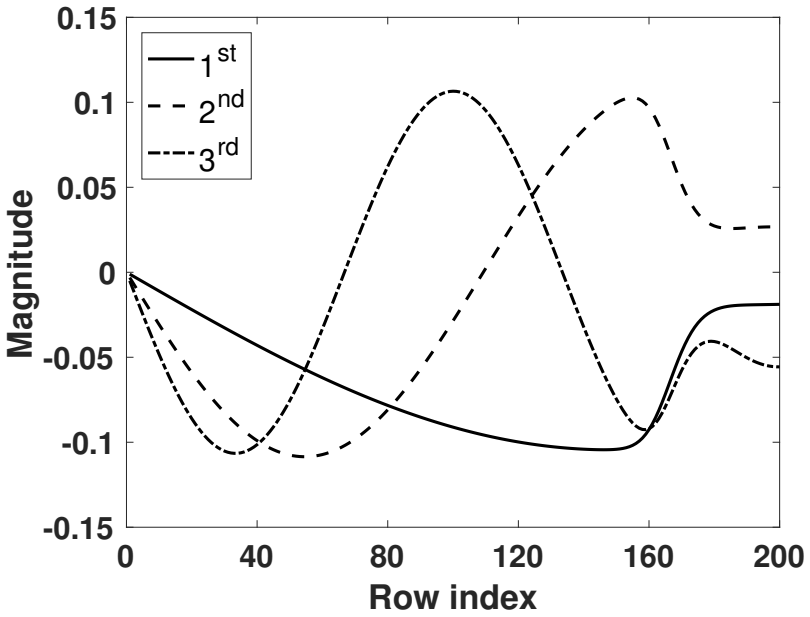

Figure A.13: The resulted first three input singular vectors linearized about $y_3^*(t)$

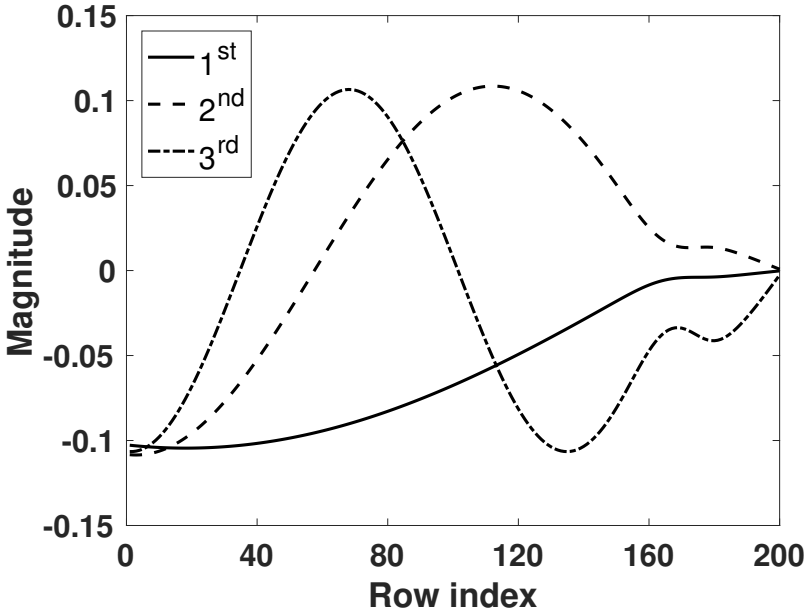

Figure A.14: The resulted first three output singular vectors linearized about $y_3^*(t)$

with different frequencies, leading to the system coefficients have components of different frequencies. And for input-output singular vector pairs, we might expect more "wiggles" in terms of distortion on the corresponding part of the time histories.

The Distortion of the Singular Value Decomposition for Linearly Changing Desired Trajectories

Observation 3: Figure A.18, Figure A.19, Figure A.20 show the systematic results of linearizing about the desired output $Y_5^*(t) = 4t$, a linearly time increasing trajectory. It could be observed that when the system coefficients change monotonically, the input-output singular vectors, instead of well "distributed" along the whole time trajectories, they may flat out to zero sooner. And it might be due to the fact of causality of P matrix, as discussed earlier, for the case of a LTI P matrix, each column is the unit pulse response starting at different time step.

The Distortion of the Singular Value Decomposition for Linearied Systems with Periodic Coefficients

There exists a nice description of the finite time influence on frequency response behavior. A very important property of the steady state frequency response of a linear time invariant system is that a singular frequency input produces a single frequency in the output. Clearly these are not pure sinusoids. In order to associating frequencies with the singular vectors, one could take the discrete Fourier transform of the singular vectors in to determine what frequency they correspond to, and also what singular value is associated with that frequency.

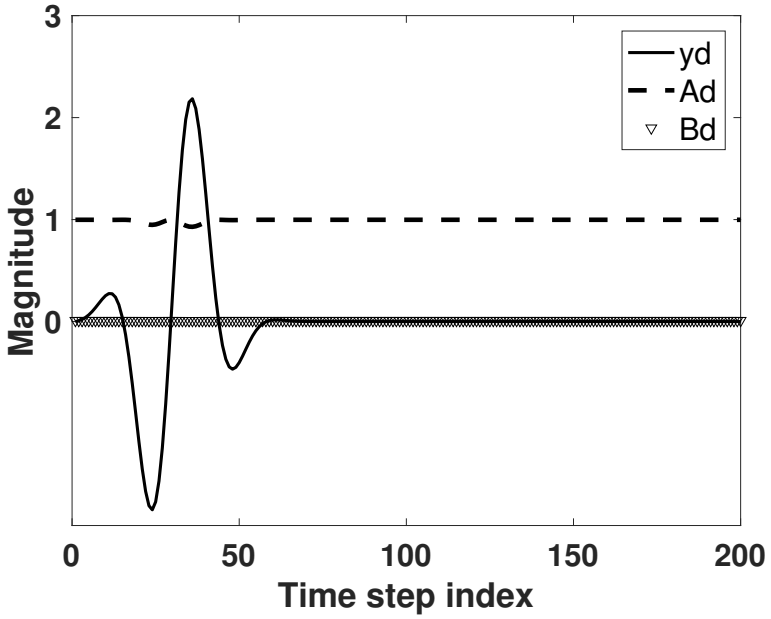

Figure A.15: The desired output $y_4^*(t)$ and time varying coefficients

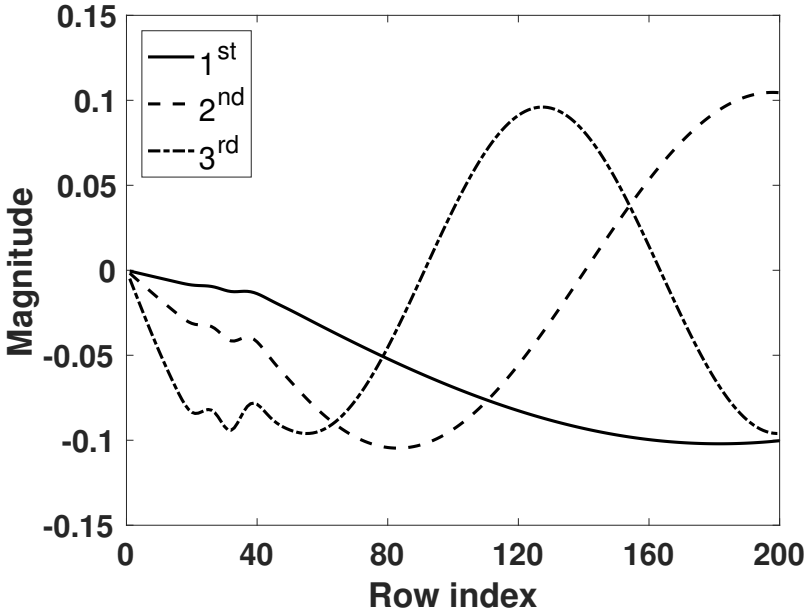

Figure A.16: The resulted first three input singular vectors linearized about $y_4^*(t)$

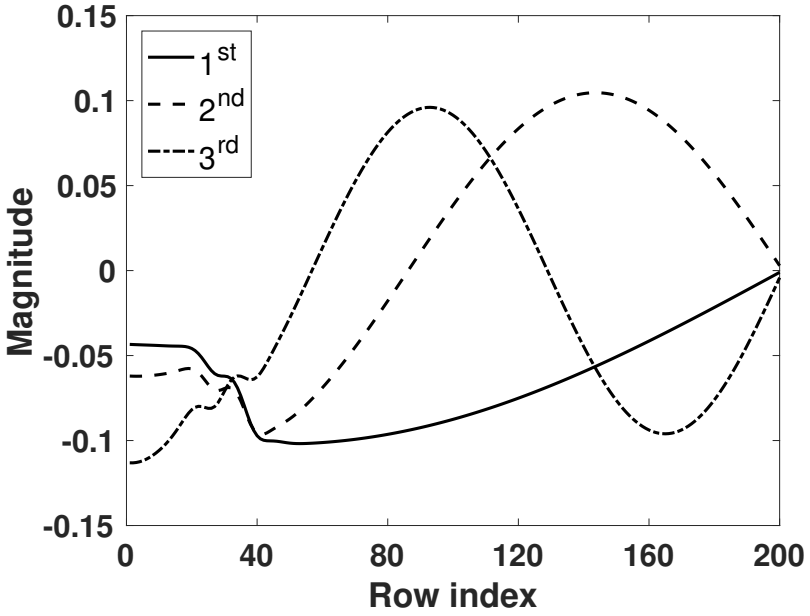

Figure A.17: The resulted first three output singular vectors linearized about $y_4^*(t)$

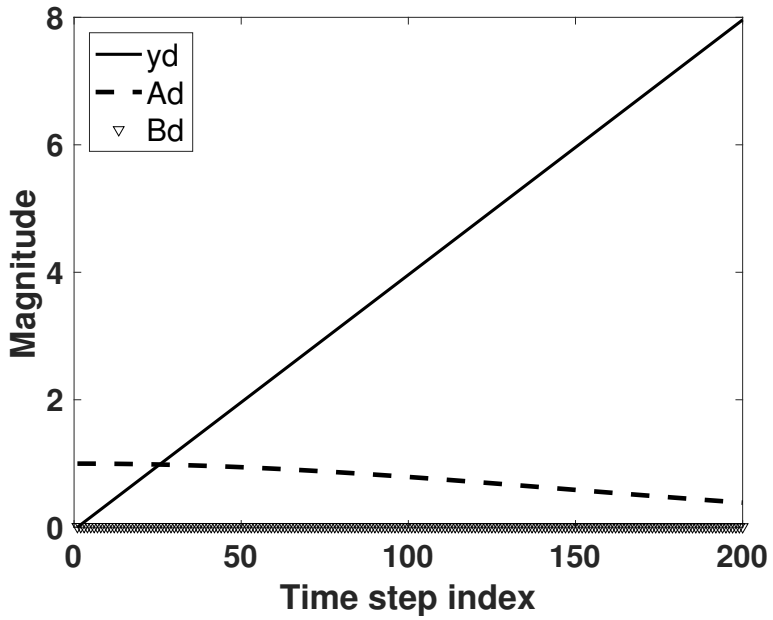

Figure A.18: The desired output $y_5^*(t)$ and time varying coefficients

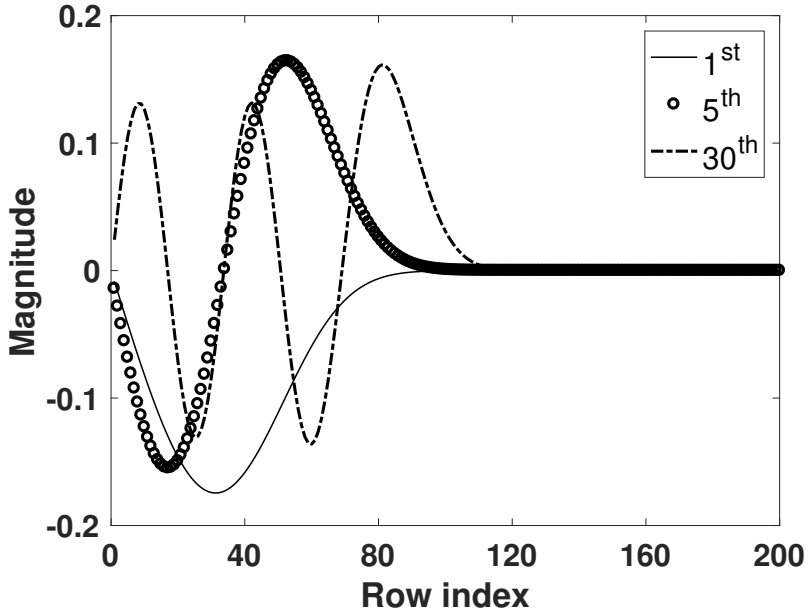

Figure A.19: The resulted first three input singular vectors linearized about $y_5^*(t)$

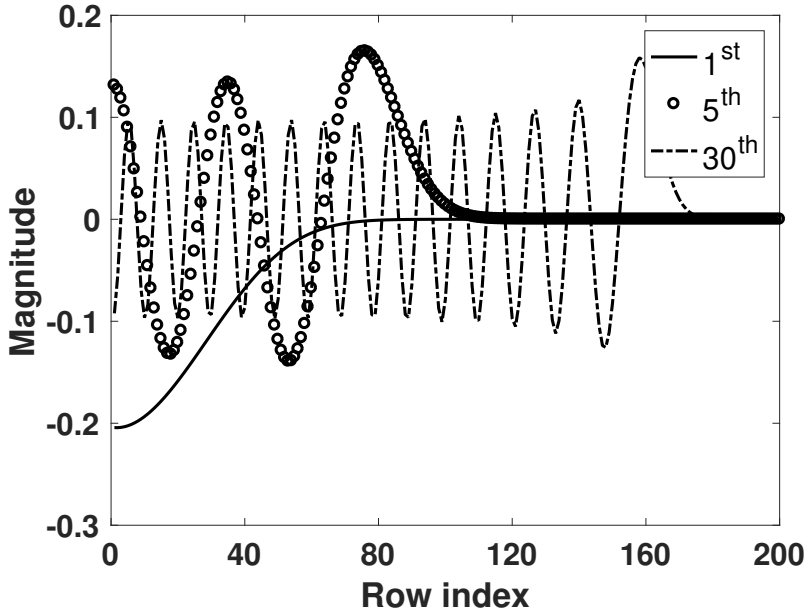

Figure A.20: The resulted first three output singular vectors linearized about $y_5^*(t)$

And take a closer look, if we take the DFT of the columns for the columns and use the largest magnitude entry as the frequency, the mapping of columns to frequency number follows: 1, 2, 2, 3, 3, 4, 4, …. There are important situations in which periodic coefficient linear systems arise discussed in Reference [76]. A large class of such systems result when one wants to apply iterative learning control in nonlinear system. Linearizing about the desired trajectory results in linear equations with periodic coefficients. One developed the inverse of the steady state frequency response of the system and used it as a compensator to update the command for a feedback control system. An important new aspect is that unlike constant coefficient system where a single frequency input produces the same single frequency in the output, the periodic coefficient problem can have multiple harmonic and subharmonics in the output. For simplicity, if there is one frequency in the periodic coefficient, we could see the effect is that it produces the sums and differences of every frequency in the input or output respectively. If there are more than one frequency in the coefficient, we would have more sums and differences. The authors try to relate this phenomenon to the distortion of the singular vectors upon the time variation on the constant coefficients. For the purpose of illustration, consider a first order nonlinear system

$$y'(t) + ay^2(t) = bu(t) \tag{A.16}$$

and then linearize about the desired trajectory to get linearized time varying system

$$\Delta y'(t) + 2ay^*\Delta y(t) = b\Delta u(t) \tag{A.17}$$

To see the updated singular vectors, we first choose $y_8^*(t) = 20 * [-1 + 0.5 * sin(4\pi t)]$ with one single relatively low frequency components, and then $y_9^*(t) = 20 * [-1.1 + 0.5 * sin(16\pi t)]$ containing higher frequency component. And then take DFT of the columns of singular vector matrices.

Observation 4: The left part of Figure A.21 shows the desired trajectory $y_8^*(t)$ with single low frequency component and periodic coefficients of the linearized system. The solid line in the Figure A.22 is DFT of the 10^{th} singular vector of U of the original system with constant coefficients. From previous experience, it is reasonable since we use the largest magnitude entry as the frequency, and the number of column 10 maps to 6 Hz. It is interesting to see that, the 10th singular vector of the time varying system has the largest magnitude at 6 Hz as well, however, it also contains relatively big magnitude at both 4 Hz and 8 Hz, as the sum and difference of the original frequency 6 Hz and the 2 Hz frequency component contained in the periodic coefficients introduced during the linearization about the desired output which contains one single frequency. Figure A.23 and Figure A.24 show the similar phenomenon except that we choose $y_9^*(t)$ to linearize about which contains one single higher frequency 8 Hz. In DFT of the 30^{th} singular vector of U of the linearize system with periodic coefficients, we observe that besides the largest magnitude entry representing the dominant centered frequency 16 Hz, also two spikes at 8 Hz and 24 Hz, again the sum and difference of original 16 Hz and 8 Hz component contained in the periodic coefficients. Note that similar behavior could be seen in the singular vectors of V. Now what if we choose $y_1 0^*(t) = y_8^*(t) + y_9^*(t)$, the combination of previous two trajectories to linearize about,

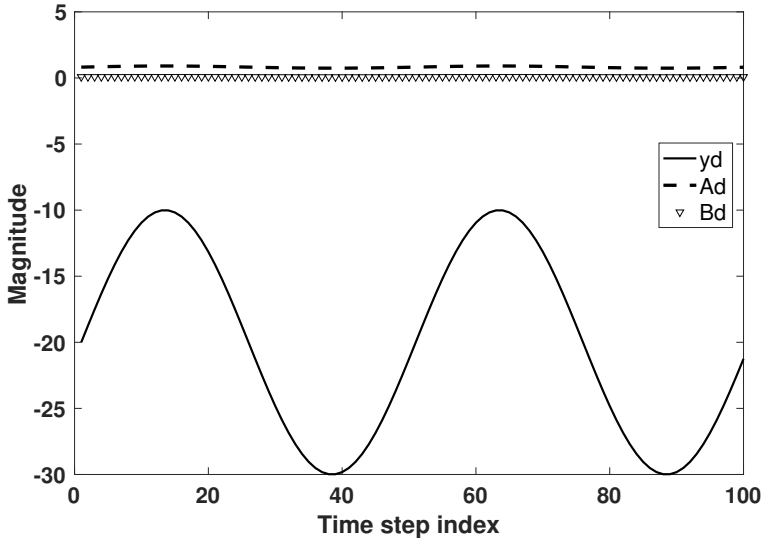

Figure A.21: The desired output $y_8^*(t)$ and time varying coefficients

since now it contains more than one single frequency. Again, shown in Figure A.25 and Figure A.26, the largest magnitude entry is at 16 Hz, and the secondary largest magnitude entries are at 14 Hz and 18 Hz, corresponding to the components of coefficients regarding to $y_8^*(t)$ whose magnitude is greater than $y_9^*(t)$, and then two relatively large entries at 8 Hz and 24 Hz, corresponding to $y_9 * (t)$. It is interesting to note there are also some other spike in between the sums and differences.

Investigation of the Stable Inverse Methods for the Modified SVD

The basic laws of contraction mapping, partial isometry, and quadratic cost learning updates can in theory be used directly with time varying systems, but no one has investigated on the issue of what happens to the troublesome singular values associated with zeros of linear transfer functions outside the unit circle for constant coefficient systems are transformed by time variation. The purpose of this section seeks to know if the anomalous singular values defined above still exist, if so, do the methods summarized in Chapter 4 still address the issus of instability of the inverse. With the method discussed in the last section, we consider the 3^{rd} order nonlinear system $y'''(t) + (2\xi_1\omega_1 + a)\, y''(t) + (\omega_1^2 + 2\xi_1\omega_1 a)\, y'(t) + y^2(t) + a\omega_1^2 y(t) = a\omega_1^2 u(t)$ with the nonlinear term $y^2(t)$, and linearize about some desired output trajectory, giving the linearized time varying system model

$$\Delta y'''(t)+(2\xi_1\omega_1 + a)\,\Delta y''(t)+\left(\omega_1^2 + 2\xi_1\omega_1 a\right)\Delta y'(t)+\left(2y^*(t) + a\omega_1^2\right)\Delta y(t) = a\omega_1^2\Delta u(t) \tag{A.18}$$

where $a = 8.8$, $\xi_1 = 0.5$, $\omega_1 = 37$. Numerical experience shows that when one linearizes about some desired output trajectory, could result in unstable linear time varying system, i.e. the eigenvalues of $A_d(k)$ could be outside the unit circle. We choose $y_6^*(t) = 0.1y_1^*(t)$ and $y_7^*(t) = 0.1y_5^*(t)$ to linearize about. Figure A.27 and Figure A.28 show the 1^{st}, 20^{th} and 60^{th}

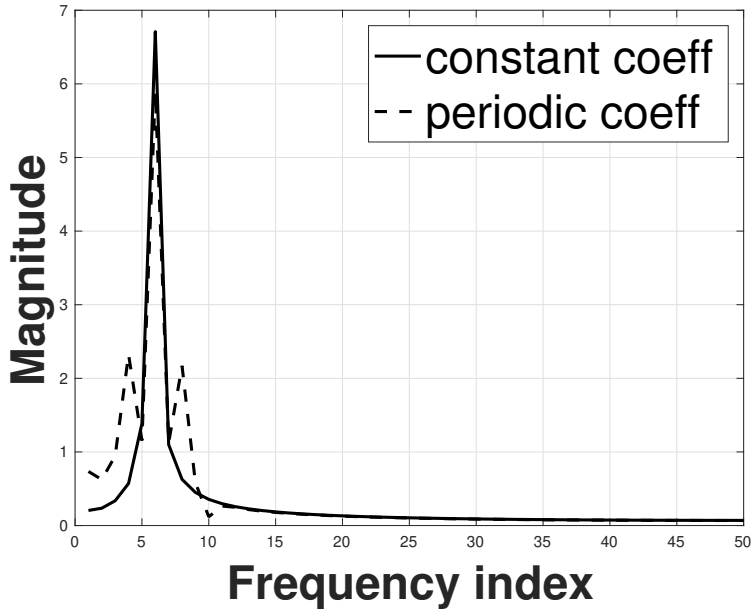

Figure A.22: DFT of th 10^{th} input singular vector of the constant coefficients system compared to the linearized system with periodic coefficients

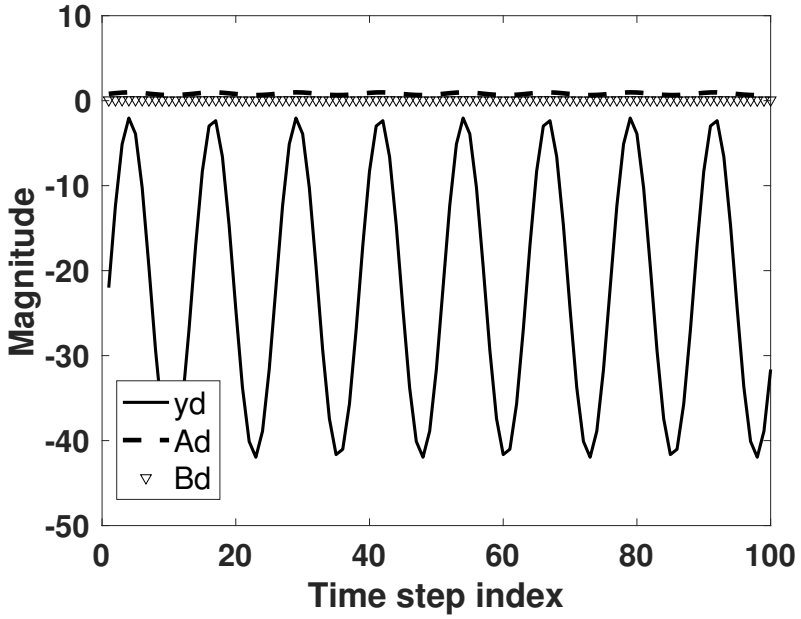

Figure A.23: The desired output $y_9^*(t)$ and time varying coefficients

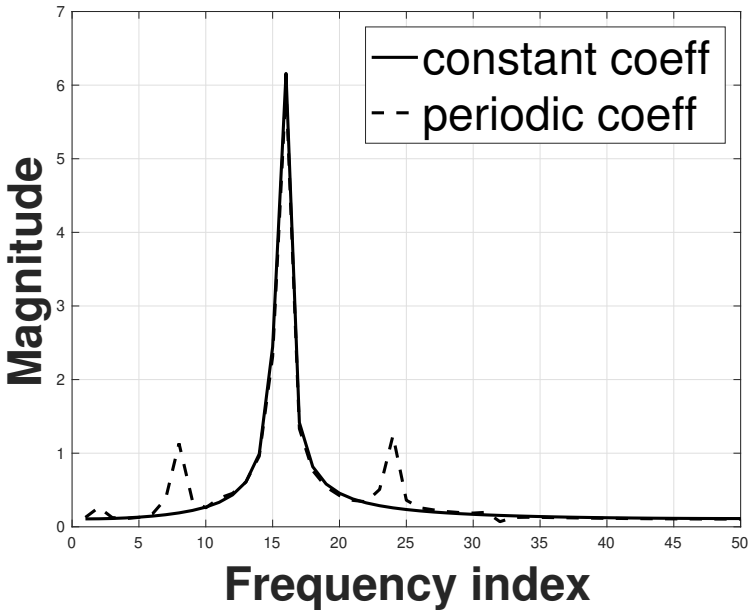

Figure A.24: DFT of th 30^{th} input singular vector of the constant coefficients system compared to the linearized system with periodic coefficients

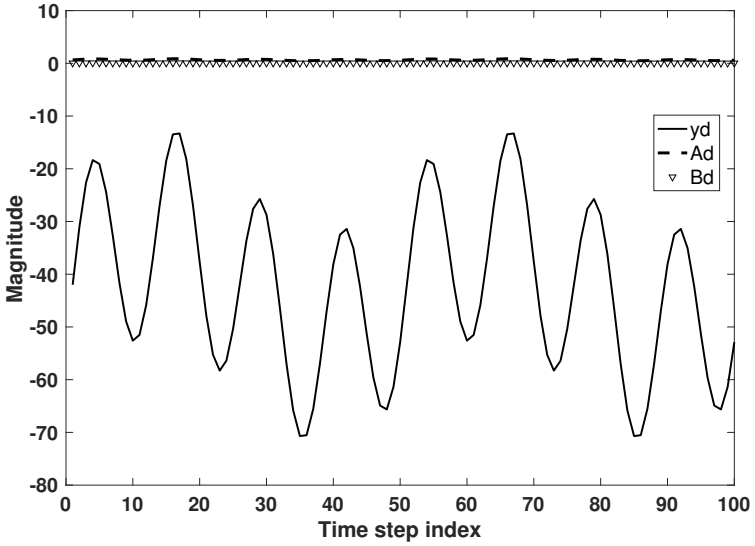

Figure A.25: The desired output $y_1 0^*(t)$ and time varying coefficients

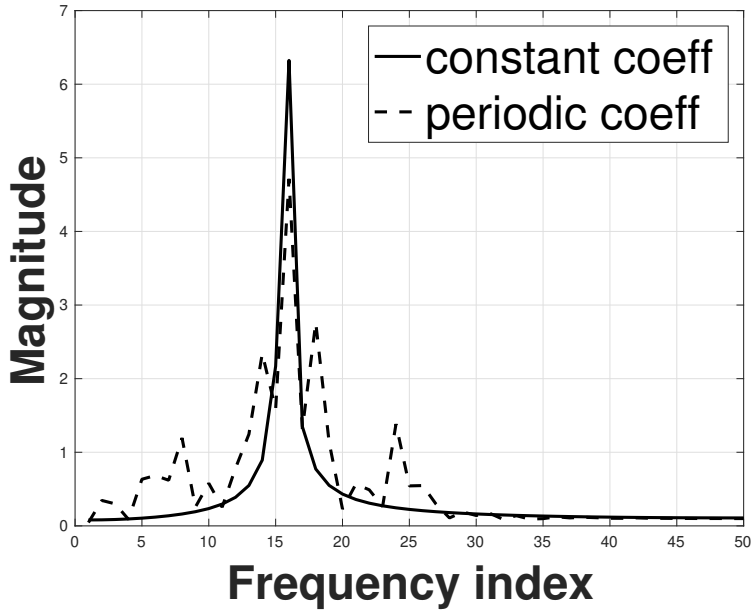

Figure A.26: DFT of th 30^{th} input singular vector of the constant coefficients system compared to the linearized system with periodic coefficients

column of P matrix constructed by the corresponding time dependent matrix coefficients $A_d(k)$, $B_d(k)$ and $C_d(k)$ in discrete time space. From the plot it is easily seen that P matrix is no longer in the Toeplitz matrix.

Observation 5: Figure A.29 and Figure A.30 show that, similar to the linear time invariant P matrix, when the system coefficients are affected by reasonable amount of time variation, we might still encounter the troublesome of the existence of the troublesome anomalous singular value preventing producing a stable inverse of the system and the pair of singular vectors associated with the anomalous singular value, with one exponentially growing and the other one exponentially decaying. It also shows that with the stable inverse methods developed by the authors and workers, by deleting the first row of time dependent P matrix (bottom left), or allow the use of two zero order hold values between each time step for which one asks for zero tracking error, i.e. the number of rows deleted to the number of anomalous singular value, they somehow still work eliminating the anomalous singular value. The updated singular values are marked with asterisks in Figure A.31 and Figure A.32.

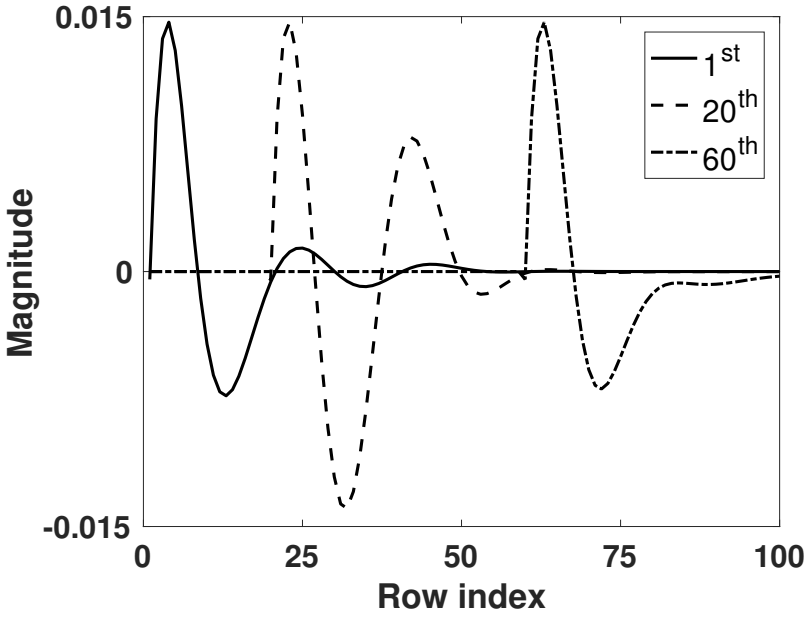

Figure A.27: 1^{st}, 20^{th} and 60^{th} column of time varying P matrix linearized about $y_6^*(t)$

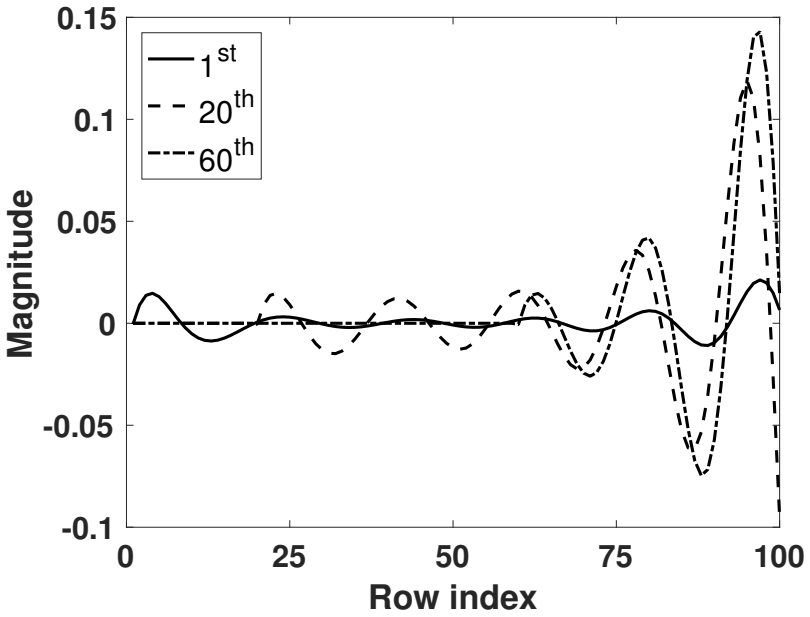

Figure A.28: 1^{st}, 20^{th} and 60^{th} column of time varying P matrix linearized about $y_7^*(t)$

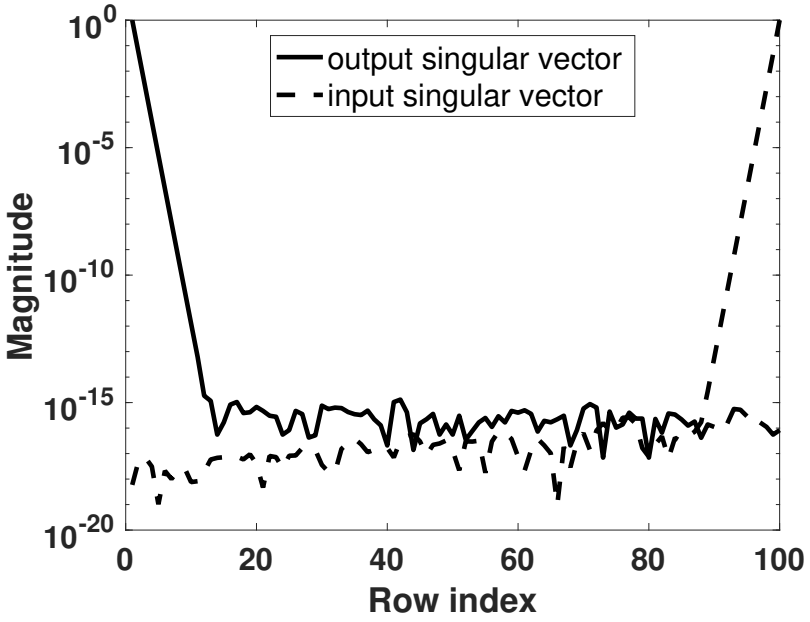

Figure A.29: Input-out singular vector pair associated with $\sigma_{\min}$ of P_6

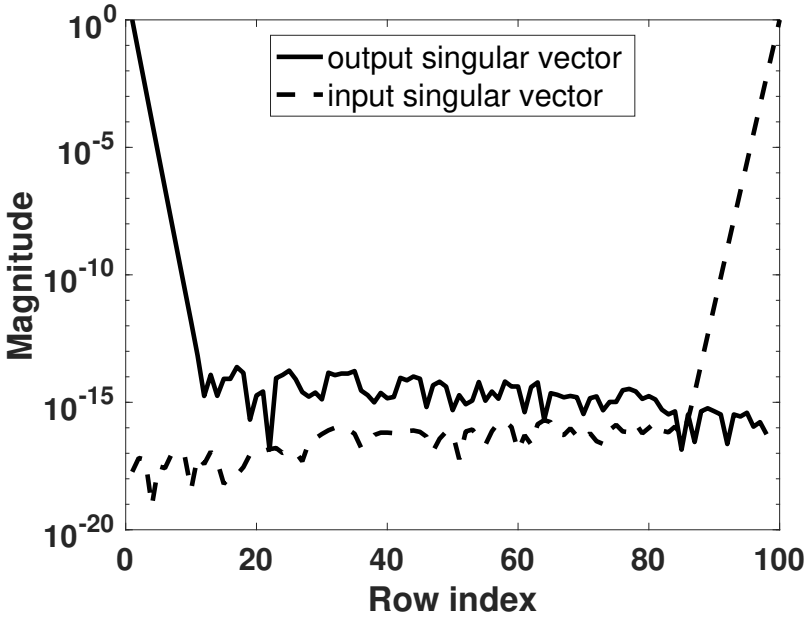

Figure A.30: Input-out singular vector pair associated with $\sigma_{\min}$ of P_7

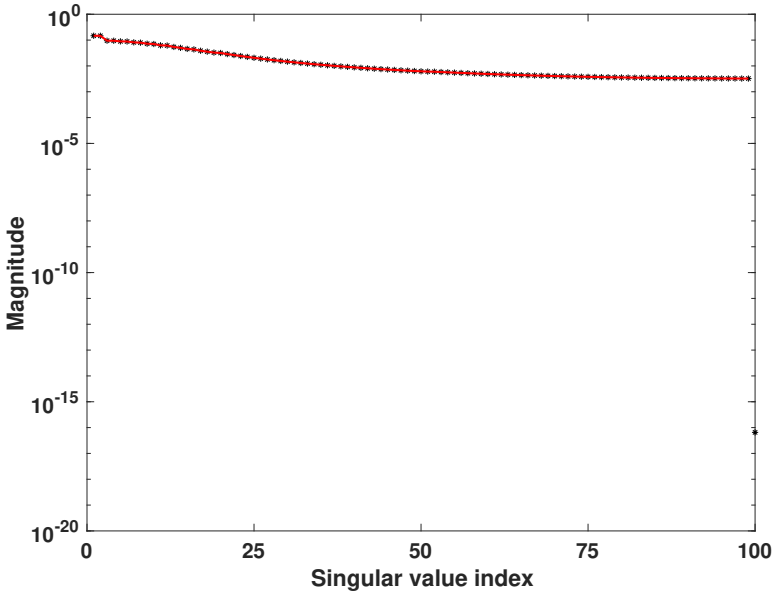

Figure A.31: The updated singular values of P_{6d}

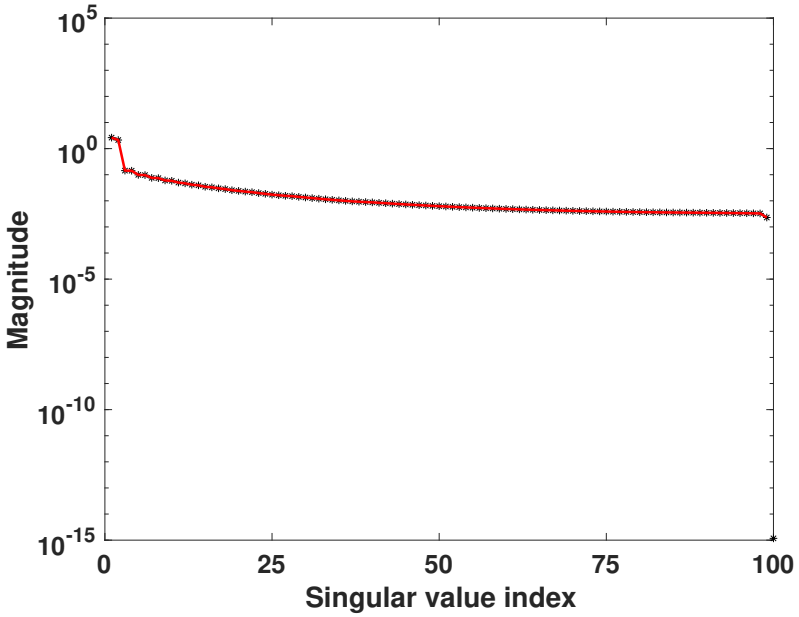

Figure A.32: The updated singular values of P_{7d}