CONTROL OF A DC MOTOR: COMPARING CLASSICAL AND ADAPTIVE TECHNIQUES

By

Jay L. Adams

Submitted in Partial Fulfillment of the Requirements

For the Degree of

Master of Science of Engineering

in the

Electrical Engineering

Program.

YOUNGSTOWN STATE UNIVERSITY

August 2004

CONTROL OF A DC MOTOR: COMPARING CLASSICAL AND ADAPTIVE TECHNIQUES

Jay L. Adams

Signature:

Jay Adams 8/12/2004

Jay Adams, Student Date

Approvals:

[illegible] 8/12/04

Dr Faramarz Mossayebi, Thesis Advisor Date

[illegible] 8/12/2004

Dr. Robert H. Foulkes, Committee Member Date

Salvatore R Pansino 8/12/2004

Dr. Salvatore Pansino, Committee Member Date

[illegible] 8/23/04

Dr. Peter J. Kavinsky, Dean of Graduate Studies Date

ABSTRACT

The intent of this project is to compare and contrast the advantages and disadvantages of using adaptive techniques as opposed to classical ones in the control of a DC motor. To that end Root Locus and the adaptive techniques of self-tuning regulation, one-step ahead adaptive control, and model-reference adaptive control are presented. The controllers are designed based on a model that is commonly used for a DC motor. To investigate the effect of a non-ideal model on the controlled system, several variations are considered. The effect of model mismatch is investigated by making the order of the plant and the type of the plant higher than the assumed model for controller design. Furthermore, the effect of load on the motor as well as the presence of additive noise is considered. The Root-Locus designed controller does not meet the criteria in most non-ideal situations. The self-tuning regulator meets the design criteria in all the non-ideal cases, but has a high control effort before the parameter estimates converge, after which the control effort is more reasonable. The weighted model-reference adaptive system best meets the design criteria with the least control effort in all the non-ideal cases if the model used to design the controller has been over-parameterized. This thesis concludes with a summary of the project results and ideas for future research topics.

ACKNOWLEDGEMENTS

It pleases me to express gratitude to several people who have been instrumental in my success with this work and the master's degree. First and foremost, I must thank the God who created me with the capacities and abilities for reasoning that I have, without whom all my efforts would have been and be in vain. Dr. Faramarz Mossayebi has contributed greatly to this work and my future education with his expertise, experience, and encouragement. Our weekly meetings have kept me on track and given me motivation to do what needed to be done. Further, I must thank Dr. Robert H. Foulkes, Jr and Dr. Salvatore Pansino for serving on my thesis committee. I am also compelled to thank Dr. Thomas J. Cavicchi who served as inspiration and introduced me to systems and controls. My parents, J. Lawrence and Beth Ann, have ever been ardent supporters of my academic endeavors, believing in me, and spurring me on toward greatness, even when future prospects looked bleak; no greater gift could I have been given. The support and patience exhibited by Lynn Irene Schwartz, listening to my defense, providing technical help, and encouraging me through out my work at Youngstown State University has been exceedingly helpful as well. Finally, I thank everyone who has been thinking about and praying for me in this and future academic endeavors.

TABLE OF CONTENTS

LIST OF SYMBOLS

SYMBOL	DEFINITION
A	Denominator of the transfer function or pulse-transfer function
A_{CL}	Closed-loop denominator
A_m	Desired closed-loop denominator
A_p	Denominator of the pulse-transfer function
$A(s)$	Denominator of the transfer function
B	Damping constant of the motor
B	Numerator of the transfer function or pulse-transfer function
B_m	Desired closed-loop numerator
B_p	Numerator of the pulse-transfer function
$B(s)$	Numerator of the transfer function
$D(z)$	Discrete-time compensator
d	Prediction horizon
d_0	Pole excess
$\frac{d}{dt}(\ )$	Derivative with respect to time
$G_m(s)$	Desired transfer function
$G_p(s)$	Transfer function of the plant
$\mathbf{H}$	Linear function
$H_{m_{pq}}(z)$	Desired pulse-transfer function

$H_{pq}(z)$	Pulse-transfer function of the plant
i	Current through the motor armature
J	Moment of inertia of the rotor
K	Motor constant
K	System gain
K_e	Electrical constant of the motor
K_t	Torque constant of the motor
L	Inductance
m	Order of numerator of pulse-transfer function
n	Order of denominator of pulse-transfer function
P	Confidence matrix
q	Forward shift operator
q^{-1}	Reverse shift operator
R	Self-tuning regulator control law denominator
R	Resistance
S	Self-tuning regulator feedback numerator
s	Laplace variable
T	Self-tuning regulator feedforward numerator
T_s	Sampling time
T_{set}	Settling time
$u(k)$	Input into the plant
$u(t)$	Input into the plant

$u_c(k)$	Reference signal
$u_c(t)$	Reference signal
$v(k)$	Noise into the plant
$v(t)$	Noise into the plant
$v(t)$	Source voltage
$\mathbf{x}$	Parameters to be estimated
$y(k)$	Output of the plant
$y(t)$	Desired output of the plant
$y_m(k)$	Desired output of the plant
$y_m(t)$	Output of the plant
$\mathbf{Z}$	Noisy observation
$ZT\{\ \}$	Z-transform
$\tilde{\varepsilon}$	Estimation error
ζ	Damping ratio
λ	Weighting factor
$\phi(k)$	Regression vector
τ	Torque of the rotor
θ	Position of the motor shaft
$\dot{\theta}$	Speed of the motor shaft
$\theta(k)$	Parameter vector
ω_n	Natural frequency
$\%OS$	Percent overshoot

LIST OF FIGURES

INTRODUCTION

Control theory is concerned with obtaining a desired response from a system, or plant. Classical control theory, in the form of root locus, pole placement, etc, can be used when the designer has a priori knowledge of the plant. Control systems allow the management of large and small equipment with a degree of precision that would otherwise be impossible. Further, the use of control systems incorporating feedback allows compensation for disturbances, such as mechanical or electrical noise [3],[8],[11],[14].

In some cases, however, a portion or the entirety of the plant is unknown; in other cases the parameters of the plant change, or disturbances are too great for the classical controller to reject. In these instances different techniques must be applied; one such class of techniques is known collectively as adaptive control. Adaptive control was first developed in the early twentieth century to automatically steer a ship. These developments contributed to proportional-integral-derivative control. Other adaptive techniques are the self-tuning regulator and the model-reference adaptive controller [1],[7],[11].

Adaptive control schemes incorporate some sort of on-line parameter estimation scheme that takes into account any changes in the external or internal factors that contribute to change in the plant. The most widespread and common parameter estimation method is that of least squares regression. Least squares regression can be used in nonlinear applications where the system is linear in the parameters, which can sometimes be selected or manipulated in such a manner as to cause this [1],[3].

The first section of this work, encompassing Chapters I and II, is concerned with the basic concepts behind classical and adaptive control. Chapter I follows the development of a transfer function for a DC motor, which is then analyzed for response and developed into a pulse-transfer function. The establishment of a desired system response is also Chapter I. The least squares estimation technique is derived in Chapter II; both a non-recursive estimation and an online estimation algorithm are given.

Chapters III, IV, and V compose the second section of the thesis, which contains the design of the three different types of controllers. The root locus technique is utilized in Chapter III to design a classical controller that conformed to the design criteria set forth in Chapter I. The self-tuning regulator is developed and designed for the dc motor in Chapter IV. This chapter contains the design of two self-tuning regulators, one with process zero cancellation and one without any cancellation. In the final chapter in this section, Chapter V, development of the one-step ahead and model-reference adaptive controllers are presented. The one-step ahead algorithm is divided into two different types, a pure one-step ahead controller and one that takes into account the desire to keep the control effort low.

The final section of this work includes Chapters VI and VII. Chapter VI compares the performance of the different controllers when the system is placed under various changes. The final chapter in the thesis presents a discussion of the results and the conclusions derived from the work contained in this thesis, as well as recommendations for future study.

CHAPTER I

MODELLING THE DC MOTOR

In this chapter, the required background will be established. First, a mathematic model of a dc motor will be developed, which will be used to determine the relation between the voltage input to the motor and the speed at which the motor rotates. Secondly, because typical control systems are implemented with computers, a discrete-time representation for the continuous-time plant must be found. This is called the pulse-transfer function. The open-loop response will be addressed in the chapter, as will the design criteria. The final topic addressed in this chapter is the effect that varying the load of the dc motor has on the transfer function.

1.1 Determining the Transfer Function

In order to enact any sort of control on the DC motor under investigation, a model must be formulated. Figure 1.1 shows a typical model of a DC motor [4],[14]. The torque produced by the motor is directly proportional to the current through the armature. In other words,

$$\tau = K_t i \tag{1.1}$$

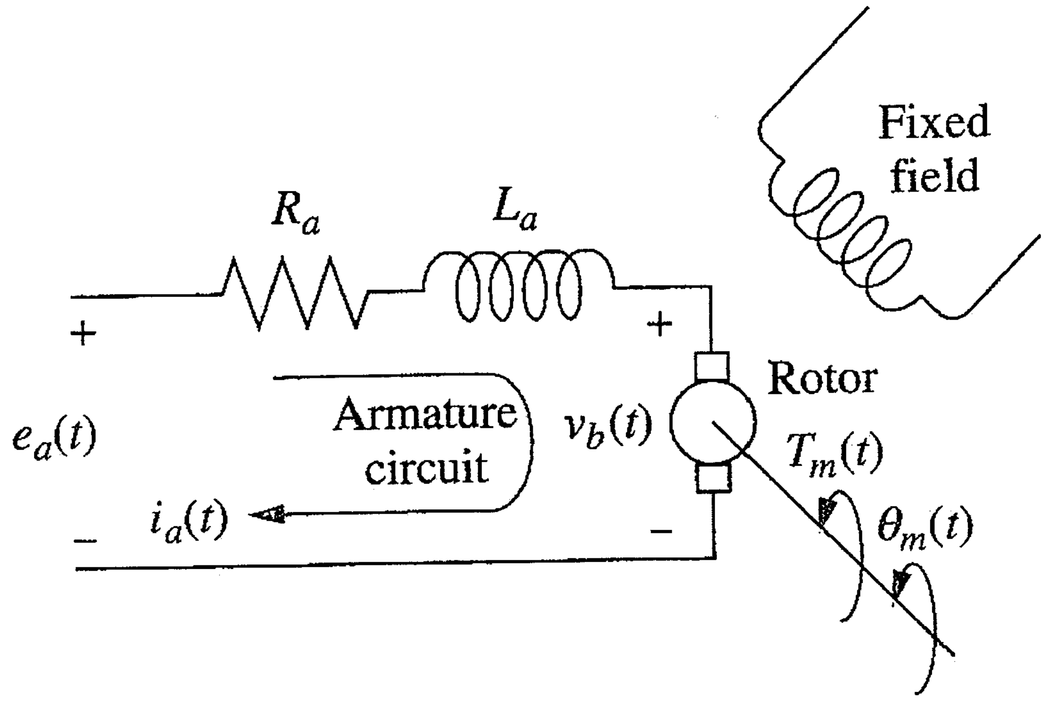

Figure 1.1: Model of the DC Motor.

holds, where K_t is the torque constant. A back electromotive force, or emf, is generated by the motor, which is directly proportional to the speed of the rotor, shown by

$$emf = K_e\dot{\theta}\,, \tag{1.2}$$

where K_e is the electric constant. In the system of units is consistant, it can be shown that

$$K = K_t = K_e \tag{1.3}$$

is true [14]. Using Kirrchoff's Voltage Law around the electronic circuit gives

$$v(t) - K\cdot\dot{\theta}(t) = i(t)\cdot R + \frac{di(t)}{dt}\cdot L \tag{1.4}$$

where $v(t)$ is the source, or input, voltage. Taking the Laplace Transform of Equation 1.4 results in

$$V(s) - K\cdot s\Theta(s) = I(s)\cdot R + sI(s)\cdot L = (sL+R)\cdot I(s)\,. \tag{1.5}$$

The motion of the rotor is opposed by the inertia of the rotor and the friction of rotor. This is described by Newton's Law and can be written as

$$K\cdot i(t) - J\cdot\ddot{\theta}(t) - b\cdot\dot{\theta}(t) = 0\,, \tag{1.6}$$

where $\theta(t)$ is the angular position of the motor. Taking the Laplace transform of Equation 1.6 yields

$$K\cdot I(s) = J\cdot s^2\Theta(s) + B\cdot s\Theta(s)\,. \tag{1.7}$$

Rearranging Equation 1.5 to solve for $I(s)$ yields,

$$I(s) = \frac{V(s) - K\cdot s\Theta(s)}{sL+R}\,. \tag{1.8}$$

In order to find a transfer function for the dc motor, with source voltage, $v(t)$, as the input and angular velocity, $\frac{d\theta}{dt}$ as the output, the expression for $I(s)$ in Equation 1.8 is substituted into Equation 1.7, resulting in

$$J \cdot s^2\Theta(s) + B \cdot s\Theta(s) = K \cdot \frac{V(s) - K \cdot s\Theta(s)}{sL + R}, \tag{1.9}$$

which can be rearranged to collect the like terms, giving

$$\left((sL + R)(sJ + B) + K^2\right) \cdot s\Theta(s) = K \cdot V(s). \tag{1.10}$$

This gives the transfer function,

$$G_p(s) = \frac{s\Theta(s)}{V(s)} = \frac{K}{(sL + R)(sJ + B) + K^2}. \tag{1.11}$$

With some algebraic manipulation, $G_p(s)$ becomes

$$G_p(s) = \frac{\frac{K}{JL}}{s^2 + \left(\frac{B}{J} + \frac{R}{L}\right)s + \frac{RB + K^2}{JL}}. \tag{1.12}$$

1.3 Numerical Transfer Function

Based on experiments conducted at Carnegie Mellon University [4], the following values were selected for the parameters of the dc motor:

$$J = 0.01\frac{kg \cdot m^2}{s^2}, \tag{1.13}$$

$$b = 0.1N \cdot m \cdot s, \tag{1.14}$$

$$K = K_e = K_t = 0.01\frac{N \cdot n}{A}, \tag{1.15}$$

$$R = 1\Omega, \tag{1.16}$$

and

$$L = 0.5H. \tag{1.17}$$

Substituting these values into Equation 1.12 results in the transfer function,

$$G_p(s) = \frac{2}{s^2 + 12s + 20.02}, \tag{1.18}$$

with poles at -2.0025 and -9.9975. Figure 1.2 shows the step response of this system as obtained via MATLAB.

1.3 Determining the Pulse-Transfer Function

For computer implementation of control systems, a discrete-time transfer function, $H_{pq}(z)$, must be developed. A common technique used to change signals from digital to analog is by means of the zero-order hold [15],[16]. When forming the discrete-time transfer function, often called the pulse-transfer function, the characteristics of the zero-order hold must be taken into account with the sampled response of $G_p(s)$. The relation between $H_{pq}(z)$ and $G_p(s)$ is given in [15],[16], as

$$H_{pq}(z) = (1 - z^{-1})ZT\left\{\frac{G_p(s)}{s}\right\}, \tag{1.19}$$

where T is the sampling period, and $ZT\left\{\frac{G_p(s)}{s}\right\}$ signifies taking the z-transform of the inverse Laplace-transform of $\frac{G_p(s)}{s}$, sampled with a sampling period of T_s. The inverse

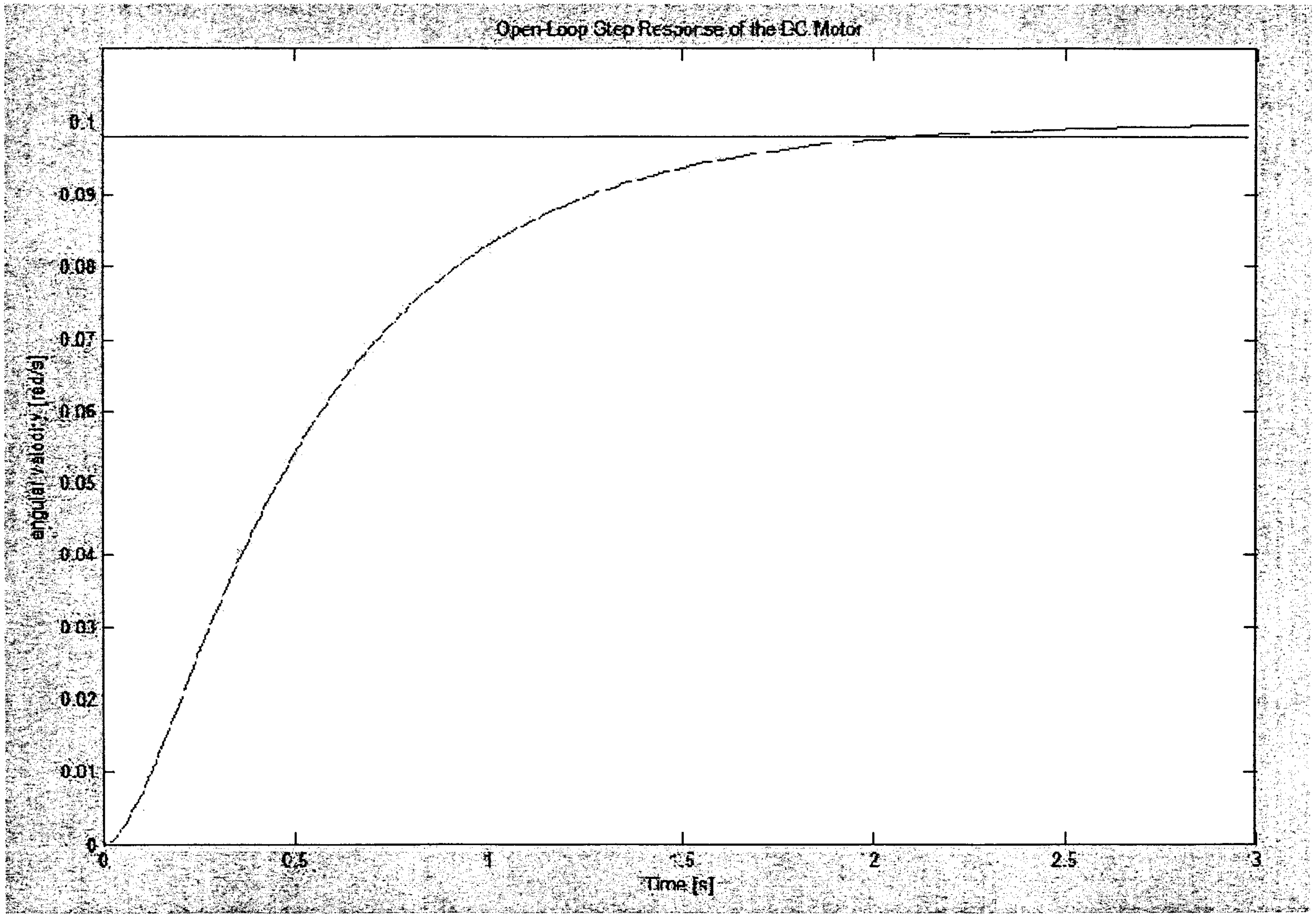

Figure 1.2: Open-Loop Step Response of the DC Motor.

Laplace transform of $\frac{G_p(s)}{s}$ is also called the step response, $g_{step}(t)$, of $G_p(s)$. Because the parameters of the continuous-time model of the motor will be changing, it is useful to derive the pulse-transfer function in terms of the continuous-time transfer function, which is of the form,

$$G_p(s) = \frac{B(s)}{A(s)} = \frac{b_0}{s^2 + a_1 s + a_2}. \tag{1.20}$$

$\frac{G_p(s)}{s}$ can be written as

$$\frac{G_p(s)}{s} = \frac{c_1}{s - p_1} + \frac{c_2}{s - p_2} + \frac{c_3}{s - p_3}, \tag{1.21}$$

where two of p_1, p_2, and p_3, are the roots of $A(s)$ and the other is 0, and c_1, c_2, and c_3 are the residues of $G_p(s)$. The step response is then,

$$g_{step}(t) = c_1 e^{p_1 t} + c_2 e^{p_2 t} + c_3 e^{p_3 t}. \tag{1.22}$$

A sampled-time version of the step response is

$$g_{step}(kT) = c_1 e^{p_1 kT} + c_2 e^{p_2 kT} + c_3 e^{p_3 kT}, \tag{1.23}$$

which can be written as

$$g_{step}(kT) = c_1 d_1^{\ k} + c_2 d_2^{\ k} + c_3 d_3^{\ k}, \tag{1.24}$$

where

$$d_i = e^{p_i T}. \tag{1.25}$$

The pulse-transfer function is $\frac{z-1}{z}$ times the z-transform of this function, which can be written as

$$H_{pq}(z) = \frac{b_0 z^2 + b_1 z + b_2}{z^2 + a_1 z + a_2}, \tag{1.26}$$

where

$$a_1 = 1 - d_1 - d_2 - d_3, \tag{1.27}$$

$$a_2 = d_1 d_2 d_3, \tag{1.28}$$

$$b_0 = c_1 + c_2 + c_3, \tag{1.29}$$

$$b_1 = -c_1(d_2 + d_3) - c_2(d_1 + d_3) - c_3(d_1 + d_2), \tag{1.30}$$

and

$$b_2 = c_1 d_2 d_3 + c_2 d_1 d_3 + c_3 d_1 d_2. \tag{1.31}$$

For the transfer function given in Equation 1.18, the pulse-transfer function is

$$H_{pq}(z) = \frac{0.961 \times 10^{-4} z + 0.9233 \times 10^{-4}}{z^2 - 1.885z + 0.8869} \tag{1.32}$$

with the sampling period, T_s, of $0.01s$. Figure 1.3 shows the step response of the pulse-transfer function on the same axes as the step response of the transfer function. The pulse-transfer function does a good job of approximating the transfer function.

The system can be written in terms of the forward shift operator, q, where

$$qf(k) \equiv f(k+1). \tag{1.33}$$

The system can now be written as

$$A_p y(k) = B_p u(k), \tag{1.34}$$

where

$$A_p = A_p(q) = q^2 + a_1 q + a_2, \tag{1.35}$$

and

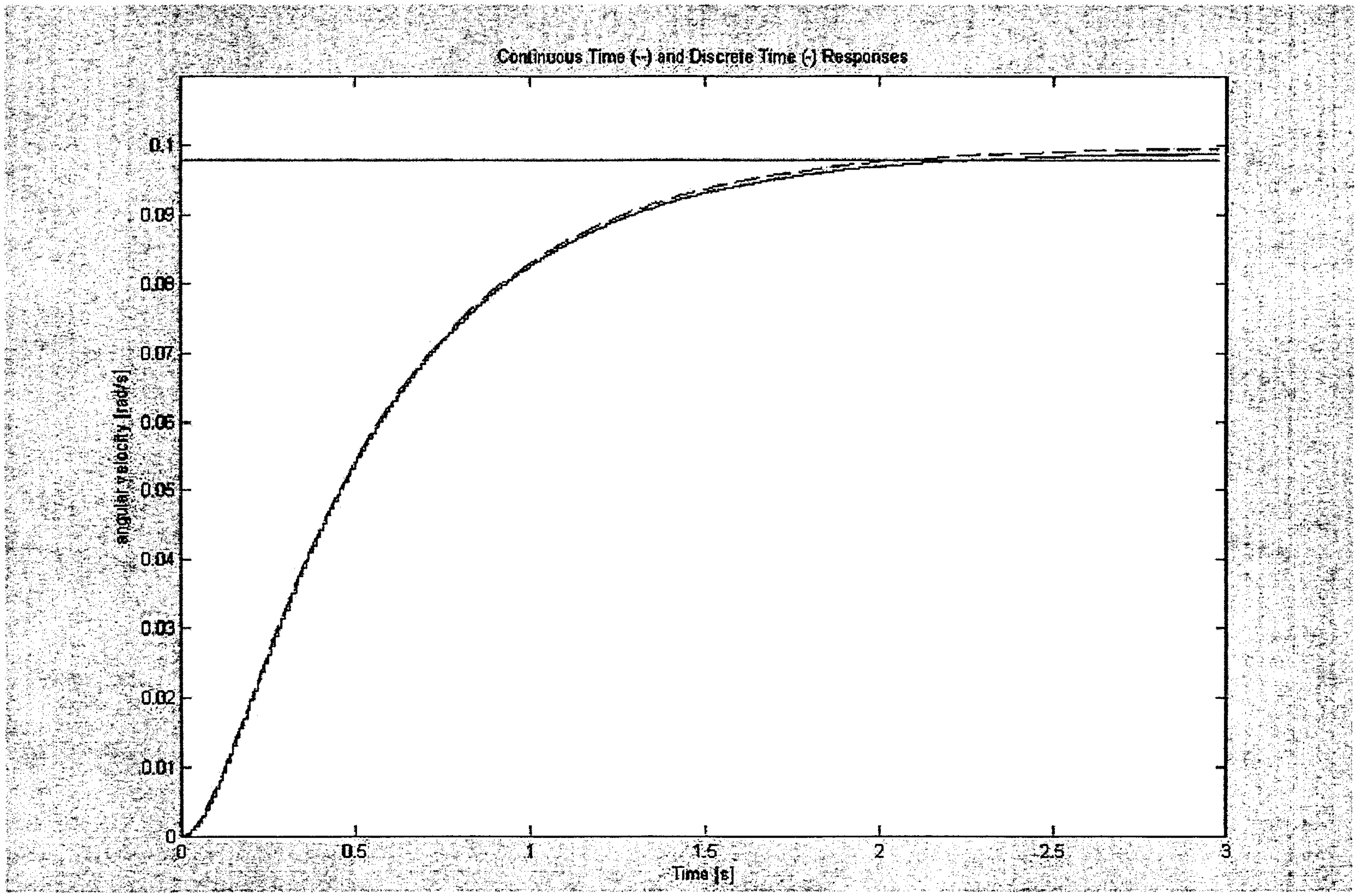

Figure 1.3: Open-Loop Step Response for the Continuous-Time and Pulse-Transfer Functions.

$$B_p = B_p(q) = b_0 q^2 + b_1 q + b_2 . \tag{1.36}$$

It is convenient in determining the response of the plant to represent the system in terms of the reverse shift operator, or delay operator, such that the system becomes

$$A_p^*(q^{-1}) y(k) = B_p^*(q^{-1}) u(k) , \tag{1.37}$$

where

$$q^{-1} f(k) \equiv f(k-1) , \tag{1.38}$$

$$A_p^*(q^{-1}) = 1 + a_1 q^{-1} + a_2 q^{-2} , \tag{1.39}$$

and

$$B_p^*(q^{-1}) = b_0 + b_1 q^{-1} + b_2 q^{-2} . \tag{1.40}$$

1.5 Characteristics of the Open-Loop Responses

Characteristics of the open-loop system must be analyzed to determine which specific portions of its performance need to be improved. Because the poles are in the left-hand plane, the system is stable, so stabilizing the system is not a design criterion, although it is imperative to maintain the system's stability. Several characteristics about the open-loop step response can be established by looking at the transfer function in terms of the natural frequency, ω_n, and the damping ratio, ζ. These constants can be determined from the transfer function by noting that for a second order system,

$$G(s) = \frac{\omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \tag{1.41}$$

holds [8],[14]. For this system, the natural frequency is $4.47\frac{rad}{s}$, and the damping ratio is 1.34. The damping ratio indicates that this system is overdamped. From Figure 1.2, it is apparent that the settling time is $2.1s$ and that the steady-state error is 90.091%.

1.6 Design Criteria

The desired performance of the system involves the steady-state error, overshoot and the settling time. A reasonable value for the overshoot is 15%. Using the relation,

$$\zeta = \frac{-\ln(\%OS/100)}{\sqrt{\pi^2 + \ln^2(\%OS/100)}}, \tag{1.42}$$

ζ, the damping ratio, is found to be 0.5169. The system is to be sped up to have a settling time, T_s, of $1s$. This implies, through use of the relation,

$$T_s = \frac{4}{\zeta \cdot \omega_n}, \tag{1.43}$$

a natural frequency, ω_n, of $7.738\frac{rad}{s}$. Using these values in Equation 1.41 gives the desired performance as

$$G_m(s) = \frac{59.88}{s^2 + 8.0s + 59.88} \tag{1.44}$$

Figure 1.4 shows the step response of this transfer function. From Figure 1.4, it can be seen that the overshoot is 5%, and the settling time is about $1s$, meeting the desired design criteria.

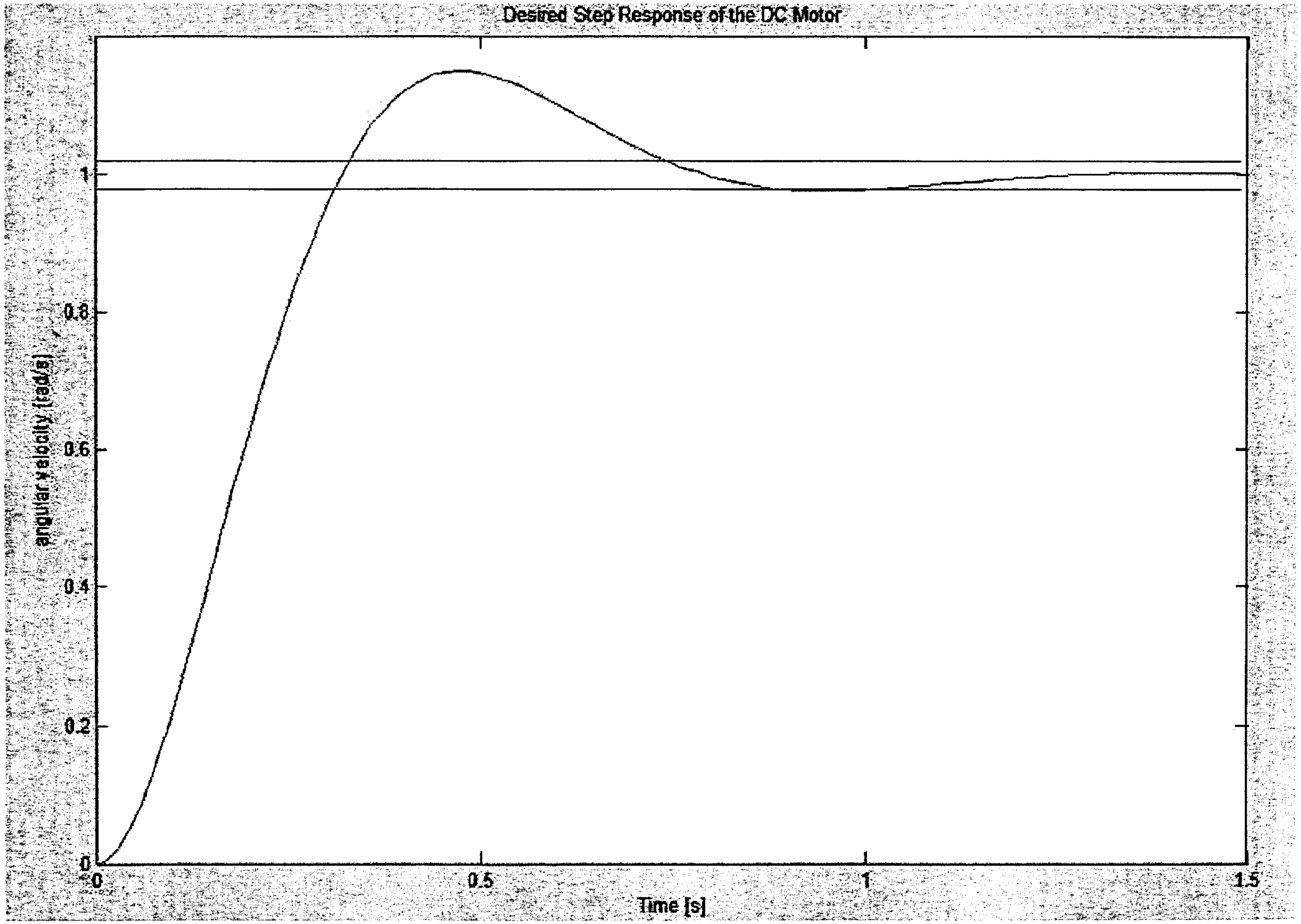

Figure 1.4: Desired Step Response of the DC Motor.

Using the procedure that was presented in Section 1.3, the pulse-transfer function of the desired transfer function can be obtained. Using a sampling time of $0.01s$, the pulse transfer function is

$$H_{m_{pq}}(z) = \frac{0.001638z + 0.001594}{z^2 - 1.92z + 0.9231}. \tag{1.45}$$

It can be observed that the pulse-transfer functions' step response is a reasonable approximation of the step response of the continuous-time step response; Figure 1.5 shows the step responses of both the pulse-transfer function and transfer function on the same axis.

1.7 Varying the Load of the Motor

The prior calculations involved the motor in an unloaded state. When a load is added to the motor, the moment of inertia, J, of the motor will effectively be changed. It will be assumed the connection of the motor and the load will be perfectly attached, that is there is no slipping between the motor and the load. This change in J changes the location of the poles of the transfer function, which will change the nature of the system. For example, if J is 0.0471, then ζ is 1.0, resulting in a critically damped system. If J is 0.0501, then ζ is 0.995; then the system becomes overdamped. For the situation where J is 0.0471, the transfer function is

$$G_p^{'}(s) = \frac{0.4246}{s^2 + 4.123s + 4.251} = \frac{0.4246}{(s + 2.06 + j0.0349)(s + 2.06 - j0.0349)} \tag{1.46}$$

and the pulse-transfer function is

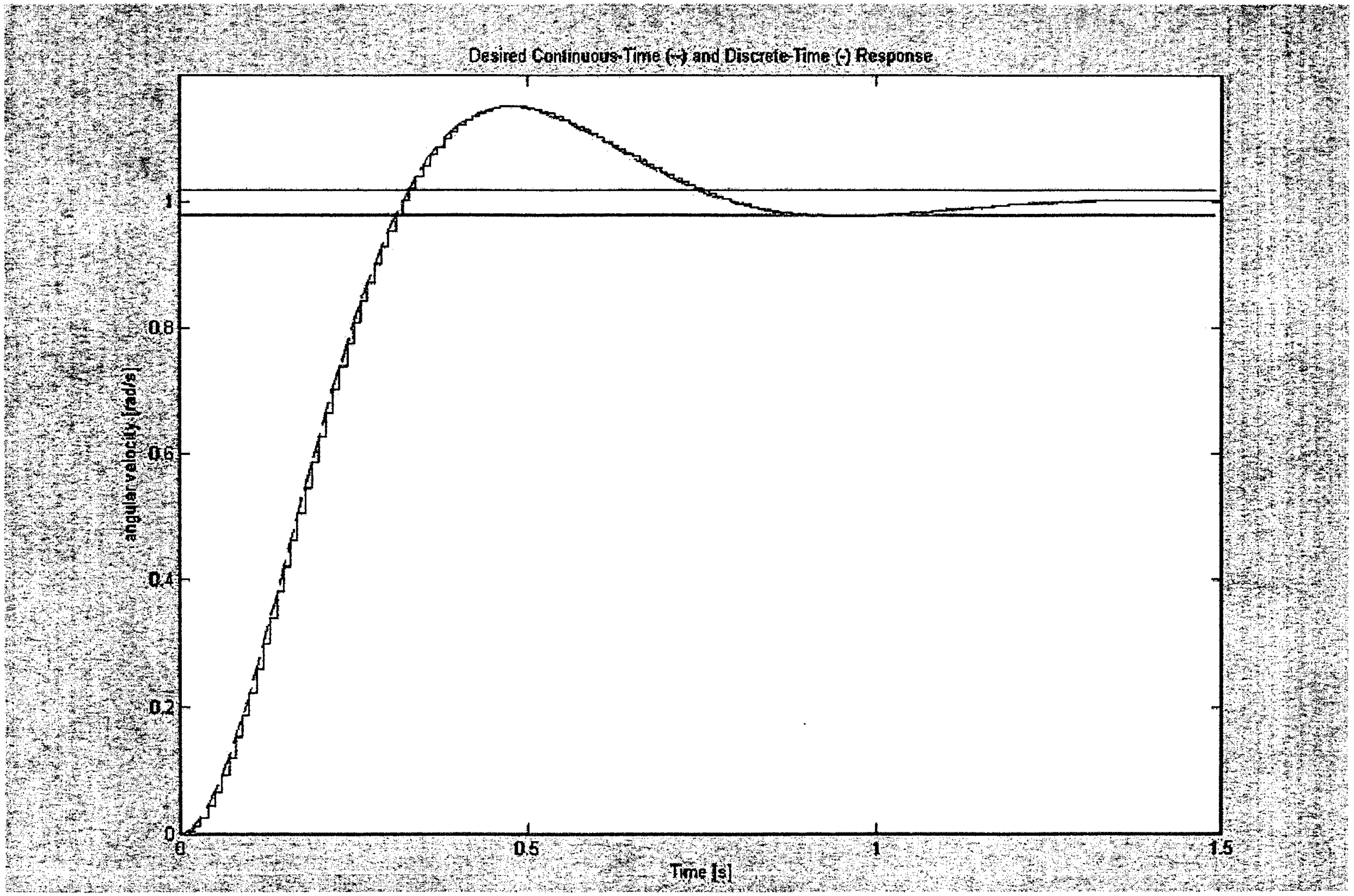

Figure 1.5: Desired Step Response for the Continuous-Time and Pulse-Transfer Functions.

$$H_{pq}(z) = \frac{20.94\times10^{-6}z + 20.66\times10^{-6}}{z^2 - 1.959z + 0.9596} = \frac{20.94\times10^{-6}(z+0.9866)}{(z-0.98+j0.0134)(z-0.98-j0.0134)}. \quad (1.47)$$

For a controller using classical control techniques, this would result in a poor response, as will be shown in Chapter 6.

CHAPTER II

PARAMETER ESTIMATION

2.1 Least Squares Estimation

Because all of the system parameters may not be known or may vary from the known values in an adaptive system, some form of parameter estimation must be enacted. For this discussion, linear estimation will be emphasized. For this situation, the measurement equation has the general form

$$\mathbf{Z} = \mathbf{H}\mathbf{x} + \bar{\varepsilon}, \tag{2.1}$$

where $\mathbf{Z}$ is the noisy observation, $\mathbf{x}$ is a vector of the parameters to be estimated, $\bar{\varepsilon}$ is the error, and $\mathbf{H}$ is the linear function operating on $\mathbf{x}$ to get the measured quantity [3]. The noise-free measurement is a linear function of the parameters to be estimated.

Least squares linear estimation is a common method for determining parameters in an adaptive control scheme. In the general case, the measurement is of the form of Equation (2.1), and the distribution of $\bar{\varepsilon}$ is unknown. The principle of least squares is that errors that are present in measurements tend to be minimized over time, and thus a logical estimate for the parameters in $\mathbf{x}$ are that values for which the sum of the squares of the error is minimum. In mathematical terms, the loss function,

$$J = \varepsilon_1^2 + \varepsilon_2^2 + \cdots + \varepsilon_N^2 = \bar{\varepsilon}^T \bar{\varepsilon} = [\mathbf{Z} - \mathbf{H}\mathbf{x}]^T [\mathbf{Z} - \mathbf{H}\mathbf{x}], \tag{2.2}$$

should be minimized. Because the function, $\mathbf{H}$, is linear, this loss function can be rewritten as

$$J = [\mathbf{Z}-\mathbf{Hx}]^T[\mathbf{Z}-\mathbf{Hx}] = \mathbf{Z}^T\mathbf{Z} - 2\mathbf{Z}^T\mathbf{Hx} + \mathbf{x}^T\mathbf{H}^T\mathbf{Hx}. \tag{2.3}$$

Because the choice of the parameters, $\mathbf{x}$, are to result in a minimal loss function, the loss function, J, must be differentiated with respect to $\mathbf{x}$, and equating the result with zero. Solving the equation,

$$\frac{\partial J}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial J}{\partial x_1} & \frac{\partial J}{\partial x_2} & \cdots & \frac{\partial J}{\partial x_N} \end{bmatrix} = -2\mathbf{Z}^T\mathbf{H} + \mathbf{x}^T\mathbf{H}^T\mathbf{H}, \tag{2.4}$$

for $\mathbf{x}$, yields the least squares estimate, denoted by $\hat{\mathbf{x}}$ [3],

$$\hat{\mathbf{x}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{Z}. \tag{2.5}$$

This can be written as [1]

$$\hat{\mathbf{x}}(k) = \left(\sum_{i=1}^{k} h_i h^T{}_i\right)^{-1}\left(\sum_{i=1}^{k} h_i z_i\right), \tag{2.6}$$

where h_i is the i^{th} member of $\mathbf{H}$.

2.2 Recursive Least Squares Estimation

Because most of the parameter estimation in adaptive controllers must be done in real time, a recursive parameter estimation is used to save computation time [1],[7]. The definition

$$P(k) = \left(\mathbf{H}^T(k)\mathbf{H}(k)\right)^{-1} = \left(\sum_{i=1}^{k} h_i h^T{}_i\right)^{-1} \tag{2.7}$$

is introduced. Equation 2.6 can now be written as

$$\hat{\mathbf{x}}(k) = P(k)\left(\sum_{i=1}^{k} h_i z_i\right) = P(k)\left(\sum_{i=1}^{k-1} h_i z_i + h(k)z(k)\right). \tag{2.8}$$

However,

$$P^{-1}(k) = \mathbf{H}^T(k)\mathbf{H}(k) = \sum_{i=1}^{k} h_i h^T{}_i, \tag{2.9}$$

which follows from the definition in Equation 2.7, and can be expanded into

$$P^{-1}(k) = \sum_{i=1}^{k-1} h_i h^T{}_i + h(k)h^T(k), \tag{2.10}$$

which can be simplified as

$$P^{-1}(k) = P^{-1}(k-1) + h(k)h^T(k). \tag{2.11}$$

Equation 2.8 leads to

$$\sum_{i=1}^{k-1} h_i h^T{}_i = P^{-1}(k-1)\hat{\mathbf{x}}(k-1), \tag{2.12}$$

which upon using a rearranged Equation 2.11 to substitute for $P^{-1}(k-1)$ results in

$$\sum_{i=1}^{k-1} h_i h^T{}_i = P^{-1}(k)\hat{\mathbf{x}}(k-1) - h(k)h^{-1}(k)\hat{\mathbf{x}}(k-1). \tag{2.13}$$

The estimate for time k can now be written in terms of the estimate for time $k-1$ as

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}(k-1) - P(k)h(k)h^T(k)\hat{\mathbf{x}}(k-1) + P(k)h(k)z(k), \tag{2.14}$$

which can be rearranged into

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}(k-1) + P(k)h(k)\left(z(k) - h^T(k)\hat{\mathbf{x}}(k-1)\right). \tag{2.15}$$

The error, $\varepsilon(k)$, associated with the prediction of $z(k)$ based on the estimate $\hat{\mathbf{x}}(k-1)$ is

$$\varepsilon(k) = z(k) - h^T(k)\hat{\mathbf{x}}(k-1). \tag{2.16}$$

Equation 2.16 can be written as

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}(k-1) + R(k)\varepsilon(k), \tag{2.17}$$

where $R(k)$ is defined as

$$R(k) = P(k)h(k). \tag{2.18}$$

In order to use these results, a recursive relation for $P(k)$ must be found rather than for $P^{-1}(k)$. This can be done by using the matrix inversion lemma found in Appendix A, which states

$$(A + BCD)^{-1} = \left(A^{-1} - A^{-1}B\left(C^{-1} + DA^{-1}B\right)^{-1} DA^{-1}\right). \tag{2.19}$$

This lemma is used by substituting $P^{-1}(k-1)$ for A, the identity matrix, I, for C, $h(k)$ for B, and $h^T(k)$ for D. This choice satisfies the required conditions for the matrix inversion lemma, as $P^{-1}(k-1)$, I, and $h(k)Ih^T(k)$ are nonsingular matrices. Using Equation 2.19 in conjunction with Equation 2.11 results in

$$P(k) = P(k-1) - P(k-1)h(k)\left(I^{-1} + h^T(k)P(k-1)h(k)\right)^{-1} h^T(k)P(k-1). \tag{2.20}$$

Using

$$R(k) = P(k-1)h(k)\left(I^{-1} + h^T(k)P(k-1)h(k)\right)^{-1}, \tag{2.21}$$

the estimation of parameters can be summarized as follows:

First find $R(k)$, using Equation 2.21 based on the choice of initial conditions for parameters and matrix, $P(0)$. Next, determine $P(k)$ using

$$P(k) = \left(I - R(k)h^T(k)P(k-1)\right), \tag{2.22}$$

which is found by substituting Equation 2.21 into Equation 2.20. The residual error in the system can be then calculated as

$$\varepsilon(k) = z(k) - h^T(k)\hat{\mathbf{x}}(k-1). \tag{2.23}$$

Finally, the parameters may be estimated using

$$\hat{\mathbf{x}}(k) = \hat{\mathbf{x}}(k-1) + R(k)\varepsilon(k). \tag{2.24}$$

The recursive least squares estimation technique can be used to estimate the coefficients in the transfer function of a system. The transfer function of the system is assumed to be of the form

$$A(q)y(k) = B(q)u(k), \tag{2.25}$$

where q is the forward shift operator. Equation 2.25 can be rewritten as

$$a_n y(k-n) + \cdots + a_1 y(k-1) + y(k) = b_m u(k-m) + \cdots + b_1 u(k-1) + b_0 u(k), \tag{2.24}$$

which can be solved for $y(k)$ as

$$y(k) = b_m u(k-m) + \cdots + b_1 u(k-1) + b_0 u(k) - a_n y(k-n) - \cdots - a_1 y(k-1). \tag{2.25}$$

Let

$$\phi(k) = h(k) = (u(k-m) \quad \cdots \quad u(k) \quad y(k-n) \quad \cdots \quad y(k-1))^T \tag{2.26}$$

and

$$\theta(k) = \mathbf{x}(k) = (b_m \quad \cdots \quad b_0 \quad a_n \quad \cdots \quad a_1)^T. \tag{2.27}$$

Equation 2.25 can now be written as

$$y(k) = \phi^T(k)\theta(k), \tag{2.28}$$

which is of the form of Equation 2.1.

2.3 MATLAB Code

The MATLAB code for estimating the parameters at each time, m, is as follows. It is assumed that there is an initial guess, theta(:,1), an initial confidence matrix, P(:,:,1), an initial K vector, K(:,1), the output at times m-1 and ,m-2, that is y(m-1) and y(m-2), and the inputs at times m-1 and m-2, which are u(m-1),u(m-2). Note that the algorithm must be initialized by providing an initial guess for the parameters, theta(:,:). This example is for a second order system, with $m = 2$ and $n = 1$.

```
% Estimate Parameters
 Pkm1=P(:,:,m-1);
  phikm1=[-y(m-1) -y(m-2) u(m-1) u(m-2)]';
  thetakm1=theta(:,m-1);
  e(m)=y(m)-phikm1'*thetakm1;
  K(:,m)=Pkm1*phikm1*inv(lambda+phikm1'*Pkm1*phikm1);
  P(:,:,m)=(diag([1 1 1 1])-K(:,m)*phikm1')*Pkm1/lambda;
  theta(:,m)=thetakm1+K (:,m)*e(m);
  a1=theta(1,m);
  a2=theta(2,m);
  b0=theta(3,m);
  b1=theta(4,m);
```

CHAPTER III

CLASSICAL CONTROL TECHNIQUES

In this chapter classical control techniques are explored for the dc motor modeled in Chapter One. At first a discrete-time lag-lead controller was designed, but it was impossible to meet the design requirements with such a controller. Either the settling time was lengthened or the error was unacceptable. In what follows a classical controller designed based on the traditional Root Locus technique will be presented.

3.1 Introduction to Root Locus

One technique that can be used to design a controller for a continuous-time system is the Root Locus method. A Root Locus is a graph that shows the positions of the closed-loop poles as the system gain, K, is varied [8],[11],[14]. This technique can be used for discrete-time systems as well because the characteristic equation used is both discrete-time and continuous-time is of the same form, only in different complex variables and domains. The stability boundary is the unit circle for discrete-time systems rather than the $j\omega$-axis, as it is in continuous time case. Likewise, the curve of constant parameters such as damping ratio and natural frequency, are changed from continuous-time to discrete-time systems; a cartoid is the curve of constant damping ratio in discrete-time instead of a line radiating from the origin and a circle centered at $z=1$ is the curve of constant natural frequency in the z-domain [15],[16]. Figure 3.1 depicts the block diagram of the system that is used for Root Locus analysis and design.

3.2 Designing the Compensator

The first step in designing the compensator is to construct the Root Locus for the uncompensated system,

$$H_{pq}(z)=\frac{96.1\times10^{-6}z+92.33\times10^{-6}}{z^2-1.885z+0.8869}=96.1\times10^{-6}\frac{z+0.9608}{(z-0.9800)(z-0.9050)}. \quad (3.1)$$

Figure 3.2 shows the Root Locus for this system. Initially, with a gain of zero, the system is stable and remains stable. If the gain is increased to about $1{,}360$, the system becomes unstable. Three system response parameters are important: no steady-state error in the step response, an overshoot of less than 15%, and a settling time of at most $1s$. In order to have zero steady-state error, integrating action is needed. This is accomplished by placing a pole at $z=1$ [16]. This, however destabilizes the system as can be seen from the Root Locus depicted in Figure 3.3 which is derived by adding a pole at $z=1$. This is because poles tend to repel the branches of a Root Locus, and there are two poles in the near vicinity of $z=1$. One solution is to cancel the other pole at $z=0.9800$ by placing a zero at the same place. Pole-zero cancellation is not perfect, however, the pole will be effectively cancelled if the zero is very close to the same location [15]. The compensator is so far given by

$$D(z)=K\frac{z-0.9800}{z-1}. \quad (3.2)$$

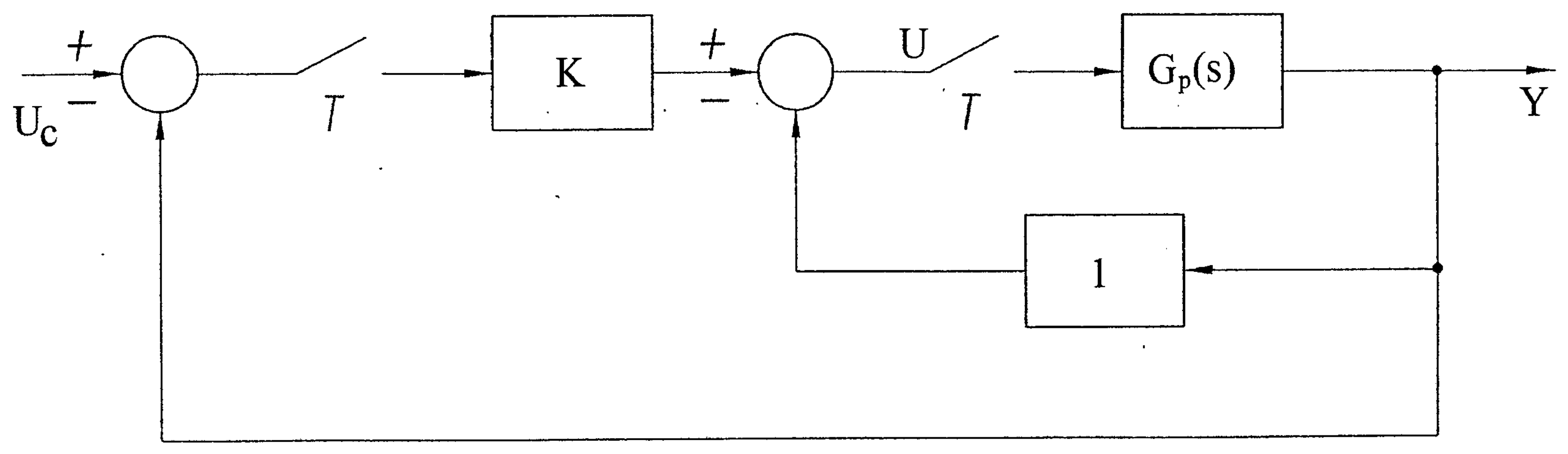

Figure 3.1: Block Diagram of Classical Controller.

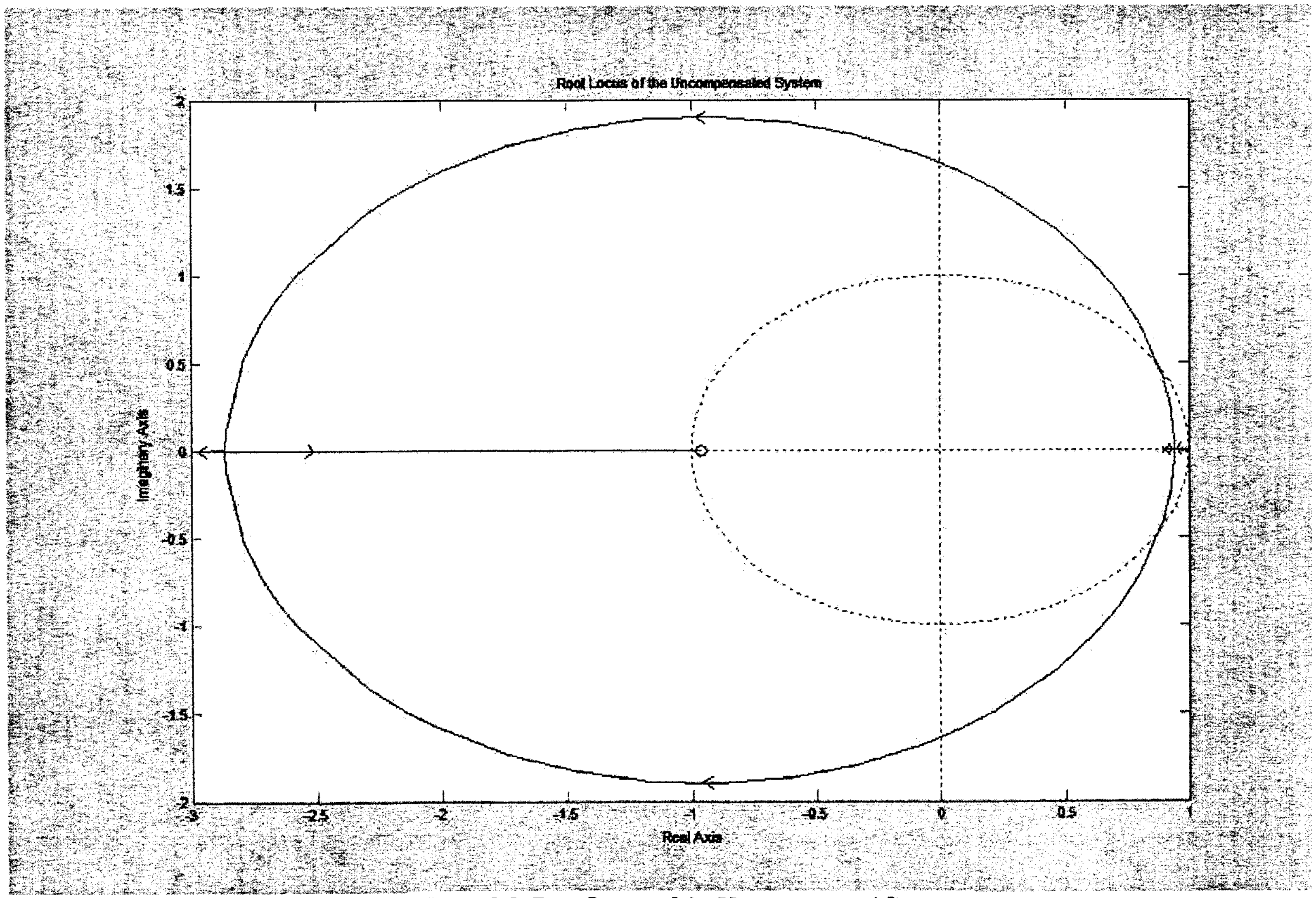

Figure 3.2: Root Locus of the Uncompensated System.

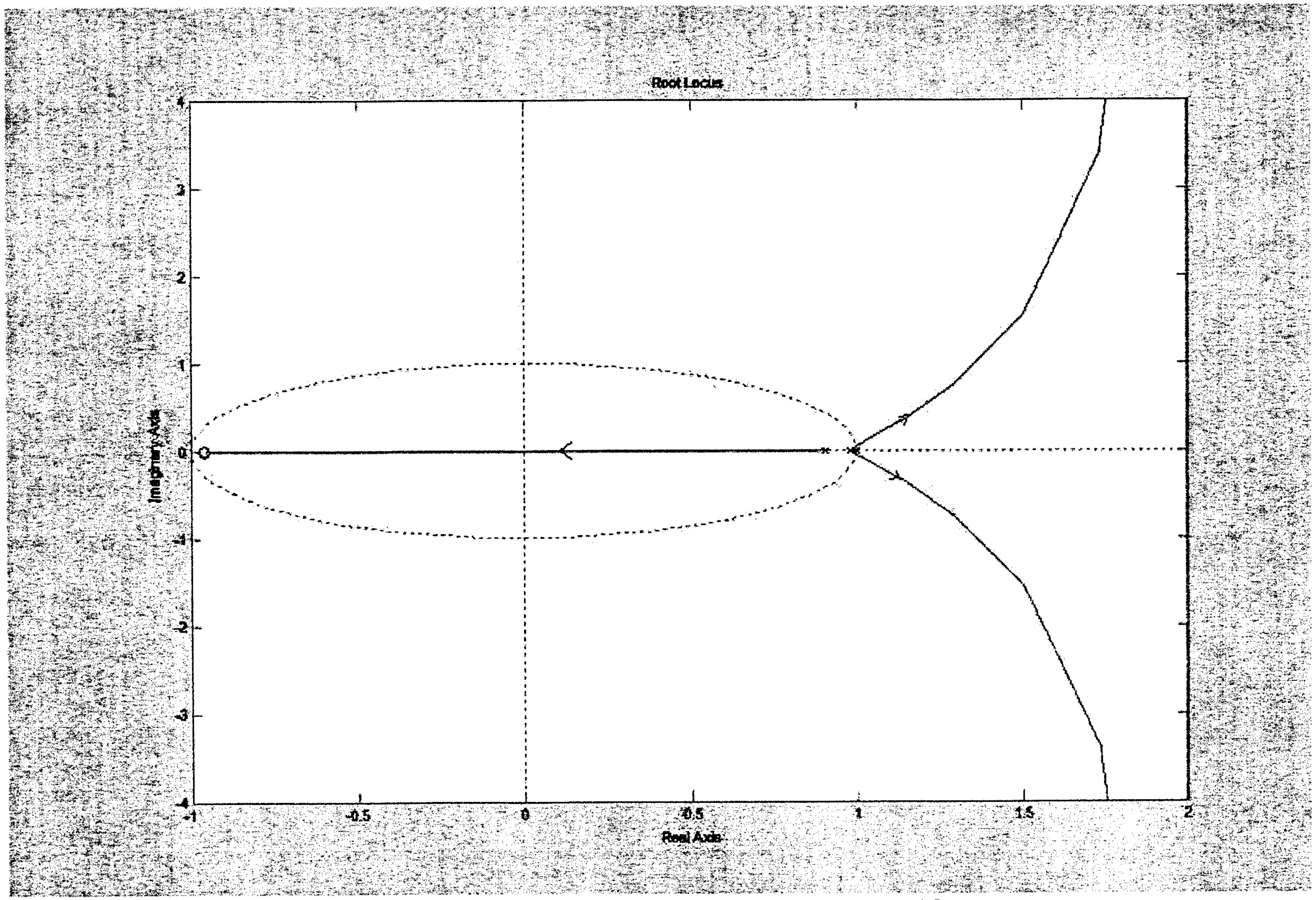

Figure 3.3: Root Locus of the Integrator Compensated System.

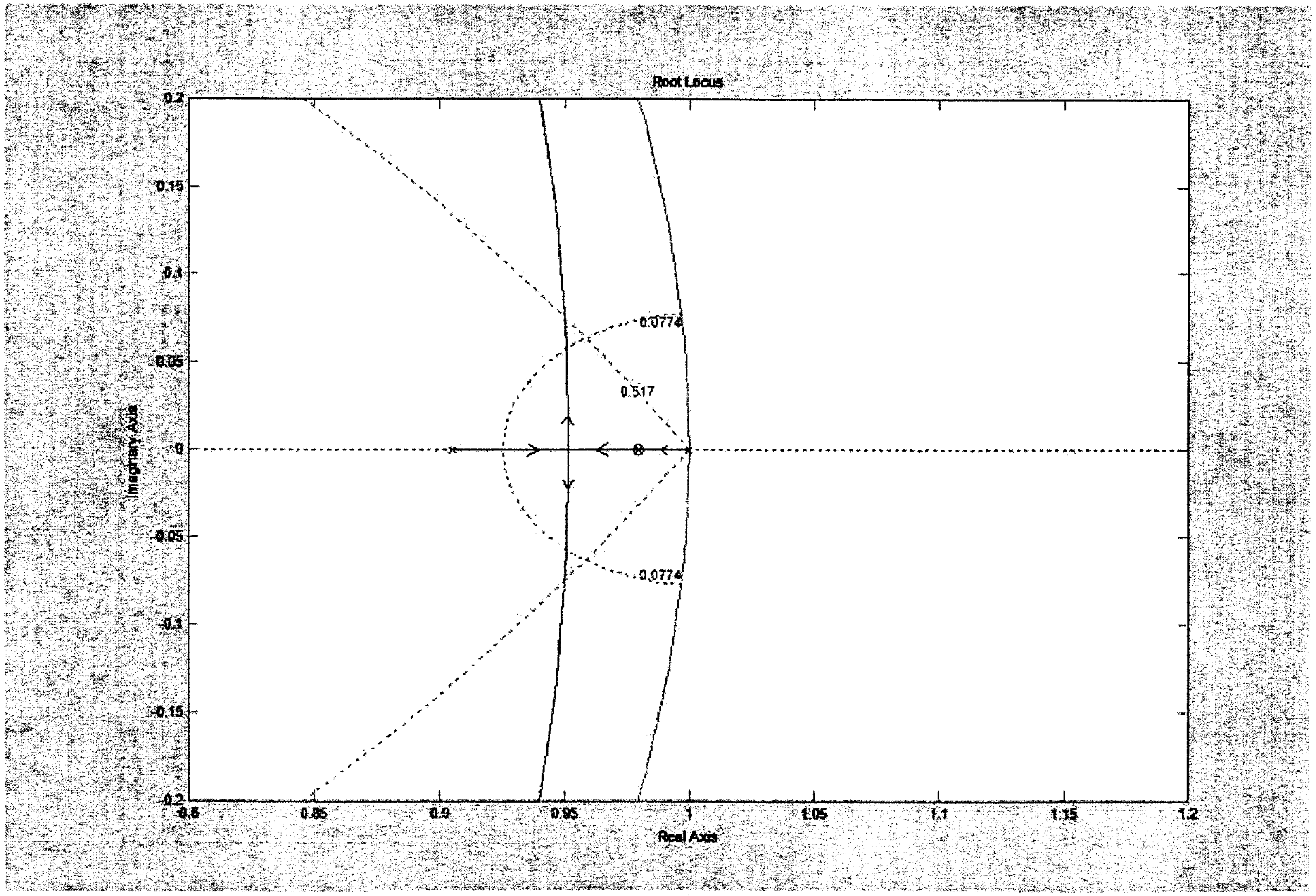

Figure 3.4: Root Locus of the Classically-Compensated System.

The Root Locus of the compensated system is shown in Figure 3.4. In this Root Locus, the curves of constant damping ratio at 0.5169 and constant natural frequency of 0.07738 have been added. The desired natural frequency for a discrete time must be in terms of

$\frac{rad}{sample}$, so the natural frequency must be multiplied by the sampling period. In order for the compensated system to exactly meet the design requirements, the Root Locus must intersect at the same point as the two curves of constant parameter, which is not the case. The control design is acceptable if all the criteria are met or exceeded. Where the Root Locus and the curve of constant damping ratio intersect, the gain is 42.8. Figure 3.5 shows the step response when the compensator gain is 42.8. The percent overshoot is about 15%, however the settling time requirement is not met; the settling time is about $1.2s$. The other option is to use the gain 30.2, which is the gain where the Root Locus and the curve of constant natural frequency meet. Because this point is inside the curve of constant damping ratio, the damping ratio is greater, thus the overshoot is less. Figure 3.6 shows the step response when the compensator gain is 30.2. The settling time is met exactly, and the percent overshoot is 8%, which exceeds the design requirement. The compensator given by

$$D(z) = 30.2\frac{z - 0.9800}{z - 1} \tag{3.3}$$

will be used in the discussions on disturbances and varied loads in Chapter 6.

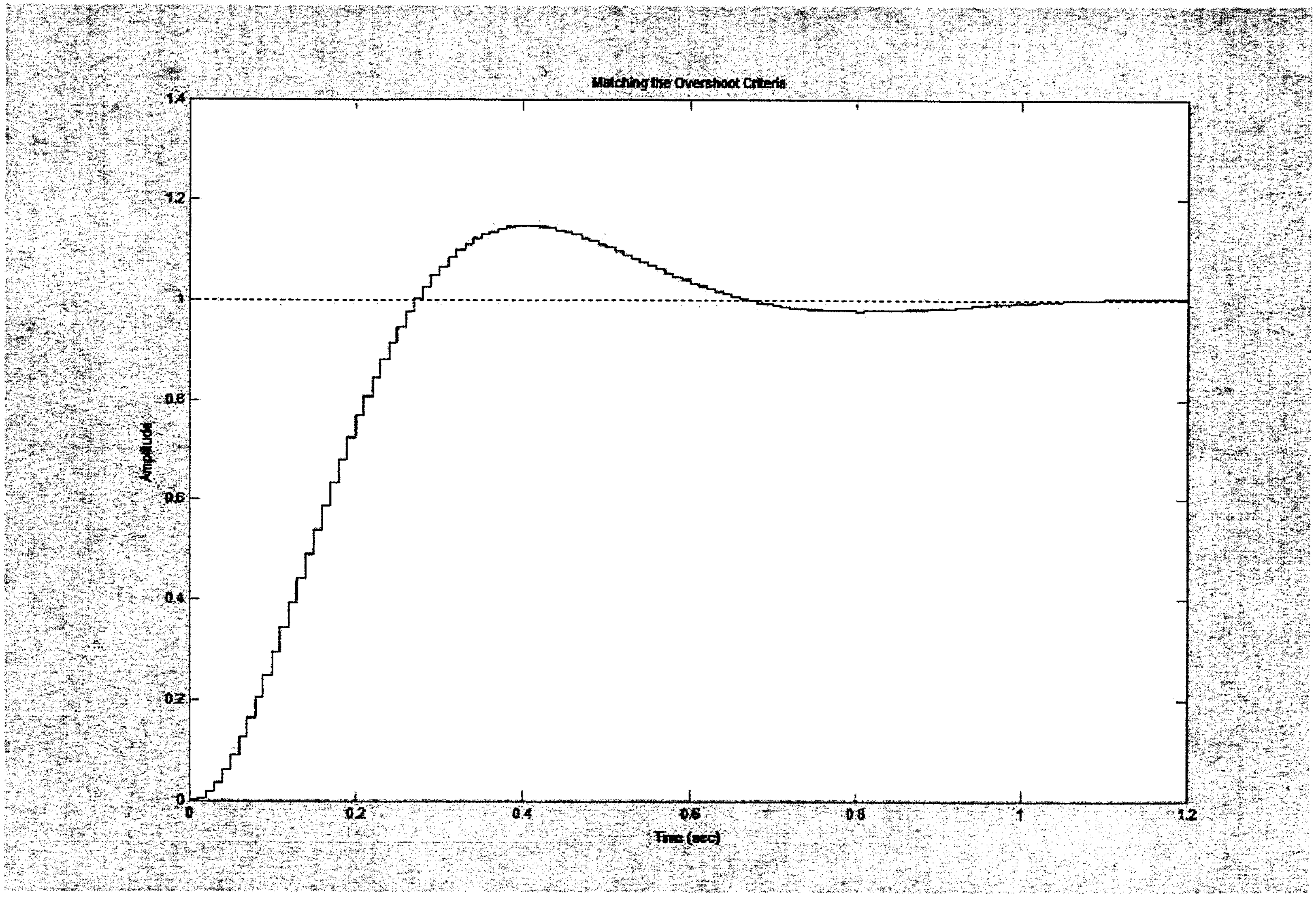

Figure 3.5: Step Response of the System with Controller Exactly Meeting the Damping Ratio Requirement.

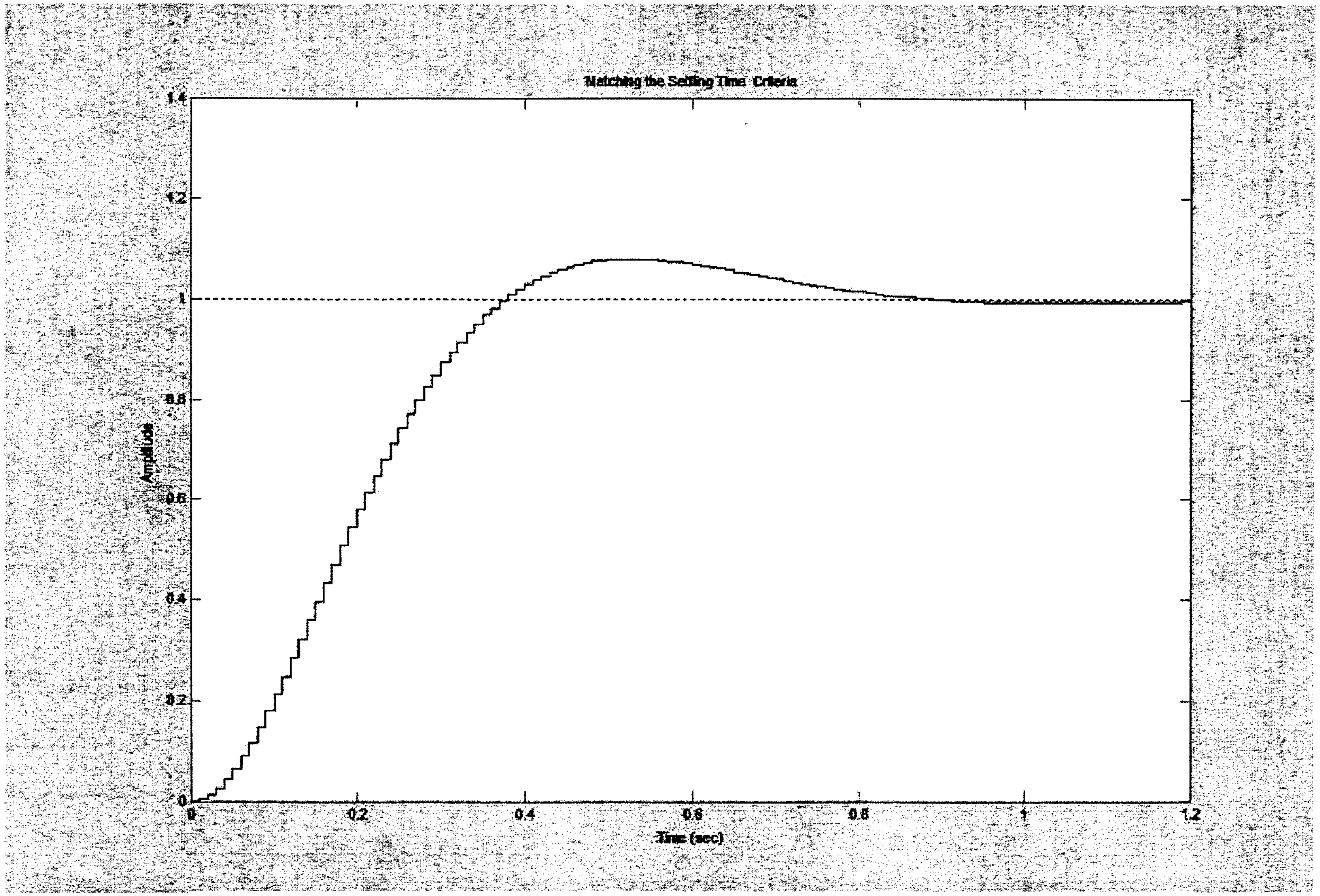

Figure 3.6: Step Response of System with Controller Exactly Meeting the Settling Time Requirement.

CHAPTER IV

SELF-TUNING REGULATORS

The self-tuning regulator is the first adaptive control scheme that will be used to control the dc motor. First, the process for designing a self-tuning regulator will be presented. Then the ramifications of designing a controller with pole-zero cancellation as well as one that has no cancellation will be observed. Finally, two self-tuning regulators will be designed and compared with respect to meeting the design criteria and the control effort.

4.1 Introduction to Self-Tuning Regulation

A self-tuning regulator is a control scheme that is useful when the environment in which the controller operates changes. The block diagram of a general self-tuning regulator is shown in Figure 4.1 [1]. The process is the system which is to be controlled. The estimation block involves the estimation of the process parameters, which are fed into the controller design block. These parameters, along with the control specification, determine the control parameters. The controller output, also called the control signal, which drives the plant changes at each time interval based on the output of the system,

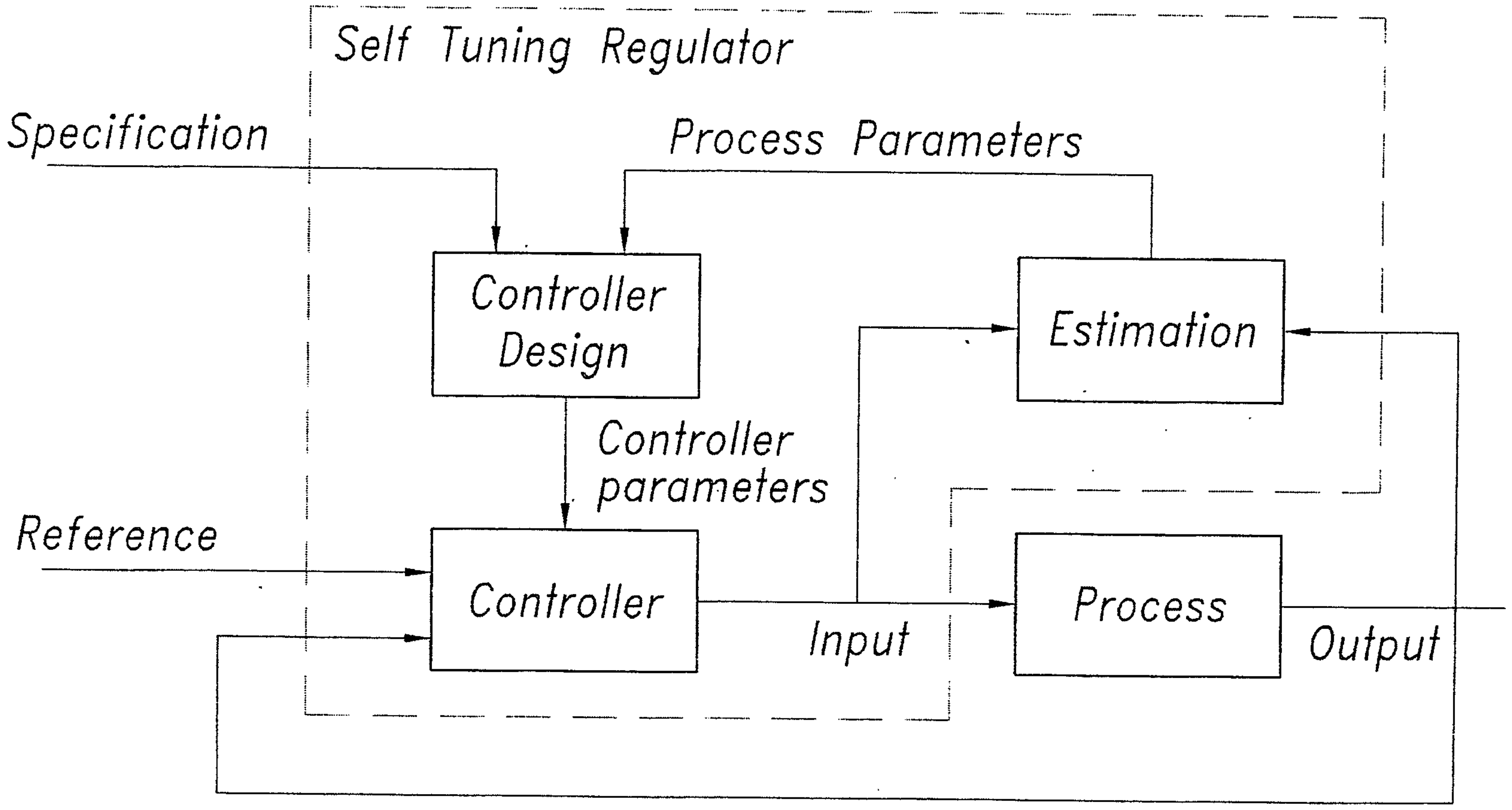

Figure 4.1: Block Diagram for Self-Tuning Regulator.

the control parameters, and the input to the system. For this discussion, only least squares estimation, as discussed in chapter 2, will be considered.

4.2 Controller Design

The design technique for a self-tuning regulator is given by Astrom and Wittenmark [1], which this development basically follows. In order to design the controller, pole placement is utilized. The model of the system is assumed to be

$$A(q)y(t) = B(q)(u(t) + v(t)), \tag{4.1}$$

where y is the output, u is the input, and v is a disturbance. The polynomials $A(q)$ and $B(q)$ have degrees of n and m, respectively. The pole excess of the system, d_0, is the difference between n and m. Two assumptions about A and B have been made; the coefficient of the highest term with highest degree in A is one, that is, A is monic, and A and B have no common factors, which means that they are relatively prime. These assumptions serve to make the generalization of the design process easier. In general, $A(q)$ will be written as A.

The controller can be written as

$$R \cdot u(t) = T \cdot u_c(t) - S \cdot y(t), \tag{4.2}$$

where R, S, and T are polynomials, and $u(t)$ is the signal fed into the system, and $u_c(t)$ is the reference signal that the system is trying to follow. Combining Equation 4.1 and Equation 4.2 results in

$$AR \cdot y(t) = B(t \cdot u_c(t) - S \cdot y(t) + R \cdot v(t)), \tag{4.3}$$

which simplifies to

$$y(t) = \frac{BT \cdot u_c(t) + BR \cdot v(t)}{AR + BS}. \tag{4.4}$$

This gives the closed-loop characteristic polynomial, A_{CL}, as

$$A_{CL} = AR + BS. \tag{4.5}$$

Equation 4.5 is called the Diophantine equation [1], and can be used to determine R and S. In order to determine T, the response of the desired system to $u_c(t)$, which is described by

$$A_m y_m(t) = B_m u_c(t), \tag{4.6}$$

is used, where $y_m(t)$ is the desired output, A_m and B_m are the numerator and denominator of the desired transfer function. The condition,

$$\frac{BT}{AR + BS} = \frac{BT}{A_{CL}} = \frac{B_m}{A_m}, \tag{4.7}$$

must hold for the system to follow the model.

In order for the controller to be causal, the conditions

$$\deg S \leq \deg R \tag{4.8}$$

and

$$\deg T \leq \deg R, \tag{4.9}$$

must be true. Equation 4.5 has numerous solutions, thus it is possible to select the solution which gives the controller with the lowest degree. Because n is greater than m, and the degree of S cannot be greater than the degree of R, the relationship

$$\deg R = \deg A_{CL} - n \tag{4.10}$$

can be used to find the degree of R. Because it is undesirable to have extra delays in the controller, R, S, and T are required to have the same degree.

This design allows some cancellation of process zeroes with poles of the controller. In order for the zeroes to be cancelled, they must be both stable and well-damped, lest the controller introduce unstable or poorly damped poles to the closed-loop system. The polynomial B can be factored as

$$B = B^{+}B^{-}, \tag{4.11}$$

where B^{+} consists of the monic polynomial whose roots are stable and well-damped, while B^{-} consists of the polynomial whose roots are either unstable or poorly-damped. If the zeroes are not cancelled, then they must appear in the desired zeroes. This implies

$$B_{m} = B^{-}B_{m}^{'}, \tag{4.12}$$

where $B_{m}^{'}$ is the desired numerator with the desired cancelled zeroes, B^{-}, factored out. Because B^{+} is cancelled, it must be a factor of A_{CL}, thus

$$A_{CL} = A_{0}A_{m}B^{+} \tag{4.13}$$

is true. Because B^{+} is a factor of both A_{CL} and B, but it is not a factor of A, B^{+} must be a factor of R, which can be written as

$$R = R'B^{+}, \tag{4.14}$$

where R' is the feedback relation with B^{+} factored out of it.

Equation 4.5 reduces to

$$AR' + B^{-}S = A_{0}A_{m}, \tag{4.15}$$

and T can be determined from

$$T = A_{0}B_{m}^{'} . \tag{4.16}$$

This leads to three types of controllers where either some zeroes are cancelled, all zeroes are cancelled or no zeroes are cancelled.

If all the process zeroes are cancelled, then the degree of A_0 is $d_0 - 1$. Choosing

$$B_m = A_m(1)q^{d_0} \tag{4.17}$$

results in the factorization of Equation 4.12 to be

$$B^+ = \frac{B}{b_0} \tag{4.18}$$

and

$$B^- = b_0. \tag{4.19}$$

The Diophantine equation becomes

$$AR' + b_0 S = A_0 A_m, \tag{4.20}$$

and T is

$$T = \frac{A_m(1)}{b_0} q^{d_0}. \tag{4.21}$$

If no zeroes are cancelled, then the degree of A_0 is $d_0 - 1$. Choosing

$$B_m = \beta B = \frac{A_m(1)}{B(1)} B \tag{4.22}$$

results in the factorization of Equation 4.12 to be

$$B^+ = 1 \tag{4.23}$$

and

$$B^- = B. \tag{4.24}$$

The Diophantine equation becomes

$$AR + BS = A_0 A_m \tag{4.25}$$

and T is defined by

$$T = \beta A_0 = \frac{A_m(1)}{B(1)} A_0 . \tag{4.26}$$

4.3 Self-Tuning Regulator Algorithm

The algorithm for determining a self-tuning regulator, taken directly from Astrom and Wittenmark [1] is as follows.

Data: Given the specifications of the desired closed-loop pulse-transfer operator and a desired A_0.

Step 1: Estimate the polynomials of the transfer function with recursive least-squares.

Step 2: Solve for R and S in Equation 4.55 and T in Equation 4.16.

Step 3: Calculate the control variable, $u(t)$, from Equation 4.2

Repeat Steps 1, 2, and 3 at each sampling period.

4.4 Self-Tuning Regulator Design Specifications

It is assumed that the model is

$$G_p(s) = \frac{2}{s^2 + 12s + 20.02}, \tag{4.27}$$

as determined in Chapter 1, with pulse-transfer function

$$H_{pq}(q) = \frac{B}{A} = \frac{0.961\times10^{-4} q + 0.9233\times10^{-4}}{q^2 - 1.885q + 0.8869}, \tag{4.28}$$

which can be generalized to

$$H_{pq}(q)=\frac{B}{A}=\frac{b_0q+b_1}{q^2+a_1q+a_2} \tag{4.29}$$

because the parameters will vary. The desired pulse-transfer function is

$$H_{m_{pq}}(q)=\frac{B_m}{A_m}=\frac{B_m}{q^2-1.6015q+0.6703}, \tag{4.30}$$

which can be generalized to

$$H_{m_{pq}}(q)=\frac{B_m}{A_m}=\frac{b_{m0}q+b_{m1}}{q^2+a_{m1}q+a_{m2}}. \tag{4.31}$$

The numerator has been left undefined because it is convenient to define the desired numerator differently depending on zero cancellation. The pole excess for the system, d_0 is one.

4.5 Self-Tuning Regulator with Zero-Cancellation

For zero cancellation, the degree of A_0 is zero. Thus,

$$A_0=1 \tag{4.32}$$

is true because A_0 is monic. The desired pulse-transfer function, obtained by choosing B_m according to Equation 4.17, is

$$H_{m_{pq}}(q)=\frac{b_{m0}q}{q^2+a_{m1}q+a_{m2}}=\frac{0.0031q}{q^2-1.92q+0.9231}. \tag{4.33}$$

B is factored into

$$B^+=q+\frac{b_1}{b_0}=q+0.9608 \tag{4.34}$$

and

$$B^{-} = b_0 = 0.961 \times 10^{-4} . \tag{4.35}$$

Because n is two, the degrees of R, S, and T are 1. Since the degrees of R and B^{+} are both 1, according to Equation 4.14 the degree of R' must be 0. Substituting into the Diophantine equation gives

$$\left(q^2 + a_1 q + a_2\right)(1) + b_0\left(s_0 q + s_1\right) = (1)\left(q^2 + a_{m1} q + a_{m2}\right) \tag{4.36}$$

Equating like coefficients yields

$$a_1 + b_0 s_1 = a_{m1} \tag{4.37}$$

for the first degree term and

$$a_2 + b_0 s_0 = a_{m2} \tag{4.38}$$

for the constant term. However, R is defined by Equation 4.14 as

$$R = (1)\left(q + \frac{b_1}{b_0}\right) \tag{4.39}$$

The expression for S is obtained from Equations 4.37 and 4.38, i.e.

$$S = \frac{a_{m1} - a_1}{b_0} q + \frac{a_{m2} - a_2}{b_0} \tag{4.40}$$

defines S. Using Equation 4.21 gives T. The simulation of this adaptive scheme is depicted in Figures 4.2 and 4.3. The output of the plant is shown in Figure 4.2, while the control signal is shown in Figure 4.3. Figure 4.4 shows that the parameter estimates converge within ten steps. The system does meet all the requirements, however, from Figure 4.3 it is seen that there is a great initial control effort. Further, the control signal has ringing which can cause additional wear on the plant.

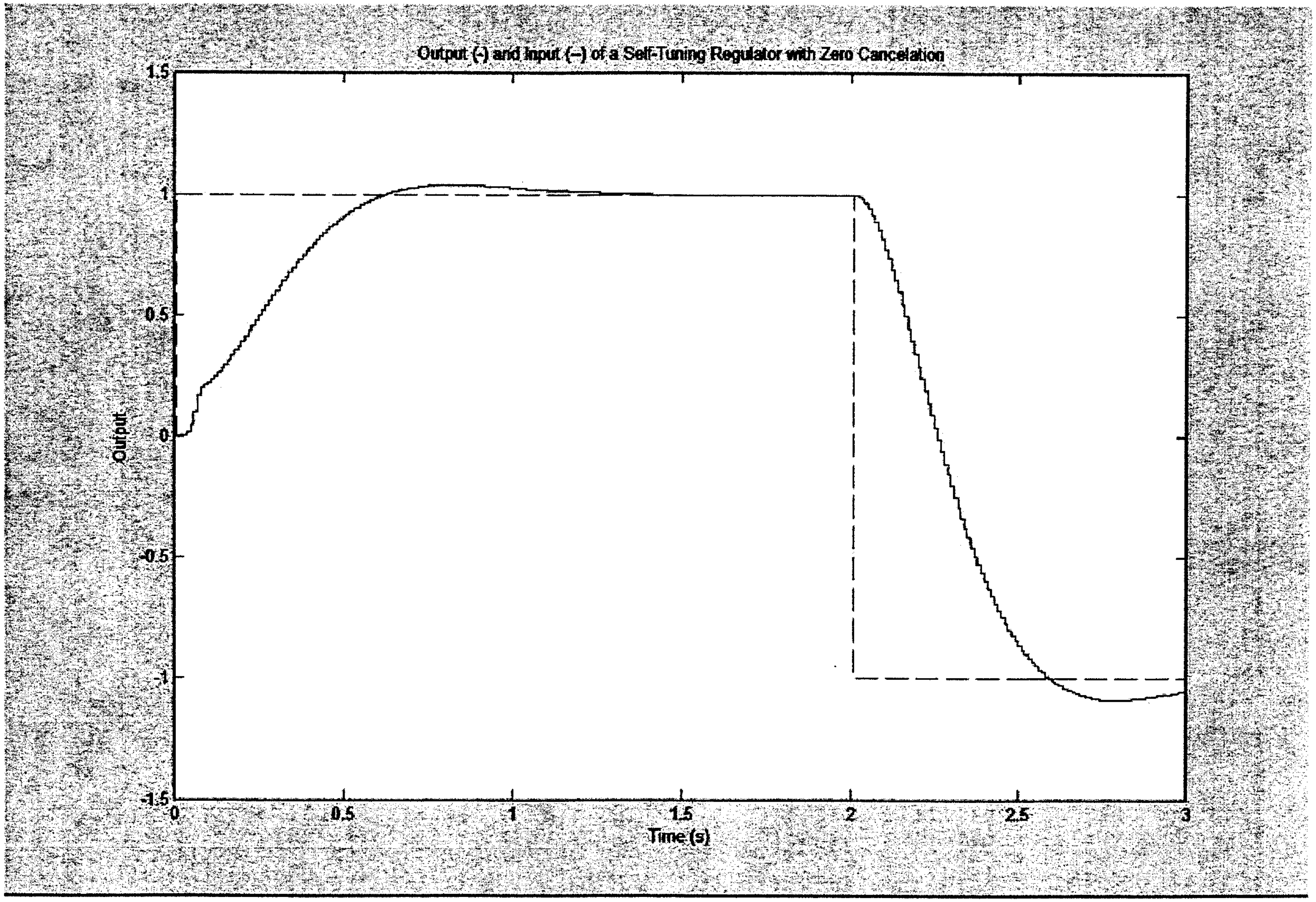

Figure 4.2: Reference and Output of a Self-Tuning Regulator-Controlled System

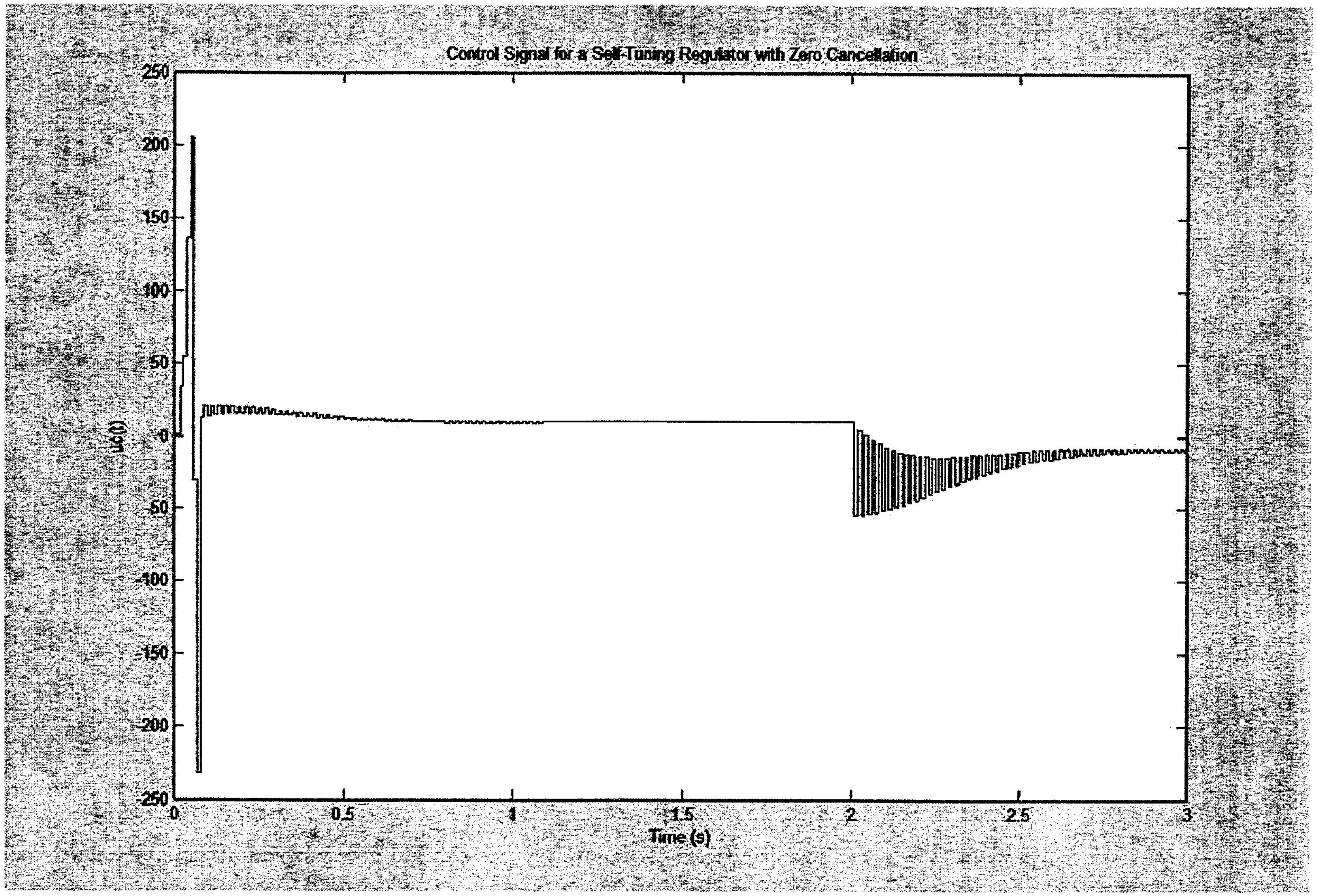

Figure 4.3: Control Signal for a Self-Tuning Regulator with Pole-Zero Cancellation.

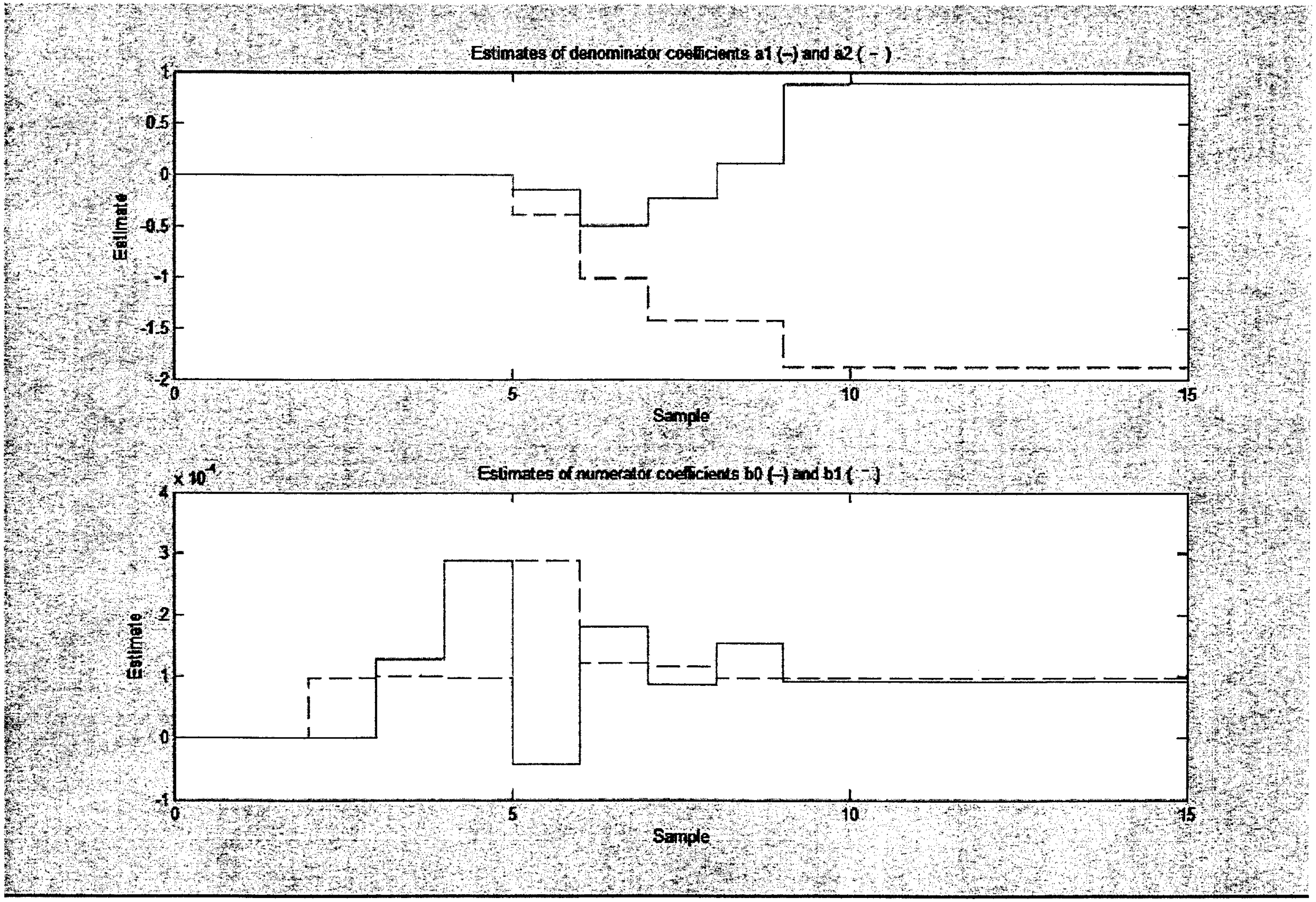

Figure 4.4: Parameter Estimates for a Self-Tuning Regulator with Pole-Zero Cancellation.

4.6 Self-Tuning Regulator without Zero-Cancellation

For a self-tuning regulator without zero-cancellation, the degree of A_0 is one. Hence, A_0 is given by

$$A_0 = q + a_{00}. \tag{4.41}$$

Selecting B_m according to Equation 4.22 results in the desired pulse-transfer function,

$$H_{m_{pq}}(q) = \frac{b_{m0}q + b_{m1}}{q^2 + a_{m1}q + a_{m2}} = \frac{\frac{0.0031}{b_0 + b_1}(b_0 q + b_1)}{q^2 - 1.92q + 0.9231}. \tag{4.41}$$

Substituting for A and B in the Diophantine equation results in

$$(q^2 + a_1 q + a_2)(q + r_1) + (b_0 q + b_1)(s_0 q + s_1) = (q^2 + a_{m1}q + a_{m2})(q + a_{00}) \tag{4.42}$$

to be solved for R and S. Collecting coefficients of q, results in

$$\begin{aligned} &q^3 + q^2(r_1 + a_1 + b_0 s_0) + q(a_1 r_1 + a_2 + b_0 s_1 + b_1 s_0) + (a_2 r_1 + b_1 s_1) \\ &= q^3 + q^2(a_{m1} + a_{00}) + q(a_{m1}a_{00} + a_{m2}) + (a_{m2}a_{00}). \end{aligned} \tag{4.43}$$

Equating like coefficients of q results in the system of equations,

$$\begin{pmatrix} 1 & b_0 & 0 \\ a_1 & b_1 & b_0 \\ a_2 & 0 & b_1 \end{pmatrix} \begin{pmatrix} r_1 \\ s_0 \\ s_1 \end{pmatrix} = \begin{pmatrix} a_{m1} - a_1 + a_{00} \\ a_{m2} - a_2 + a_{00}a_{m1} \\ a_{m2}a_{00} \end{pmatrix}. \tag{4.44}$$

Solving for the vector consisting of the R and S coefficients gives

$$\begin{pmatrix} r_1 \\ s_0 \\ s_1 \end{pmatrix} = \begin{pmatrix} 1 & b_0 & 0 \\ a_1 & b_1 & b_0 \\ a_2 & 0 & b_1 \end{pmatrix}^{-1} \begin{pmatrix} a_{m1} - a_1 + a_{00} \\ a_{m2} - a_2 + a_{00}a_{m1} \\ a_{m2}a_{00} \end{pmatrix}, \tag{4.45}$$

or

$$\begin{pmatrix} r_1 \\ s_0 \\ s_1 \end{pmatrix} = \frac{1}{a_2 b_0^2 + b_1^2 - a_1 b_0 b_1} \begin{pmatrix} b_1^2 & -b_0 b_1 & b_0^2 \\ a_2 b_0 - a_1 b_1 & b_1 & -b_0 \\ -a_2 b_1 & a_2 b_0 & b_1 - a_1 b_0 \end{pmatrix} \begin{pmatrix} a_{m1} - a_1 + a_{00} \\ a_{m2} - a_2 + a_{00} a_{m1} \\ a_{m2} a_{00} \end{pmatrix} \quad (4.46)$$

The solutions for r_1, s_0, and s_1 are

$$r_1 = \frac{b_0^2 a_{m2} a_{00} - b_0 b_1 (a_{m2} - a_2 + a_{m1} a_{00}) + b_1^2 (a_{m1} - a_1 + a_{00})}{a_2 b_0^2 + b_1^2 - a_1 b_0 b_1}, \quad (4.47)$$

$$s_0 = \frac{b_0 (a_{00} a_2 + a_2 a_{m1} - a_1 a_2 - a_{00} a_{m2}) + b_1 (a_1^2 + a_{m2} + a_{00} a_{m1} - a_{00} a_1 - a_{m1} a_1 - a_2)}{a_2 b_0^2 + b_1^2 - a_1 b_0 b_1} \quad (4.48)$$

and

$$s_1 = \frac{b_0 (a_{00} a_2 a_{m1} + a_2 a_{m2} - a_{00} a_1 a_{m2} - a_2^2) + b_1 (a_1 a_2 + a_{00} a_{m2} - a_{00} a_2 - a_2 a_{m1})}{a_2 b_0^2 + b_1^2 - a_1 b_0 b_1}. \quad (4.49)$$

T is defined by Equation 4.26 to be

$$T = \frac{1 + a_{m1} + a_{m2}}{b_0 + b_1} A_0 = \frac{0.0031}{b_0 + b_1} (q + a_{00}). \quad (4.50)$$

The simulated system output is shown in Figure 4.5. It can be observed that the system's output settles to its final values with zero error in about one second. The overshoot of the system is 4%. Figure 4.6 shows the control signal, which has no ringing and a lesser control effort than is the case for the self-tuning regulator with zero cancellation. Figure 4.7 shows the convergence of the parameter estimates, which once again converge within 10 samples.

4.7 Conclusions

Both self-tuning regulators, with and without zero cancellations, meet the requirements of the design. However, due to the lesser control effort and the lack of ringing, the self-tuning regulator without zero cancellation can be considered to be a

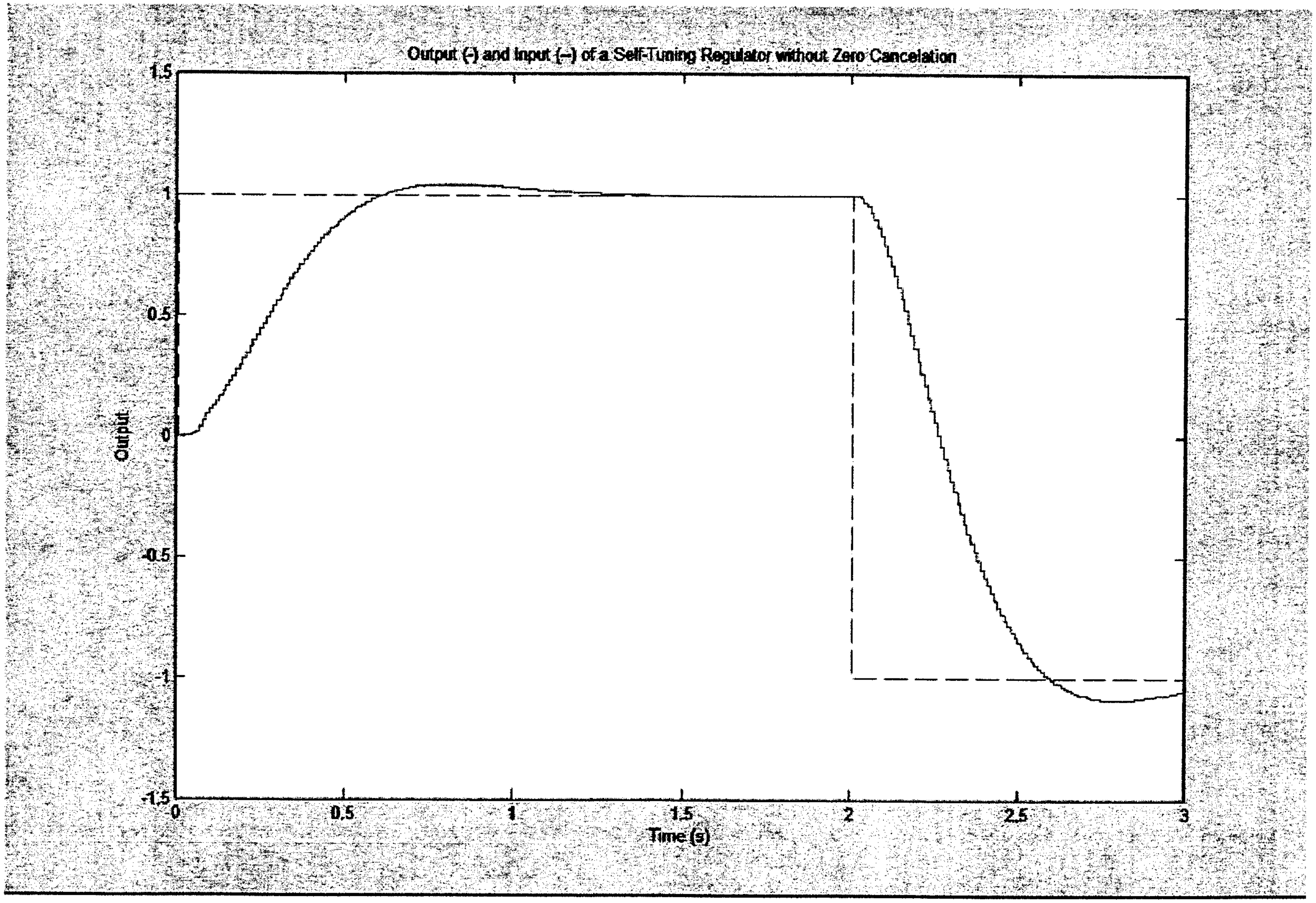

Figure 4.5: Reference and Output of a Self-Tuning Regulator-Controlled System without Pole-Zero Cancellation.

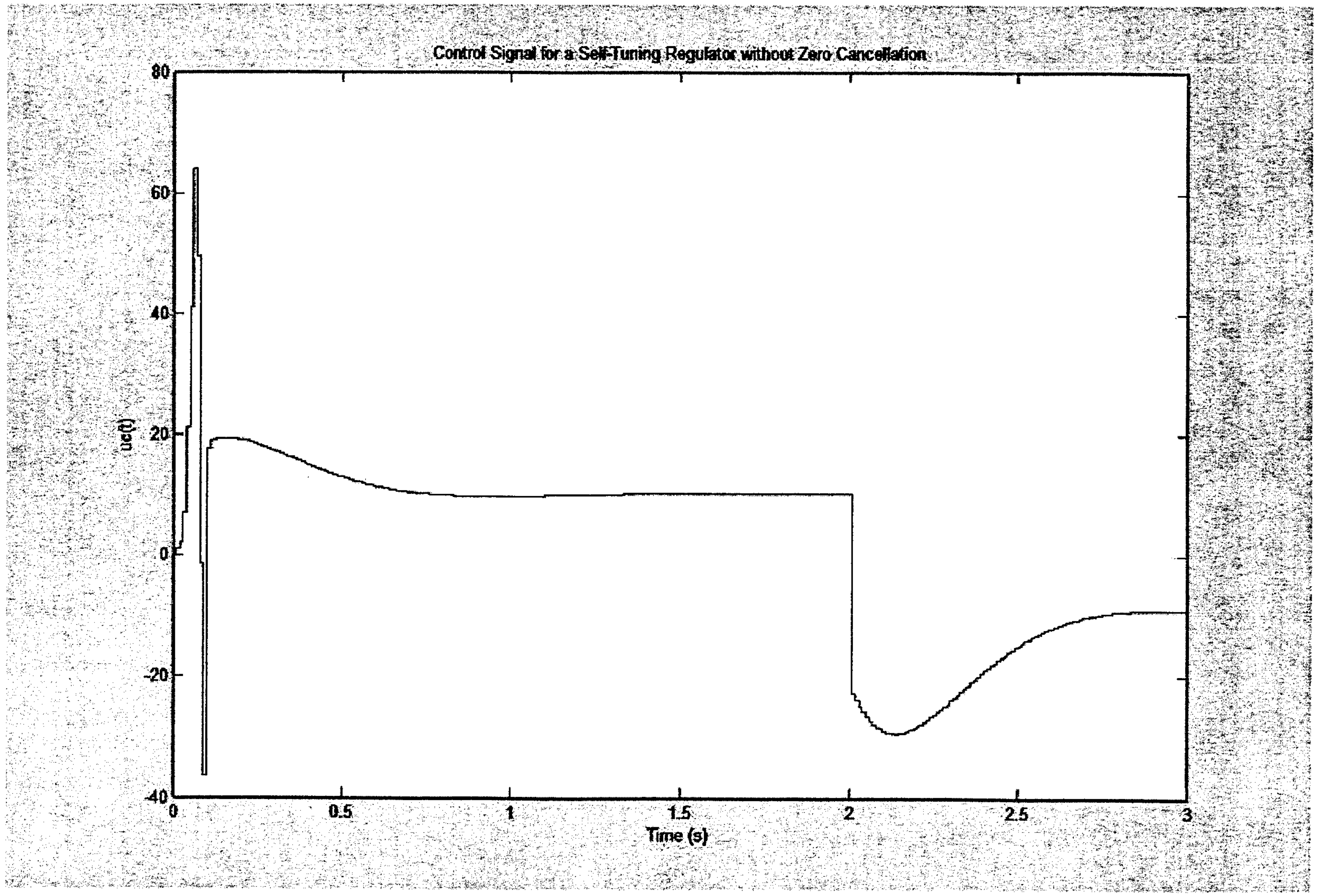

Figure 4.6: Control Signal for a Self-Tuning Regulator without Pole-Zero Cancellation.

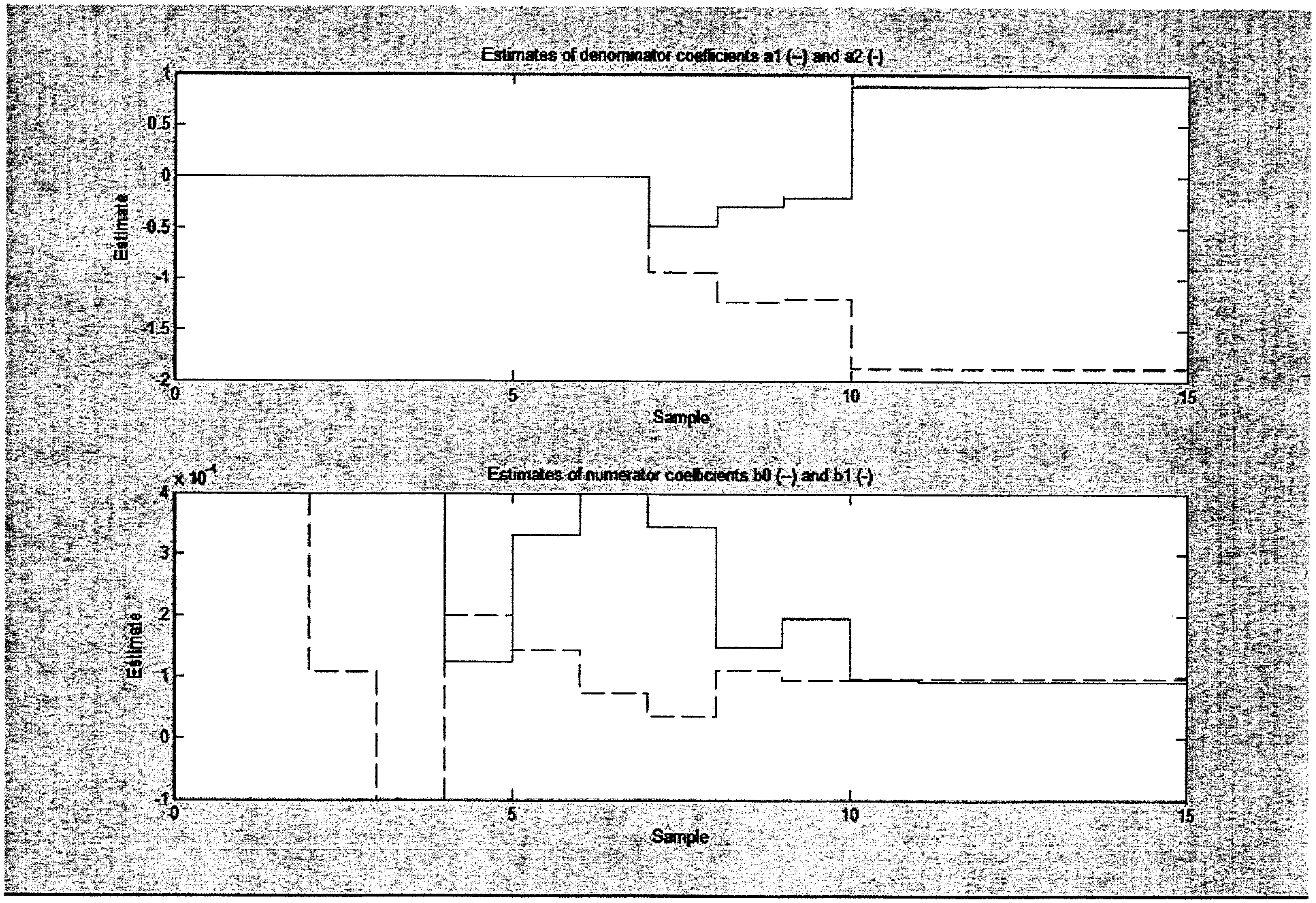

Figure 4.7: Parameter Estimates for a Self-Tuning Regulator without Pole-Zero Cancellation.

better choice. This control technique will be used in the discussion on disturbances and varying loads on the motor , which is provided in Chapter 6.

CHAPTER V

ONE-STEP AHEAD AND MODEL-REFERENCE ADAPTIVE CONTROLLERS

This chapter covers two related types of adaptive controller, the one-step ahead controller and the model-reference adaptive controller. First the one-step ahead controller is used to attempt to bring the output of the closed-loop system to the desired value in one step. This in general requires a great control effort, so a form of weighting on the control signal is also considered to reduce the control effort. A further step to reduce the control effort is taken. Rather than attempting to bring the output to the desired output in a single step, the model-reference adaptive technique endeavors to bring the output of the system to that of the desired model. Several controllers with different weightings for model-reference adaptive controllers are designed and compared in this chapter.

5.1 Introduction to One-Step Ahead Control

The one-step ahead (OSA) controller is another adaptive control technique [7]. The one-step ahead controller brings the predicted output $y(k+d)$ to a desired value, $y^{*}(k+d)$ in a single step. As will be discovered, the control effort for such a controller is often great, so the controller can be weighted in such a way as to reduce the control effort.

5.2 The One-Step Ahead and Weighted One-Step Ahead Controllers

Once again, the model is assumed to be

$$A_p^*(q^{-1})y(k) = q^{-d}B_p^*(q^{-1})u(k), \tag{5.1}$$

where

$$A_p^*(q^{-1}) = 1 + a_1 q^{-1} + \cdots + a_n q^{-n} \tag{5.2}$$

and

$$B_p^*(q^{-1}) = q^{-d}(b_0 + b_1 q^{-1} + \cdots + b_m q^{-m}) = q^{-d}B'(q^{-1}). \tag{5.3}$$

as described in Equation 1.37. This model can be rewritten in a d-step-ahead predictor form,

$$y(k+d) = \alpha(q^{-1})y(k) + \beta(q^{-1})u(k), \tag{5.4}$$

where

$$\alpha(q^{-1}) = \alpha_0 + \alpha_1 q^{-1} + \cdots + \alpha_{n-1} q^{-n+1} = G(q^{-1}) \tag{5.5}$$

and

$$\beta(q^{-1}) = \beta_0 + \beta_1 q^{-1} + \cdots + \beta_{m+d-1} q^{-m-d+1} = F(q^{-1})B'(q^{-1}), \tag{5.6}$$

with the coefficients of $F(q^{-1})$ and $G(q^{-1})$ given by

$$f_0 = 1, \tag{5.7}$$

$$f_i = -\sum_{j=0}^{i-1} f_j a_{i-j}, \tag{5.8}$$

and

$$g_i = -\sum_{j=0}^{d-1} f_j a_{i+d-j} \,. \tag{5.9}$$

The time delay, d, is selected in such a manner as to ensure that b_0 in Equation 5.3 is nonzero, which assures that β_0 in Equation 5.6 is nonzero as well. In order to bring $y(k+d)$ to the desired value of $y^*(k+d)$, while minimizing the cost function,

$$J(k+d) = \frac{1}{2}\varepsilon^2(k+d) = \frac{1}{2}\left(y(k+d) - y^*(k+d)\right)^2, \tag{5.10}$$

the desired value is substituted into Equation 5.4, resulting in

$$y^*(k+d) = \alpha(q^{-1})y(k) + \beta(q^{-1})u(k). \tag{5.11}$$

This can be expanded into

$$y^*(k+d) = \alpha(q^{-1})y(k) + \beta_0 u(k) + \beta_1 u(k-1) + \cdots + \beta_{m+d-1}u(k-m-d+1), \tag{5.12}$$

which can be solved for $u(k)$

$$u(k) = \frac{y^*(k+d) - \alpha(q^{-1})y(k) - \beta_1 u(k-1) - \cdots - \beta_{m+d-1}u(k-m-d+1)}{\beta_0}. \tag{5.13}$$

Because the selection of b_0 always results in a nonzero β_0, the construction is possible [6].

For the DC-motor plant as given in Equation 1.45

$$H(q^{-1}) = \frac{q^{-d}(b_0 + b_1 q^{-1})}{1 + a_1 q^{-1} + a_2 q^{-2}} = \frac{q^{-1}(0.0367 + 0.0321q^{-1})}{1 - 1.6015q^{-1} + 0.6703q^{-2}}, \tag{5.14}$$

the order of the numerator, m, is one, the order of the denominator, n , is two, and the pole excess, d, is one. This means that $F((q^{-1})$ is zeroeth order, and $G(q^{-1})$ is first order.

Because $F\left(q^{-1}\right)$ is zeroeth order, the only coefficient present is f_0, which from Equation 5.7 is one. The coefficients of $G\left(q^{-1}\right)$ are given by Equation 5.9, which evaluates to

$$g_0 = -f_0 a_1 = -a_1 \tag{5.15}$$

and

$$g_1 = -f_0 a_2 = -a_2 . \tag{5.16}$$

The predictor is described by

$$y(k+d) = \left(-a_1 - a_2 q^{-1}\right) y(k) + \left(b_0 + b_1 q^{-1}\right) u(k). \tag{5.17}$$

The control signal, $u(k)$, is given by

$$u(k) = \frac{y^*(k+d) + a_1 y(k) + a_2 y(k-1) - b_1 u(k-1)}{b_0}. \tag{5.18}$$

For adaptive one-step ahead control, the parameters are unknown and must be estimated, in this case with a recursive least squares estimation algorithm. Figure 5.1 shows the response for this closed-loop system, and Figure 5.2 shows the control signal for the DC motor using one-step ahead control.

A simple modification to Equation 5.12 results in the weighted one-step ahead controller control signal [7], given by

$$u(k) = \beta_0 \left(\frac{y^*(k+d) - \alpha\left(q^{-1}\right) y(k) - \beta_1 u(k-1) - \cdots - \beta_{m+d-1} u(k-m-d+1)}{\beta_0^2 + \lambda} \right). \tag{5.19}$$

This is a generalized case of one-step ahead control, which minimizes the cost function

$$J(k+d) = \frac{1}{2}\left(y(k+d) - y^*(k+d)\right)^2 + \frac{\lambda}{2} u^2(t). \tag{5.20}$$

When λ is set to zero, the control signal obtained by one-step ahead adaptive control is recovered. For the DC motor, the control signal becomes

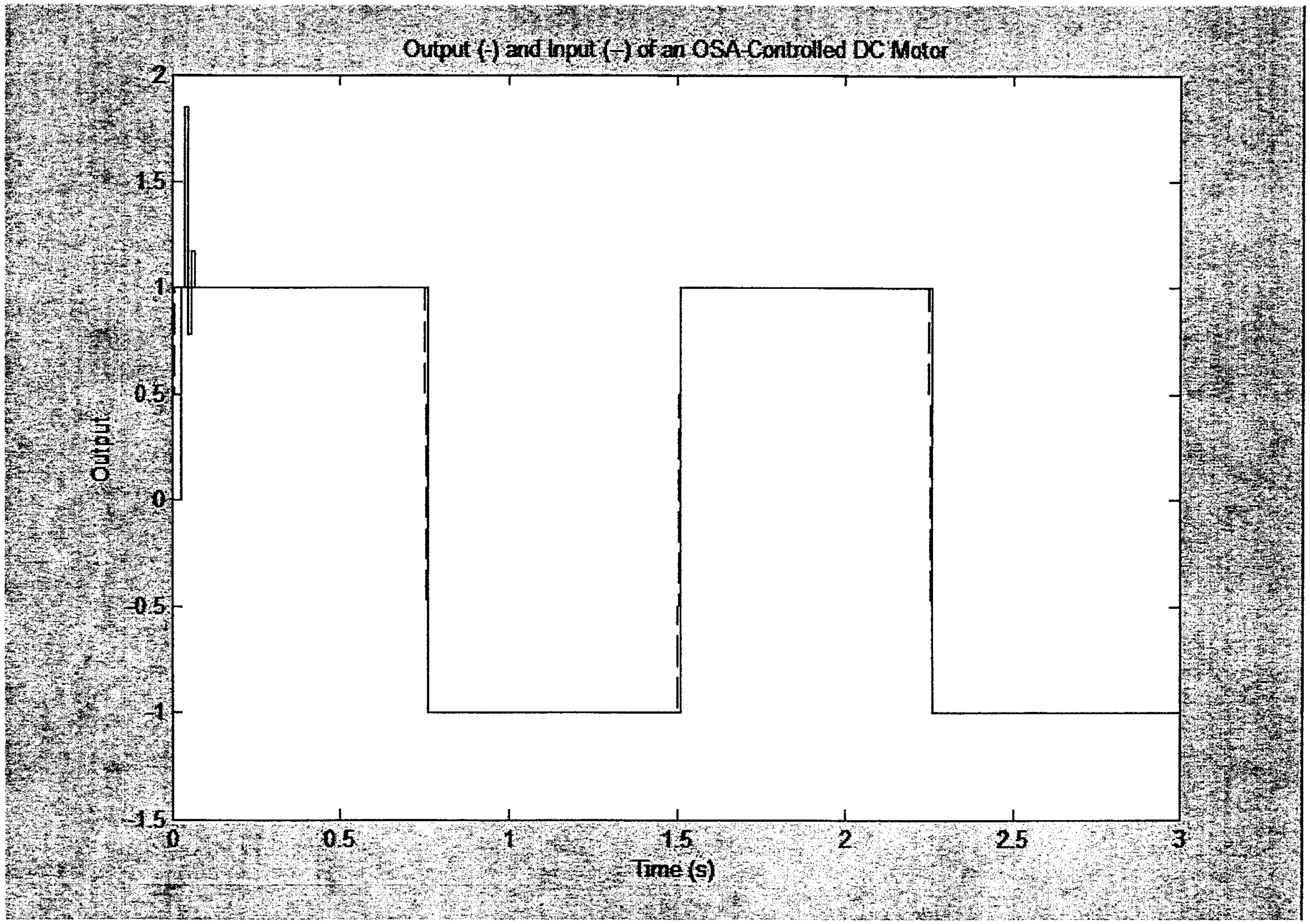

Figure 5.1: System Response of a One-Step Ahead Adaptive Controller.

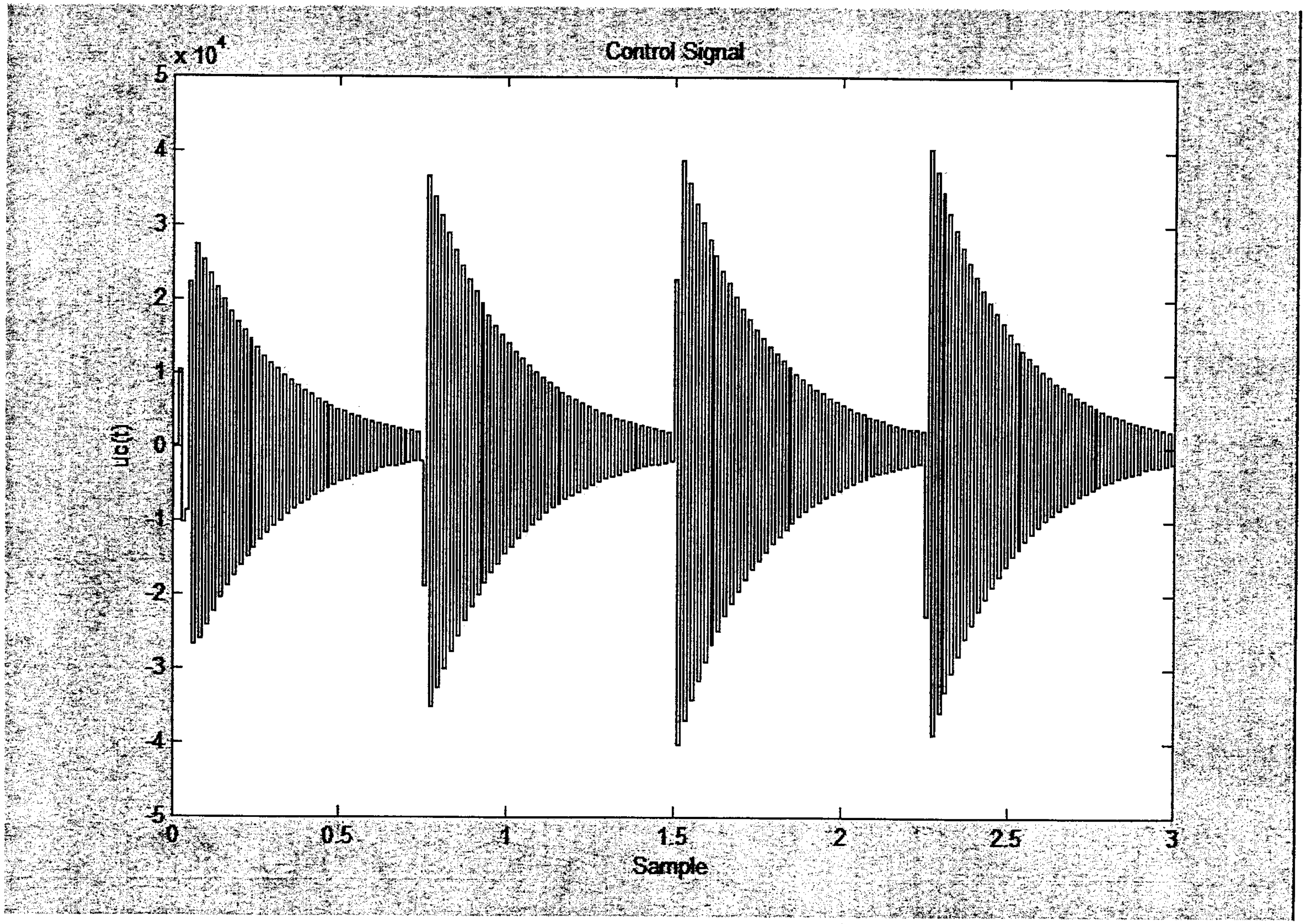

Figure 5.2: Control Signal for a One-Step Ahead Adaptive Controller.

$$u(k) = b_0 \left(\frac{y^*(k+d) + a_1 y(k) + a_2 y(k-1) - b_1 u(k-1)}{b_0^2 + \lambda} \right), \tag{5.21}$$

where a_1, a_2, b_0, and b_1 are estimated recursively. Because b_0 is of the magnitude of 10^{-3}, λ must be less than 10×10^{-6} lest λ dominate the denominator. The closed-loop responses of the system for several different values of λ and the corresponding control signals are depicted in Figures 5.3 through 5.6. Figure 5.3 shows the results when λ is 10×10^{-6}. This system does reach its final value rather quickly, however, the final value is 0.4879, which gives an unacceptable steady-state error. The control signal for this system has a maximum magnitude of 15, which will be the smallest control signal. One additional benefit is the elimination of the ringing in the control signal. Figure 5.4 shows the system information when λ is 10^{-6}. This system has a final value of 0.905, which is better than the first system. This is countered by the fact that this system has a maximal magnitude of 185 for the control signal. Once again there is no ringing in the control signal. Figure 5.5 shows the system when λ is 500×10^{-9}. The final value for this system is 0.9506 with a maximal magnitude of 370 for the control signal. The overshoot for this system is 25%. The final simulation is in Figure 5.6 when λ is 100×10^{-9}. The final value of this simulation is 1, however, the cost for the perfect steady state is high as the maximum magnitude of the control signal approaches $1{,}800$, with an overshoot of about 45%. Based on comparing these four simulations, either a λ of 500×10^{-9} or a λ of 10^{-6} is the best controller for this plant, depending on how important the steady state accuracy is.

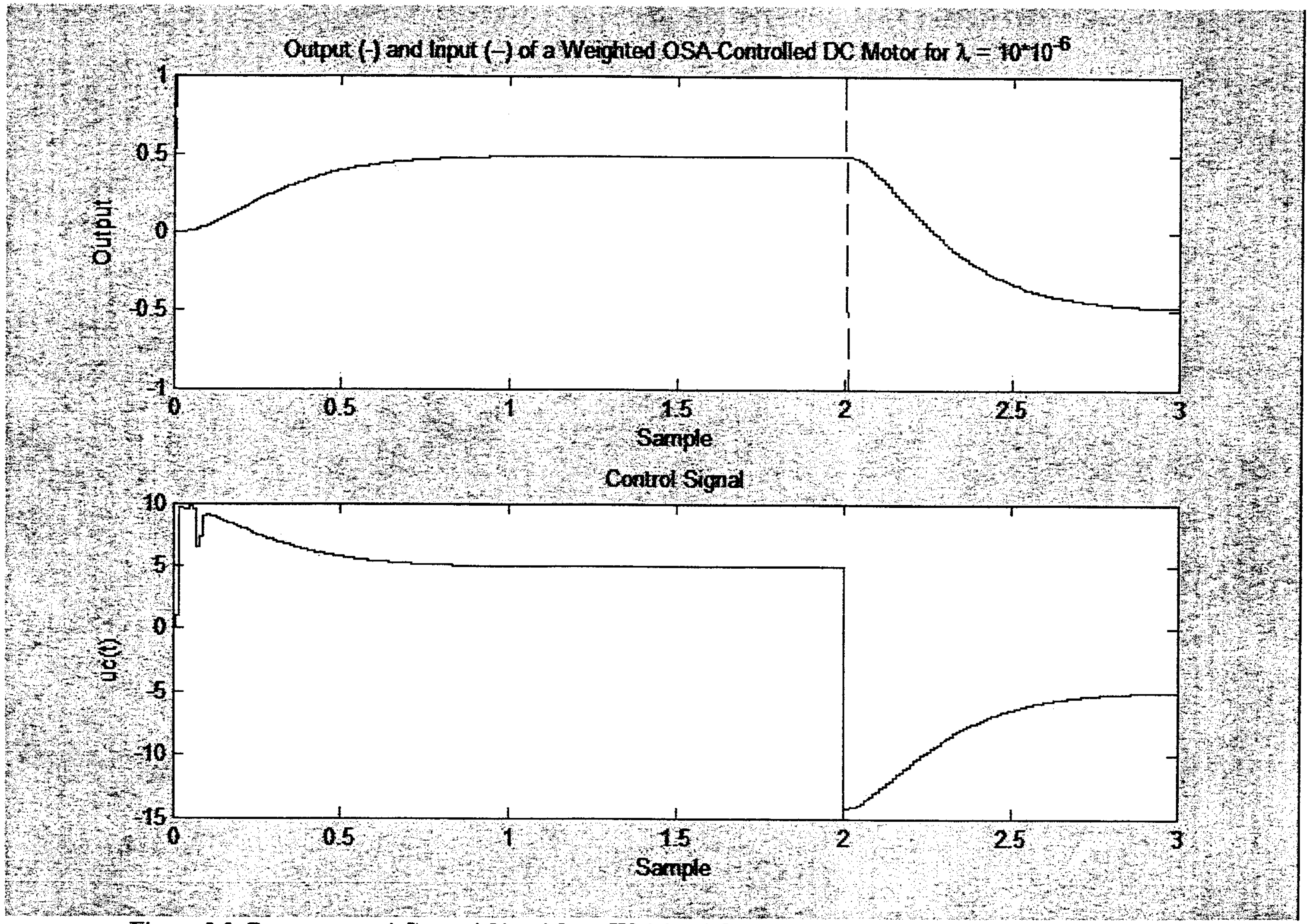

Figure 5.3: Response and Control Signal for a Weighted One-Step Ahead Adaptive Controlled System.

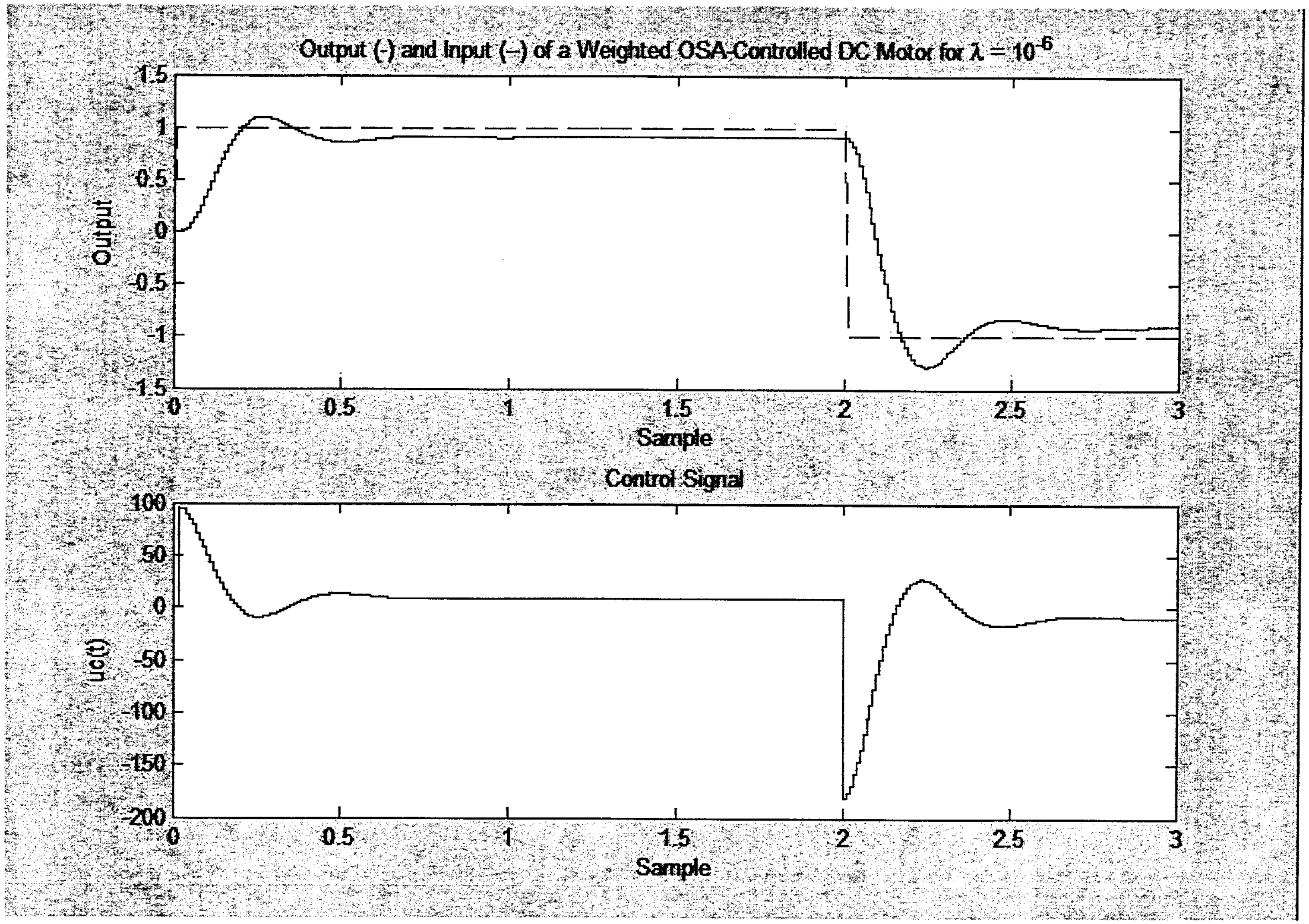

Figure 5.4: Response and Control Signal for a Weighted One-Step Ahead Adaptive Controlled System.

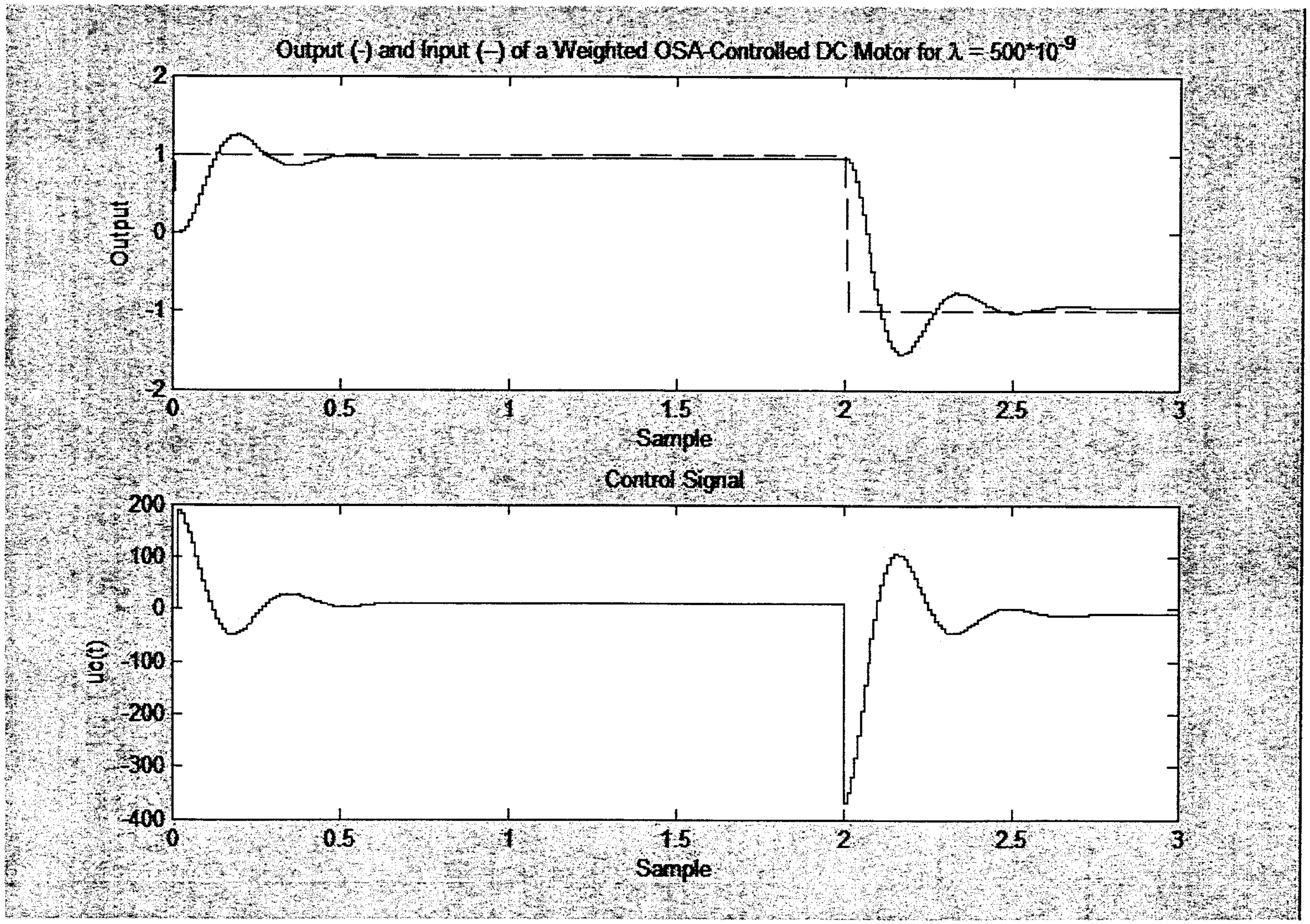

Figure 5.5: Response and Control Signal for a Weighted One-Step Ahead Adaptive Controlled System.

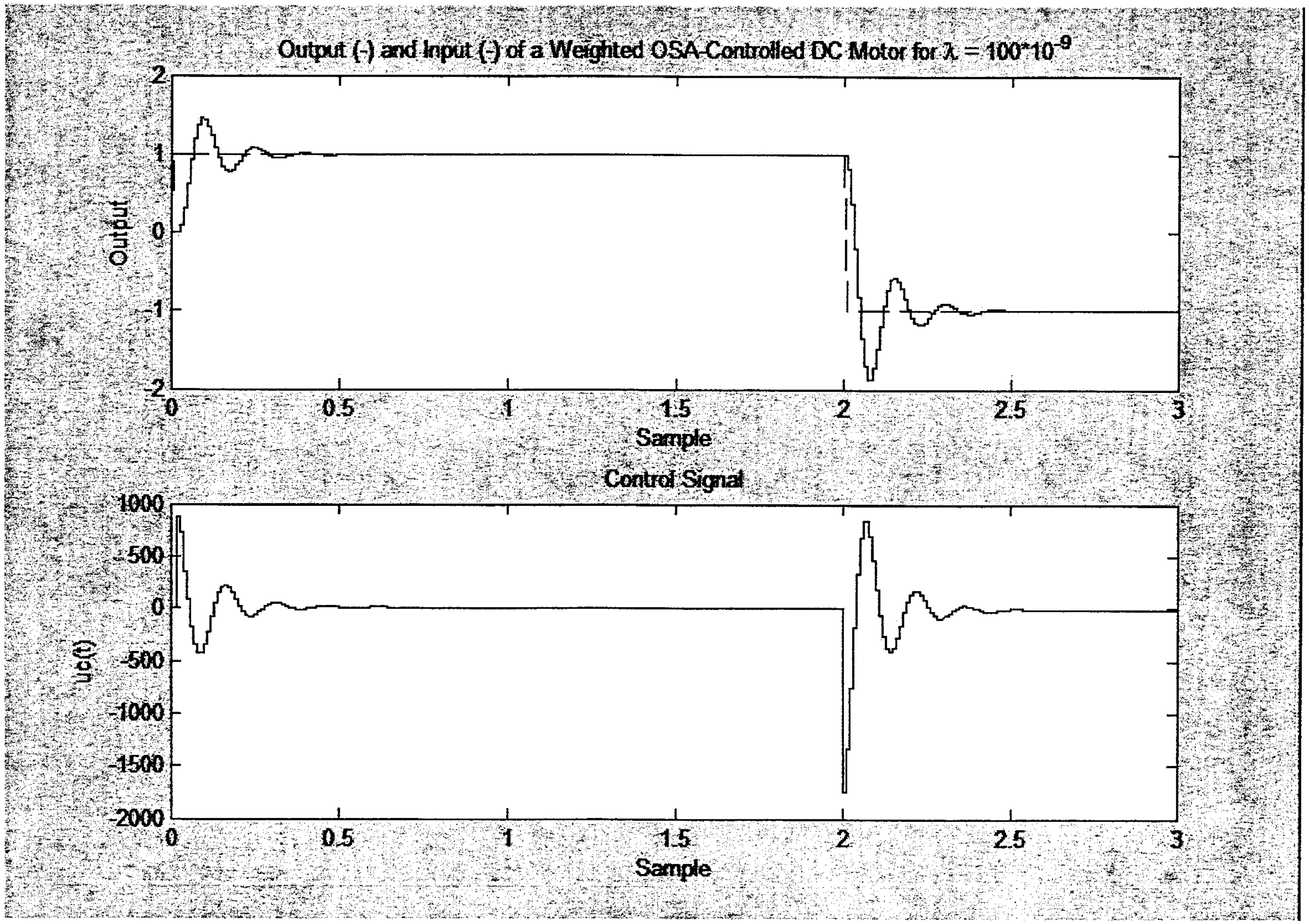

Figure 5.6: Response and Control Signal for a Weighted One-Step Ahead Adaptive Controlled System.

5.3 Model-Reference Adaptive System

Rather than trying to follow the reference signal, following a model that has an acceptable transfer function will serve to lessen the control effort. This is the so-called model-reference adaptive control, which is somewhat related to the self-tuning regulator [1],[7]. The model that will be used for reference is given by Equation 1.45,

$$H_{m_{pq}}(z) = \frac{0.001638z + 0.001594}{z^2 - 1.92z + 0.9231}. \tag{5.22}$$

The mechanism that was used to implement the model-reference adaptive control is one-step ahead and weighted one-step ahead control. Figure 5.7 shows the simulation for model-reference one-step ahead control. It can be observed that the steady-state error is 0, with the overshoot of 15%. The maximum magnitude of he control signal is about 700, and ringing is present in the control signal. Once again, the control effort can be improved by using a weighted one-step ahead approach. The system simulation for a λ of 10×10^{-6} is shown in Figure 5.8. The final value of this system output is 0.4873, that is the steady state error is large and un acceptable. The magnitude of the maximal control input is 12.5. A lesser weighting of λ, 10^{-6} is used for the simulation shown in Figure 5.9. This simulation shows an overshoot of 10% and a steady-state error of 9.2%. The control effort has a maximum magnitude of 60. Weighting λ even less, to 500×10^{-9} results in the system shown in Figure 5.10. The steady-state error is only 4.56% and the overshoot is 11%. The control effort is relatively small with a maximal magnitude of 72. Figure 5.11 shows the final simulation, for the system with λ equal to 100×10^{-9}. This system has no steady-state error and an overshoot of 15%. The maximum control

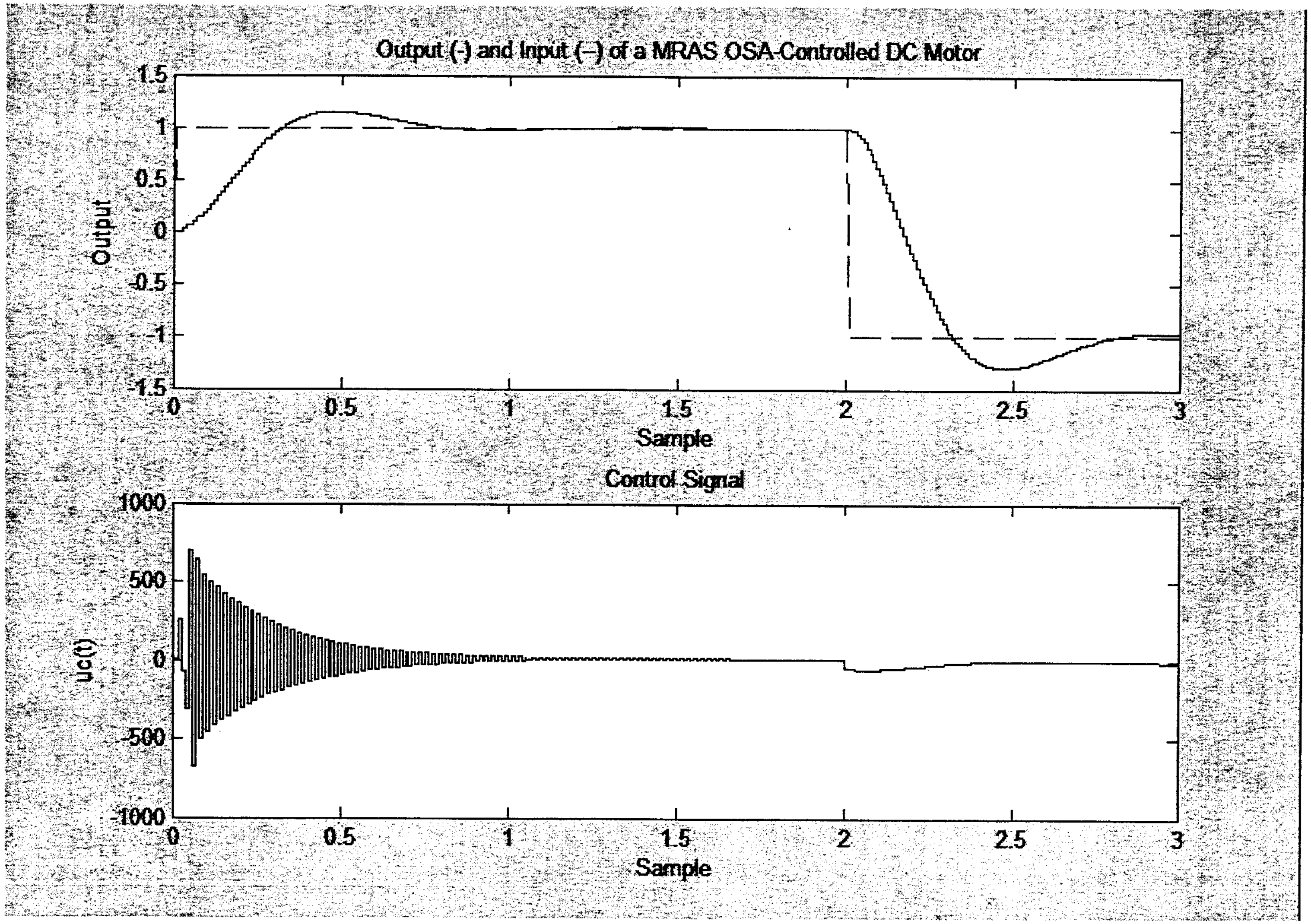

Figure 5.7: Response and Control Signal for a Model-Reference Adaptive System.

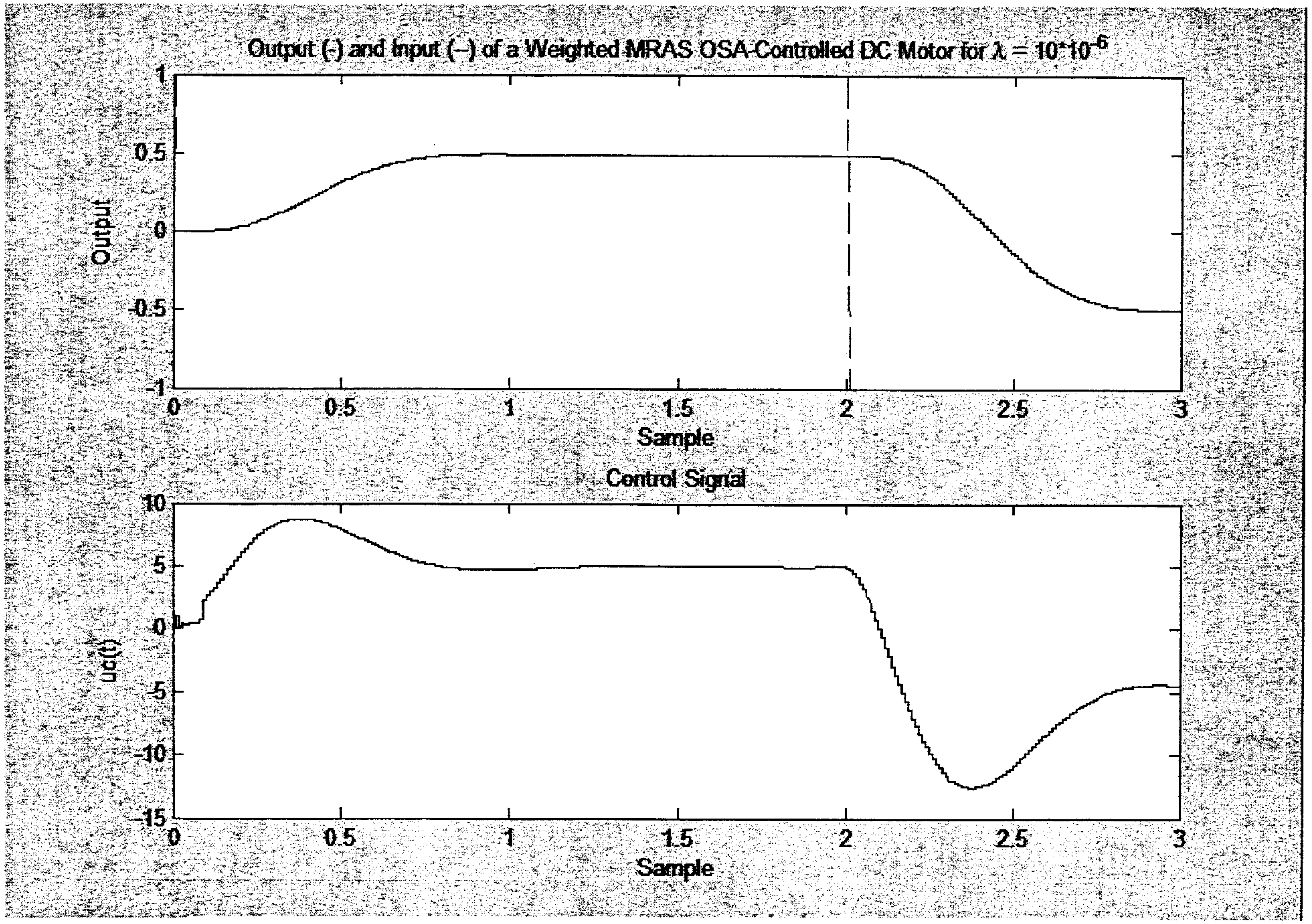

Figure 5.8: Response and Control Signal for a Weighted Model-Reference Adaptive System.

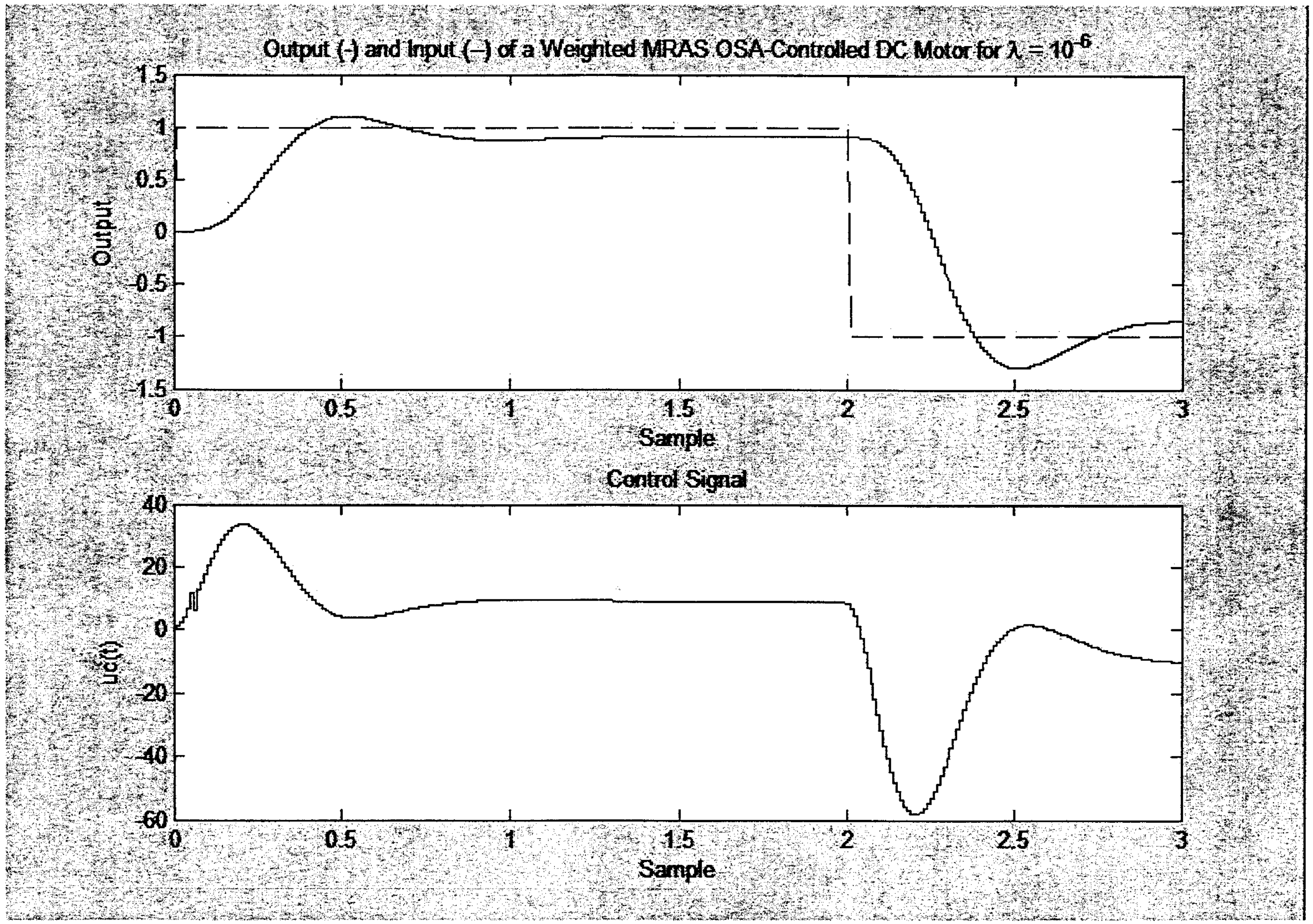

Figure 5.9: Response and Control Signal for a Weighted Model-Reference Adaptive System.

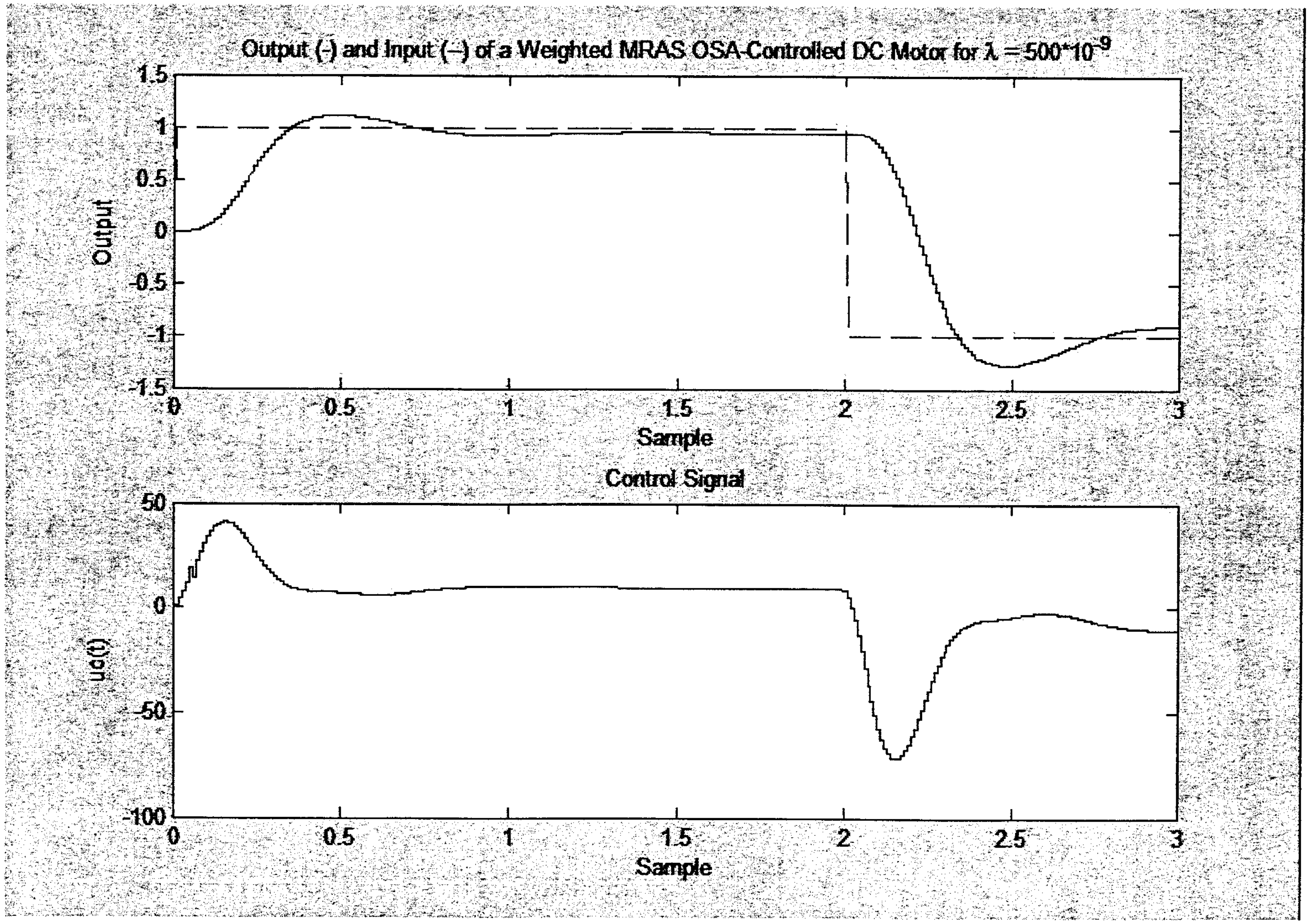

Figure 5.10: Response and Control Signal for a Weighted Model-Reference Adaptive System.

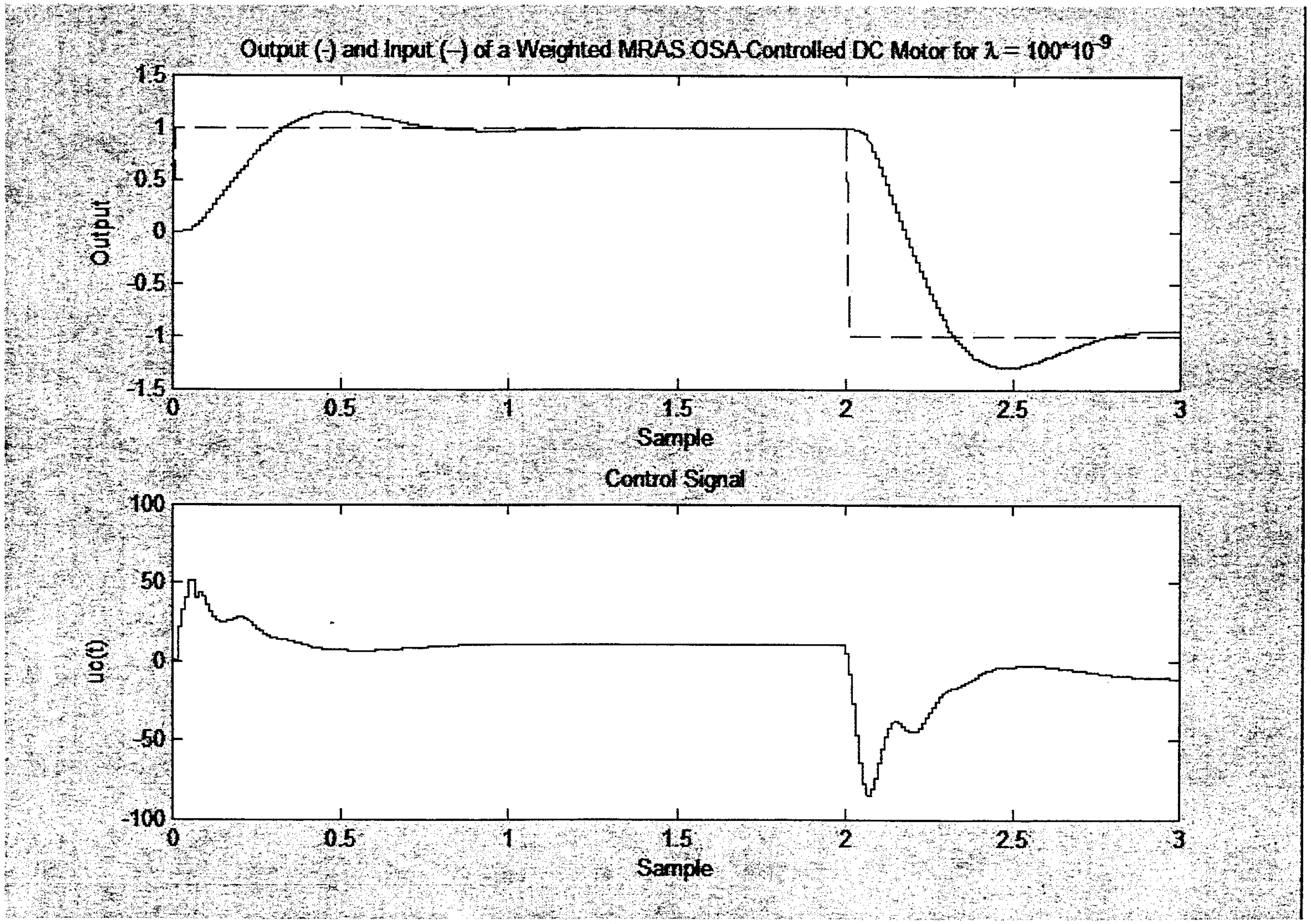

Figure 5.11: Response and Control Signal for a Weighted Model-Reference Adaptive System.

effort is 85. Each of these closed-loop systems has a settling time of less than $1s$. This system has similar performance compared to the unweighted model-reference adaptive system, but at a greatly reduced control effort.

5.4 Conclusions

Several adaptive control schemes were considered in this chapter. The model-reference adaptive controller based on weighted one-step ahead adaptive contol provided the best response. By using a weighting of 100×10^{-9}, all of the design criteria, overshoot of 15%, no steady-state error, and a settling time of $1s$, were met with out an extremely unreasonable control effort. This controller will be used as the basis for the discussion in Chapter 6, which deals with the effect of varying the load as well as noise on the closed-loop system.

CHAPTER VI

SYSTEM AND LOAD ERROR RESULTS

In this chapter the effect of model uncertainty and noise on the behavior of the closed-loop controlled systems are investigated. First the disturbance rejection properties of the controlled systems designed based on an ideal model are considered. The effect of model uncertainty is then investigated by imposing errors in the type, order, and the stability of the system. Finally, the classical and adaptive techniques are compared when the motor has a nonzero load.

6.1 Comparison of Techniques

The closed-loop response of the classical controller, self-tuning regulator, and model-reference adaptive system derived and shown in Chapter 3, Chapter 4, and Chapter 5, respectively, are shown in Figure 6.1. The plant has the assumed transfer function,

$$G_p(s) = \frac{2}{s^2 + 12s + 20.02} \tag{6.1}$$

and assumed pulse transfer function

$$H_{pq}(z) = \frac{96.10 \times 10^{-6} z + 92.3310^{-6}}{z^2 - 1.885z + 0.8869} \tag{6.2}$$

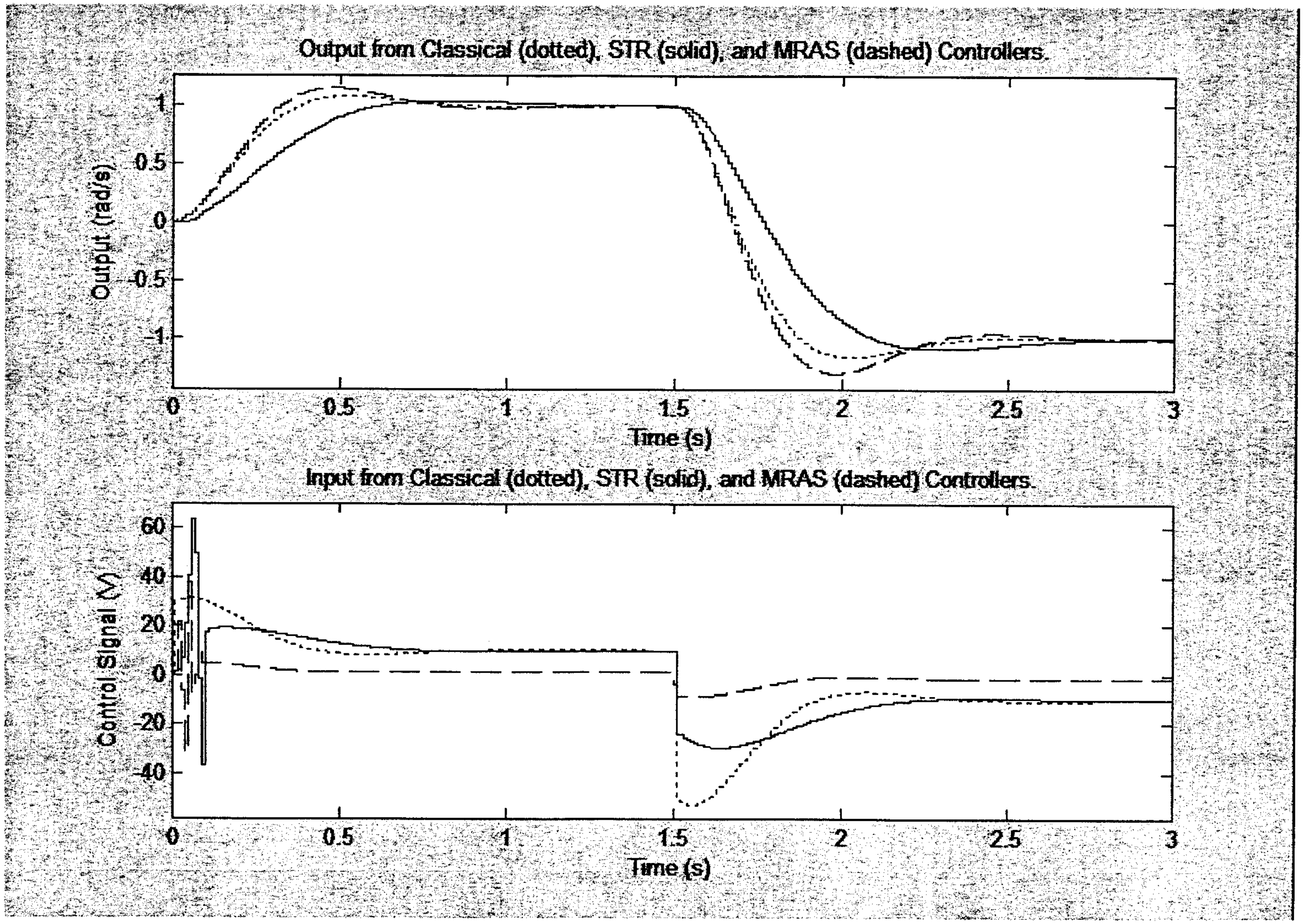

Figure 6.1: Comparison of Responses and Control Signals with Ideal Models.

From Figure 6.1, it is apparent that all three controllers meet the design criteria of a maximum overshoot of 15% and a settling time of $1s$, with the sole exception that the self-tuning regulator has a settling time of about $1.04s$, which is close enough. The self-tuning regulator has the best performance in terms of percent overshoot, having only 4.21% overshoot. The model-reference adaptive controller has the least desirable overshoot, just meeting the overshot requirement. The classical controller has an overshoot of 8.04%. Figure 6.1 also shows the control effort for each of the three controllers. Both the self-tuning regulator and the model-reference adaptive controller have higher control signals as the parameters are being estimated. However, in the steady state, the model-reference adaptive controller has the best control signal.

6.2 Disturbance Rejection

The first non-ideal situation that will be explored is the case where the input signal has noise coupled to it. For this purpose, a random signal was created in MATLAB with the RAND command. The amplitude of the noise was 10% of that of the input. Figure 6.2 shows the results of that simulation as well as the noise that was added to the input. The disturbance rejection of all three of the controllers is adequate for this magnitude of noise. It is intuitive that the classical controller will have a decent amount of disturbance rejection due to the presence of the pole at $z = 1$, which provides integrating action. The overshoot on the classical controlled system remains about 8%, the overshoot on the model-reference adaptive system remains at about 15%, while the self-tuning regulator is reduced to about 3%. The settling time on each system is about $1s$. The control effort remains largely similar to that of the noise free case simulation.

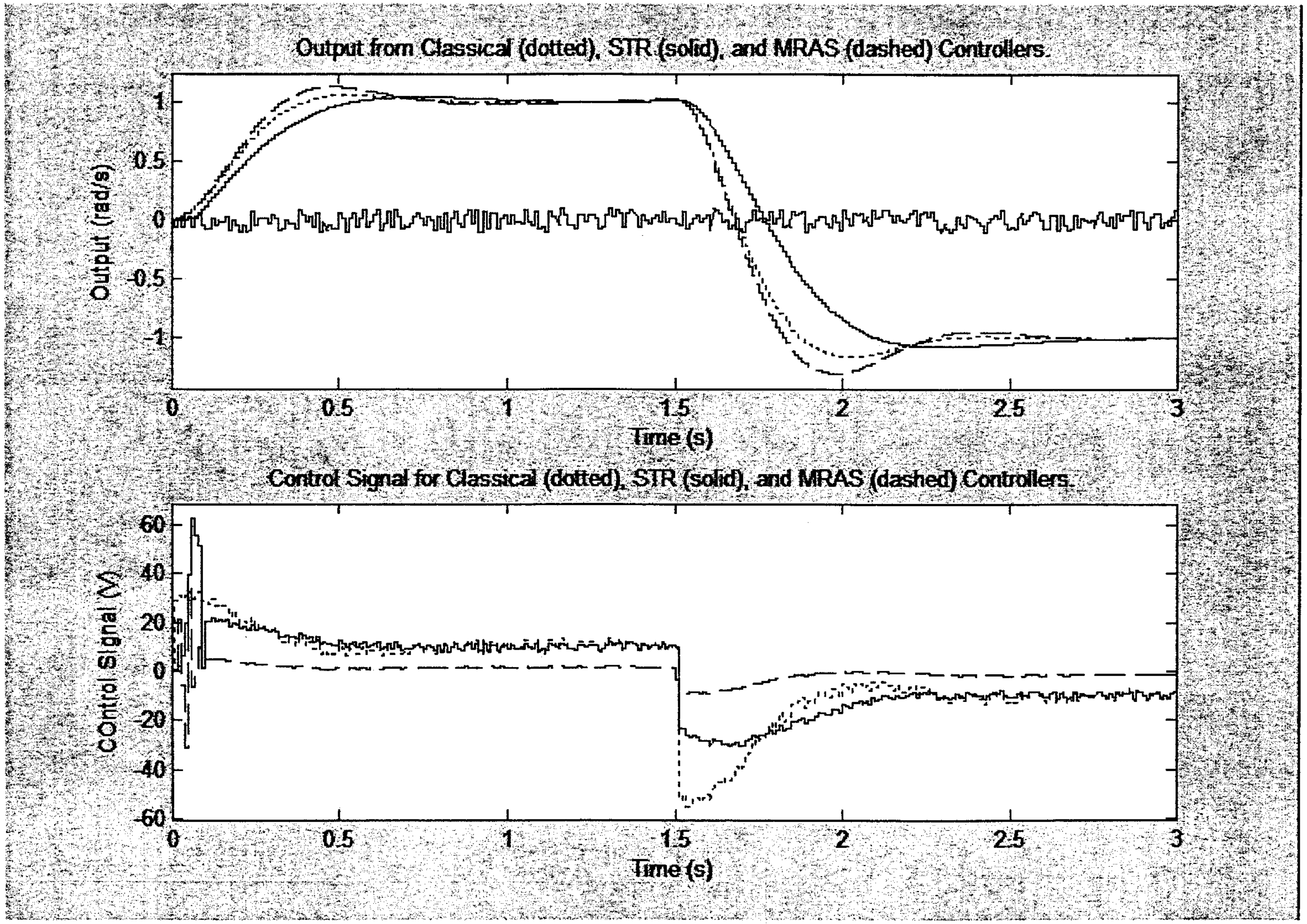

Figure 6.2: Comparison of Disturbance Rejection.

6.3 Non-Ideal Model Situation

This situation encompasses a number of circumstances, the three of interest for this discussion is an error in the type of the system, in the stability of the system, and in the order of system. First, suppose that there is an error in the type of the system, that is, the plant to be controlled is a type-1 system while the controllers were designed based on a type-0 system. Another popular model for a DC motor is

$$G_p(s) = \frac{b}{s(s+a)}, \tag{6.3}$$

which is a Type-I system. Suppose that the pole at -2.0025 remains, but that there is a zero at -200, and the dc gain is 0.0049; the resulting transfer function would be

$$G_p(s) = 0.049 \frac{s+200}{s(s+2.0025)}, \tag{6.4}$$

and the pulse-transfer function would be

$$H_{pq}(z) = \frac{96.92 \times 10^{-6} z}{z^2 - 1.9802z + 0.9802}. \tag{6.5}$$

Typically, the zero at $s = -200$ could be ignored because it has a magnitude of 100 times that of the next pole or zero. However, it is mathematically convenient to include it in this discussion because the zero in the continuous-time transfer function increases the magnitude of the pulse-transfer function. Figure 6.3 shows the simulation of the system with this plant. Because the closed-loop system using the classical controller has a double pole at 1, it is

expected that the output will oscillate and not settle down to a steady state. The classically-controlled system exhibits marginally stable behavior rather than explicitly unstable behavior because the pole at 0.9802 is still cancelled by the controller. However, if

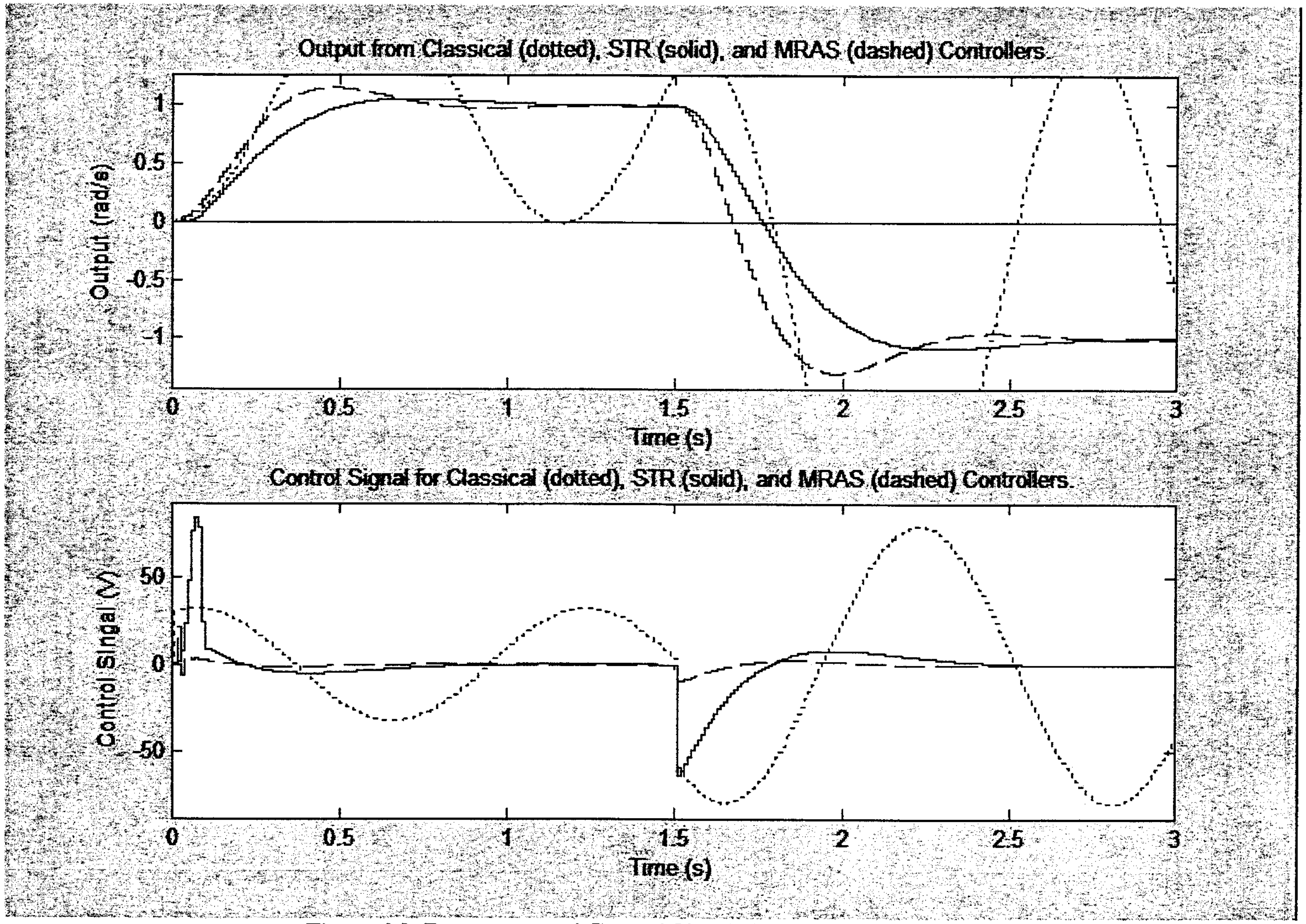

Figure 6.3: Responses and Control Signals for a Plant of Higher Type.

the pole is moved to 0.983, with the resulting pulse-transfer function

$$H_{pq}(z)=\frac{96.92\times 10^{-6}z}{z^2-1.983z+0.983}, \tag{6.6}$$

the resulting output from the classically controlled is unstable, which is the case if the pole is moved any closer to 1 than 0.9802. The closed-loop responses for this plant is depicted in Figure 6.4, from which it can be seen that the response from the model-reference adaptive controller and the self-tuning regulator are unchanged by these variations in the model.

Suppose that the system, rather than being open-loop stable, is open-loop unstable. This is accomplished if a pole is in the right hand side of the s-plane for the continuous-time plant and if a pole is on or outside the unit disc for the discrete-time model. Suppose that the continuous-time plant is described by

$$G_p(s)=\frac{2}{(s+2)(s-1)}. \tag{6.7}$$

The resulting pulse-transfer function is

$$H_{pq}(z)=\frac{0.0001003z+0.0001007}{z^2-2.01z+1.01}=\frac{0.0001003(z+1.004)}{(z-1)(z-1.01)}, \tag{6.8}$$

which has a pole at $z=1.01$, which is outside the unit circle. Because the order of he numerator and the order of he denominator of the pulse-transfer function are the same as the assumed model, the self-tuning regulator and the model-reference adaptive controller have no trouble keeping the system stable and within the design parameters, as shown in Figure 6.5. The control effort for the adaptive schemes once again have spikes on the order of 60 during the parameter estimation, but settle down to 20 for the self-tuning regulator and 5 for

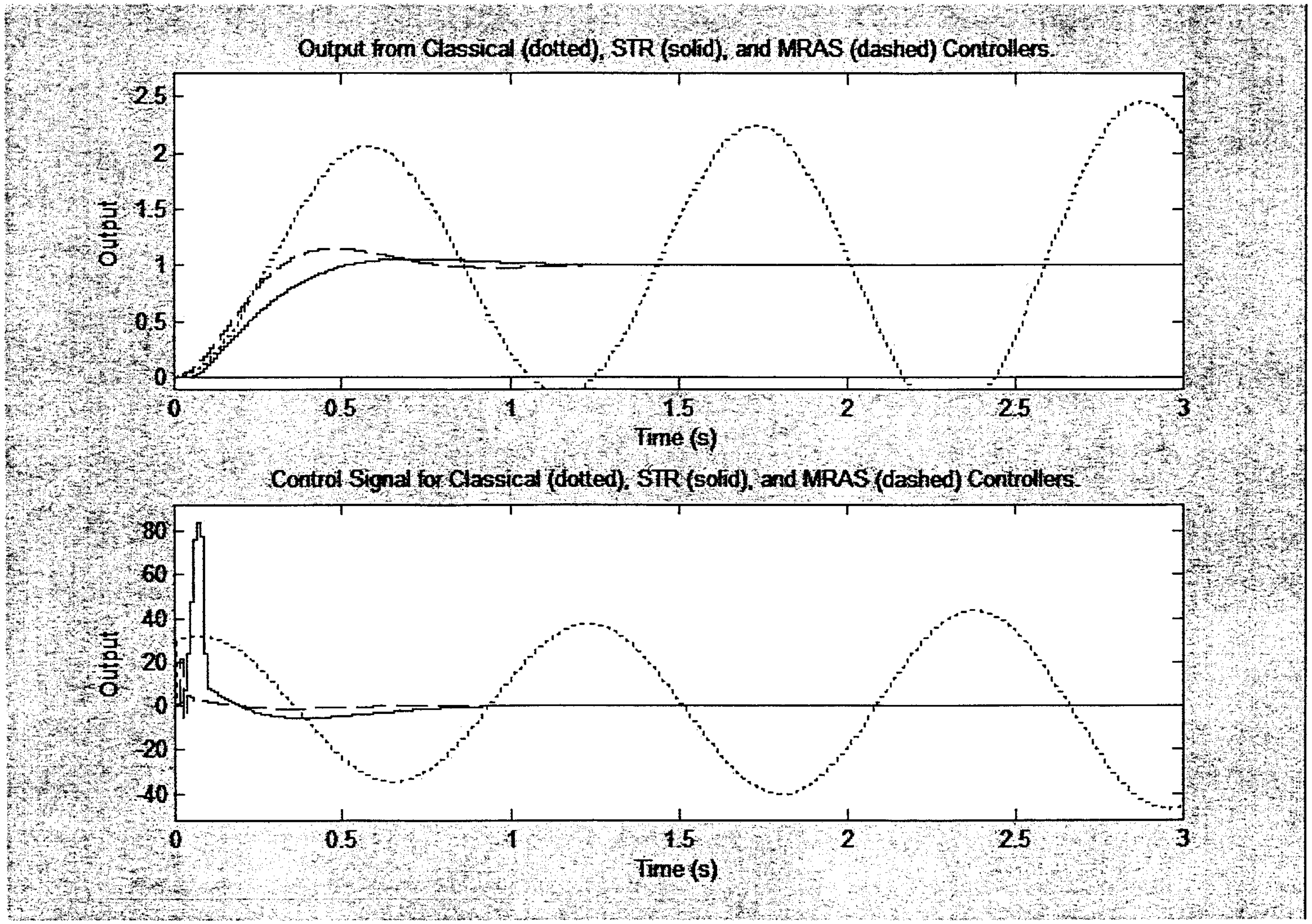

Figure 6.4: Responses and Control Signals for a Plant of Higher Type.

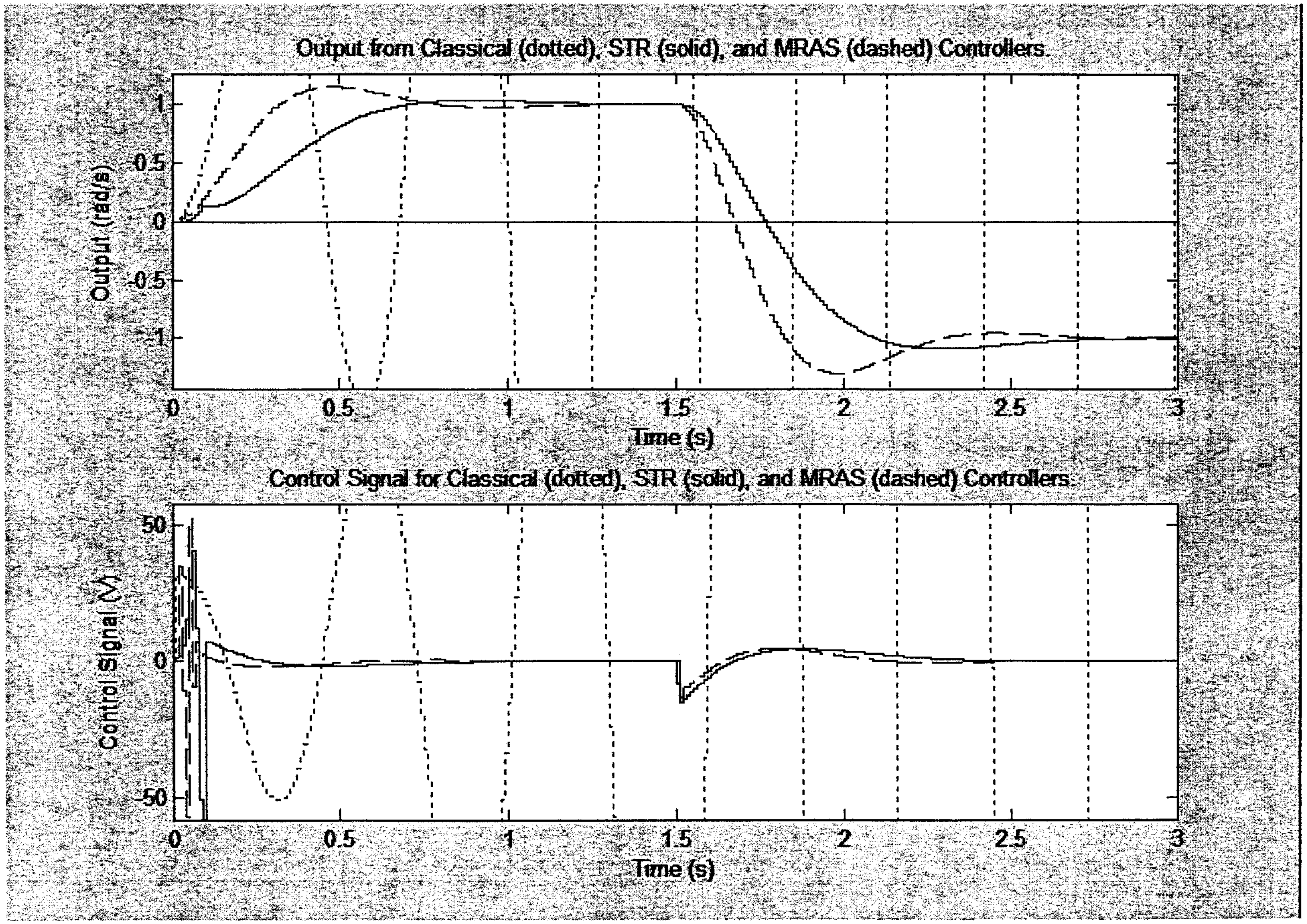

Figure 6.5: Responses and Control Signals for an Unstable Plant.

the model-reference adaptive controller. After the parameter estimates have converged, the control effort of the model-reference adaptive system and the self-tuning regulator are largely the same. On the other hand, because the classical system is unstable, the control signal is unstable as well.

As a final consideration in model-type errors, the assumption will be made that there is an error in the order of the model. Suppose that the transfer function has a pole at $s = -5$, resulting in

$$G_p(s) = \frac{2}{(s+2.0025)(s+5)(s+9.9975)} \tag{6.9}$$

and a pulse-transfer function of

$$H_{pq}(z) = \frac{323\times10^{-9}(z^2+3.867z+0.9350)}{z^3-2.874z^2+2.752z-0.8773} = \frac{323\times10^{-9}(z+3.608)(z+0.2591)}{(z-0.9049)(z-0.9802)(z-0.9891)} \tag{6.10}$$

The simulation for the three different controllers is shown in Figure 6.6. The self-tuning regulator has no problem keeping the output within the desired criteria. This is because the self-tuning regulator does not need the exact values of the system parameters, only a proper ratio of system parameters. The estimator tries to formulate a model that has a second-order denominator and a first-order numerator that fits the input-output relation. The self-tuning regulator then uses the ratios between numerator and denominator coefficients to formulate the control signal. The estimator in the model-reference adaptive controller also attempts to fit the input and output into a second-order denominator and a first-order numerator. However, the values that are estimated are used directly in formulating the control signal, so errors in the model estimation dramatically effect the output; in this case causing the output of the system to be unstable, as can be seen in Figure 6.6. The classically controlled system,

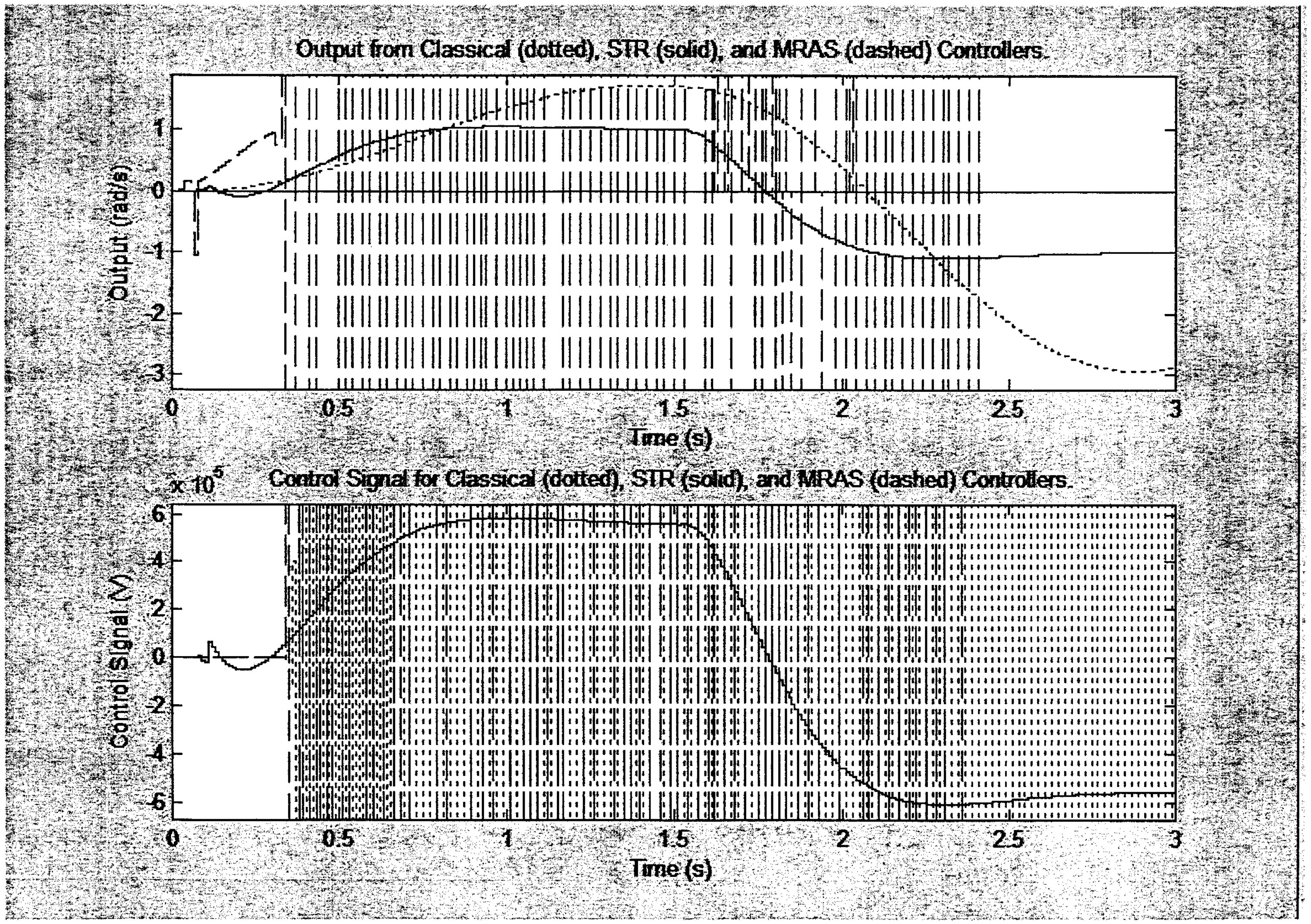

Figure 6.6: Responses and Control Signals for a Plant of Higher Order.

while stable, has an overshoot of 71%, and has a very long settling time. For this situation the self-tuning regulator is the best controller. The control effort for any of these controllers has much to be desired. The self-tuning regulator has the best control effort, but it settles at a value on the order of 6×10^5, while the control signal of the model-reference adaptive controller and the classical controller oscillate with an increasing magnitude. From these results, it is seen that the self-tuning regulator can control a system that has a higher order than designed for, but that the control effort creates an incentive to have the order correct when designing the controller.

6.4 Load Error

Recall that the transfer function of the plant is described by

$$G_p(s) = \frac{\frac{K}{JL}}{s^2 + \left(\frac{B}{J} + \frac{R}{L}\right)s + \frac{RB + K^2}{JL}}, \tag{6.11}$$

as shown in Chapter 1. When using the values given in Equations 1.14, 1.15, 1.16, and 1.17, but letting the value of J vary, Equation 6.11 becomes

$$G_p(s) = \frac{\frac{0.02}{J}}{s^2 + \left(\frac{2J + 0.1}{J}\right)s + \frac{0.2002}{J}}. \tag{6.12}$$

Assuming that the value of J for the motor shaft and load is $1\frac{kg \cdot m^2}{s^2}$; the transfer function is then, after plugging the value of J into Equation 6.12,

$$G_p(s) = \frac{0.02}{s^2 + 2.1s + 0.2002} = \frac{0.02}{(s + 2.000)(s + 0.1001)}, \tag{6.13}$$

with corresponding pulse-transfer function as

$$H_{pq}(z) = \frac{99.3 \times 10^{-6}(z + 0.9931)}{z^2 - 1.979z + 0.9792}. \tag{6.14}$$

Figure 6.7 shows the simulation of the system controlled by the classical controller, a self-tuning regulator and a model-reference adaptive controller. The performance of both the self-tuning regulator and the model-reference adaptive controller are unaffected by the change is system parameters. However, the classical controller's performance is drastically reduced. By virtue of the integrator the steady-state error is zero, but the settling time is greatly increased. At a time of $3s$, the output is still only at 93.4% of the final value. The overshoot requirement is met; however, the controller does not reduce the settling time as compared to the open-loop system. It is reasonable to assume that as the load increases, the transfer function coefficients vary more from the unloaded values. It follows that the system response for the classically controlled system will be poorer as the difference in coefficients grows. The most extreme case that will be addressed is when the inertial constant of the motor and the load is $5\frac{kg \cdot m^2}{s^2}$. Substituting this value of J into Equation 6.12 results in the transfer function,

$$G_p(s) = \frac{0.004}{s^2 + 2.02s + 0.04004}, \tag{6.15}$$

which has the pulse-transfer function,

$$H_{pq}(z) = \frac{19.87 \times 10^{-9}(z + 0.9929)}{z^2 - 1.98z + 0.98}. \tag{6.16}$$

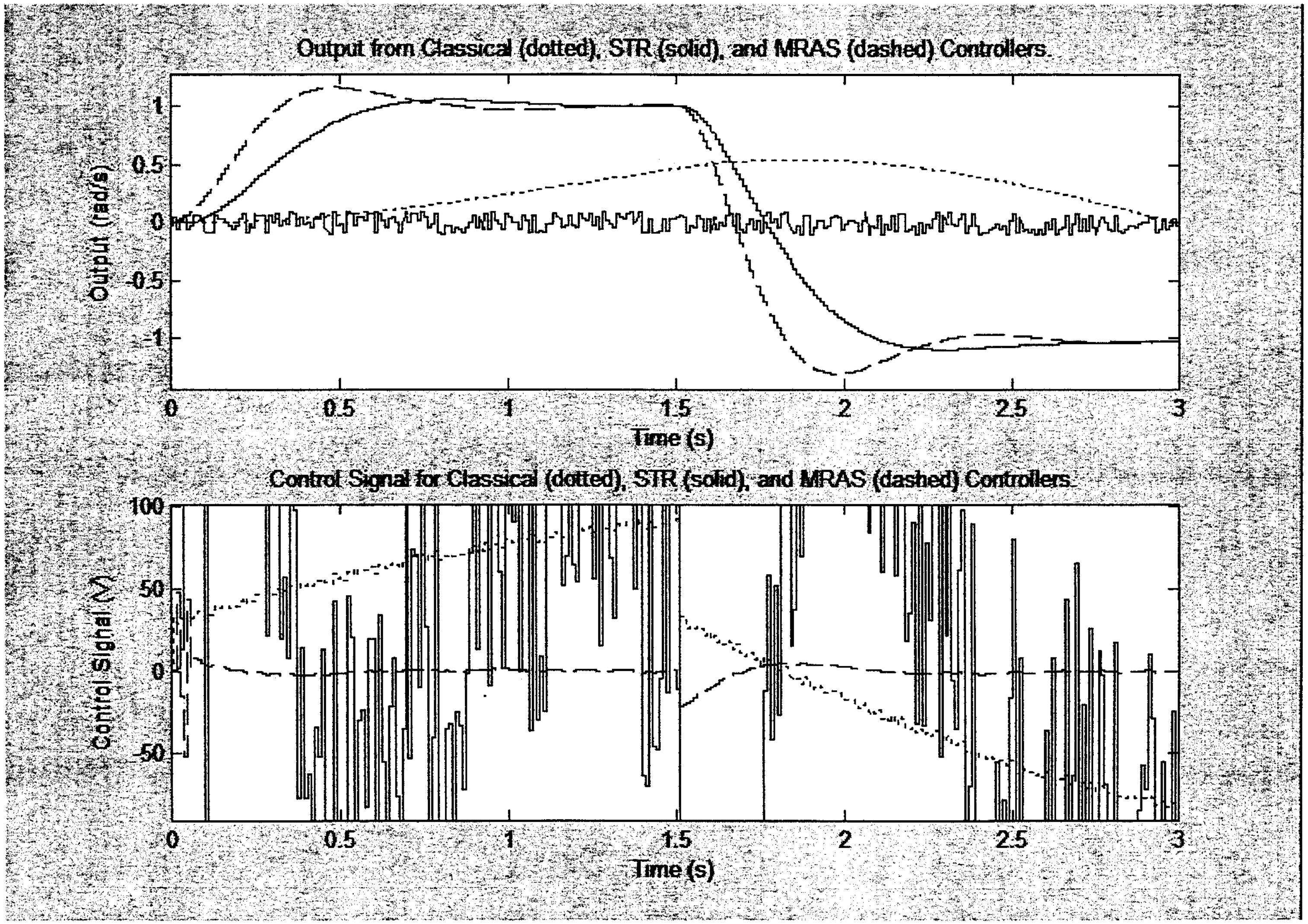

Figure 6.7: Responses and Control Signals for a Loaded Plant.

The zero of the pulse-transfer function remains in the same vicinity, as do the pole. The main impact that changing the inertia of the load has on the pulse-transfer function is to change the value and magnitude of the dc-gain. Because the dc-gain is so small, it greatly lessens the effect of the gain introduced by the classical controller. This decreased effect of gain tends to keep the closed-loop poles in very close proximity to the open-loop poles. The controller has little effect on the damping ratio and natural frequency, and thus little effect on the overshoot and settling time of the system. The presence of the integrator does indicate that eventually, the system will reach zero steady-state error. The self-tuning regulator does not remain unaffected by the change in dc gain. While the output remains the same, the control effort is increased by a similar magnitude as the dc-gain of the pulse-transfer function is decreased. Figure 6.8 shows the simulation of the system with $J = 5\frac{kg \cdot m^2}{s^2}$. The classically-controlled system has reached a value that is 5.3% of its final value by $3s$. The output of the self-tuning regulator and the model-reference adaptive controller are both unchanged by the change in system parameters.

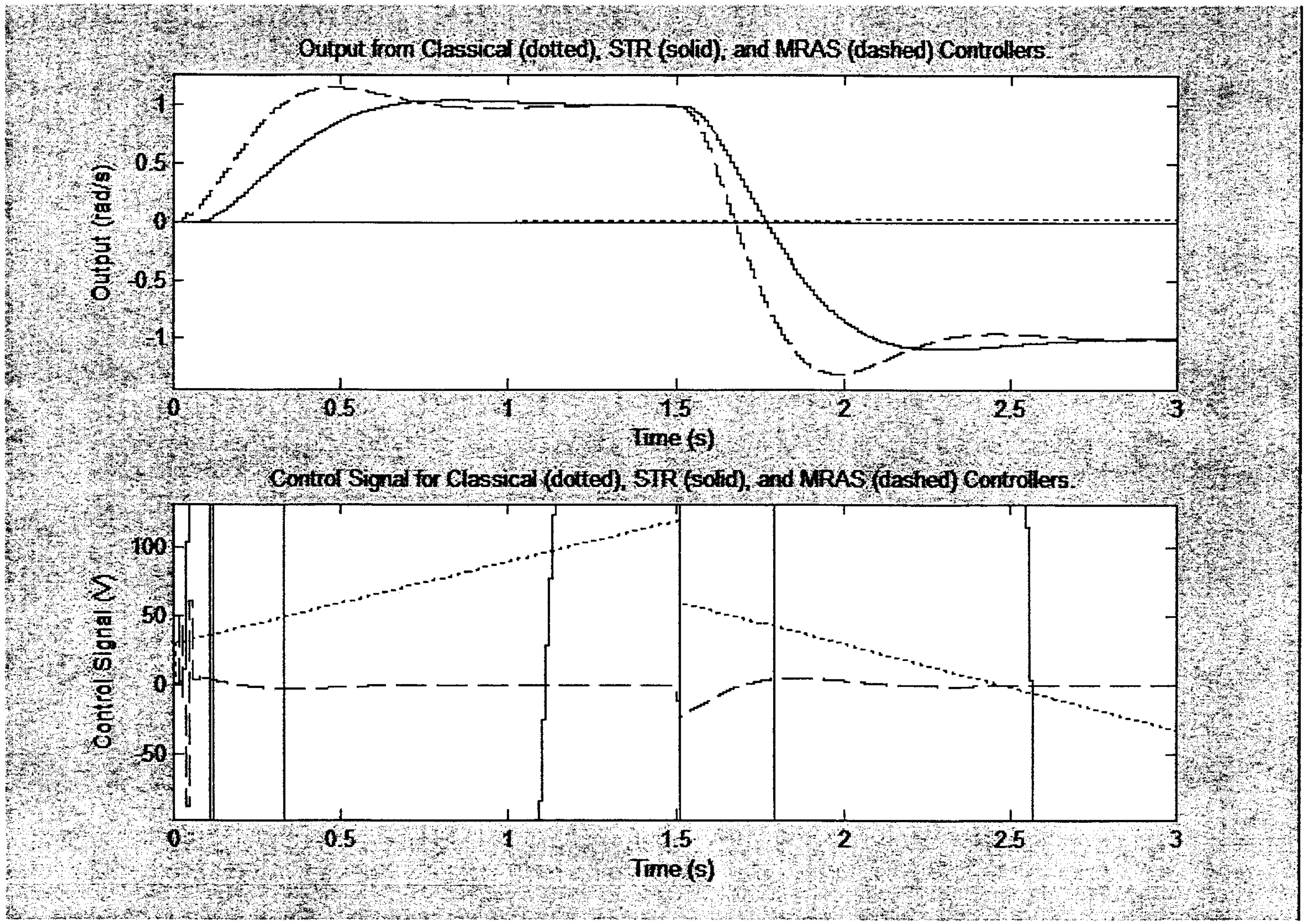

Figure 6.8: Responses and Control Signals for a Maximally Loaded Plant.

CHAPTER VII

SUMMARY AND CONCLUSIONS

7.1 Summary

In the first chapter a mathematical model for a dc motor was developed, which was formulated as a continuous-time transfer function. Subsequently, a pulse-transfer function was derived as well. The desired characteristics of the output of the plant were used to determine a desired transfer and pulse-transfer function. It was shown that any variation on the load would change the poles and dc gain of the transfer function.

The classical root locus technique was used to design the controller that was given by

$$D(z) = 30.2\frac{z - 0.9800}{z - 1}. \qquad 7.1$$

which exceeds the requirements of overshoot of 8%, a settling time of $1s$, and zero steady-state error.

A self-tuning regulator was designed that had total pole-zero cancellation. This adaptive controller resulted in an output that reached its steady-state in about $1s$, with an overshoot of only 6% and zero steady-state error. This controller, however, produced a control signal whose value greatly changed from sample to sample, a phenomenon known as ringing. A second self-tuning regulator was designed that did not have pole-zero cancellation. The output of this system has zero steady-state error, an overshoot of

4% and a settling time of $1s$. The control signal produced by this controller did not have ringing and also had a smaller magnitude.

The third type of controller was a one-step ahead adaptive controller. This type of controller met the overshoot and settling time specifications of 15% and $1s$ exactly. Two variants, weighted and unweighted one-step ahead controllers were considered. The unweighted controller resulted in zero steady-state error, however the control signal exhibited ringing and had too great a control effort. The ringing and control effort problems were lessened by weighting the control effort. This introduced some steady-state error into the system.

The fourth and final type of controller considered , model-reference adaptive controllers, is related to the one-step ahead controller. Rather than trying to follow the reference signal, the controller attempts to follow the output of an acceptable model to the reference signal. This results in the lessening the control effort. In the model-reference adaptive controller, the overshoot and settling times were met exactly. As in one-step ahead control, the unweighted output had the greatest control effort as well as some ringing in the control signal. There was a direct correlation between the amount of steady-state error and the weight that was given to having a low control effort.

The control efforts of each of the controllers were compared. The lowest control effort was generated by the classical controller. The self-tuning regulator had the greatest control effort, while the model-reference adaptive system had a moderate control effort. Based solely on the design criteria of the unloaded dc motor whose mathematical model was entirely correct and the control effort, the best controller would be the classical controller. Each off the three controllers exhibited an acceptable amount of inherent disturbance rejection, with the model-reference adaptive controller displaying the greatest disturbance rejection, and the classical controller displaying the least amount of disturbance rejection.

Several situations in which the assumed model was inexact were investigated with the three different controllers, the classical controller, the self-tuning regulator, and the model reference adaptive controller. The variations from the ideal model were obtained by including an error in the type of the system, the stability of the system, and the order of the system. When the plant was of a higher type than the assumed plant, the classical controller remained in sustained oscillation for closed-loop response. The self-tuning regulator and the model-reference adaptive controller remained unaffected by this type of error. The control effort was largely unaffected for all three of the controllers.

For the second type of plant variation from the ideal case, one of the poles of the plant was moved to the right half plane, resulting in an unstable system. When the classical controller was used, the system remained unstable. Once again, neither the self-tuning regulator nor the model-reference adaptive controller were effected by this change. Again, the control effort of the three controllers for the unstable case remained largely unchanged from that of the stable case. The variation from the ideal model was created by adding a pole to the open-loop transfer function, increasing the order of the system. The output of the self-tuning regulator controlled system was unaffected by this error. The control effort was slightly greater with this variation as compared to the ideal case. The classically controlled system, while stable, did not perform within the desired criteria. The overshoot was 71% and the settling time was great. This change had a catastrophic effect on the model-reference adaptive controlled system, which could be avoided by over-parameterizing the model used in design. Both the control effort and the output of the model-reference adaptive system grew without bound.

The final changes that were made to the system were changes to the load of the dc motor, which changed the location of the poles and zeroes of the transfer function from the assumed poles and zeroes of the transfer function. For the self-tuning regulator and the model-reference adaptive controller, the response did not change. However, as the load was increased, the control effort for both the self-tuning regulator and the model-reference adaptive controller increased. Increasing the load also increased the control effort of the classical controller. In addition, the settling time also greatly increased as the load to the controller increased.

7.2 Conclusions

The classical controller has several disadvantages. The first is that it results in an unstable closed-loop system if the plant is of higher type. Further, this controller cannot compensate for an unstable plant. The most likely situation that is disadvantageous for the classical controller is when the dc motor is loaded, for which the settling time is greatly increased over the open loop transfer characteristic.

If the order of the system is known, then the model-reference adaptive controller is the best controller because it can follow a desired output no matter what the load, the type, or the stability of the plant is. However, because of the catastrophic reaction to a system of higher order than assumed, the model-reference adaptive is a bad controller if there is the possibility of a higher order plant. Designing the controller assuming a higher order for the controller would be a solution for this type of error, but would increase the number of calculation required at each iteration. The increase in calculations should not present any problem because inexpensive high-speed processors are readily available.

If the smallest control effort is not absolutely necessary, the self-tuning regulator is the best choice for a controller. It is able to retain its response even if the plant is of a different order, type or even unstable. The load on the motor has little effect on the performance of the system controlled by a self-tuning regulator.

The work presented here can be expanded by considering experimental implementation of developed controllers using a microprocessor and digital signal processing. An extension of the implementation would be to observe how quantization of the process effects the system, and which controller is most effected by the quantization. The dependence of the closed-loop system response on the initial guess for the system parameters could also be observed.

REFERENCES

[1] Astrom, Karl J. and Björn Wittenmark. Adaptive Control, Second Edition. New York: Addison-Wesley Publishing Company, 1995.

[2] Chen Jie and Guoxiang Gu. Control-Oriented Sytem Identification, An H_∞ Approach. Hew York: John Wiley and Sons, 2000.

[3] Elbert, Theodore F. Estimation and Control of Systems. New York: Van Nostrand Reinhold Company, 1984.

[4] Example: DC Motor Speed Modeling. *Carnegie Mellon Controls Tutorials for MATLAB*. 16 June 2004. <http://www.engin.umich.edu/group/ctm/examples/motor/motor.html>.

[5] Feuer, A and G.C. Goodwin. *Linear Deterministic Adaptive Control: Fundamental Limitations?*. Systems and Control Letters 49, 2003.

[6] Friedland, Bernard. Control System Design: An Introduction to State-Space Methods. New York: McGraw-Hill Book Company, 1986.

[7] Goodwin, Graham C and Kwai Sang Sin. Adaptive Filtering Prediction and Control. Englewood Cliffs, NJ: Prentice Hall, 1984.

[8] Goodwin, Graham C, Stefan F Graebe and Mario Salgado. Control System Design. Upper Saddle River, NJ: Prentice Hall, 2001.

[9] Grewal Mohinder S and Angus P Andrews. Kalman Filtering, Theory and Practice. Englewood Cliffs, NJ: Prentice Hall, 1993.

[10] Ilchmann, A. and E.P. Ryan. *On Tracking and Disturbance Rejection By Adaptive Control.* Systems and Controls Letters 52, 2004.

[11] Kuo, Benjamin C. Automatic Control Systems, Seventh Edition. Englewood Cliffs, NJ: Prentice Hall, 1995.

[12] Lindfield, George and John Penny. Numerical Methods Using MATLAB, Second Edition. Upper Saddle River, NJ: Prentice Hall, 2000.

[13] Mossayebi, Faramarz. *Adaptive Control of Chaotic Systems.* The University of Akron: Dissertation, 1994.

[14] Nise, Norman S. Control System Engineering, Third Edition. New York: John Wiley and Sons, 2000.

[15] Ogata, Katsuhiko. Discrete-Time Control Systems, Second Edition. Upper Saddle River: Prentice Hall, 1995.

[16] Phillips, Charles C and H. Troy Nagle. Digital Control System Analysis and Design, Second Edition. Englewood Cliffs, NJ: Prentice Hall, 1984.

[17] Pratap, Rudra. Getting Started with MATLAB. Orlando: Harcourt Brace College Publishers, 1996.

APPENDIX A:

MATRIX DERIVATIVE AND MATRIX INVERSION LEMMA

A.1 Matrix Derivative

If $\mathbf{f}(\mathbf{x})$ is a vector function of the vector $\mathbf{x}$, then the formal definition of the partial derivative of $\mathbf{f}(\mathbf{x})$ with respect to $\mathbf{x}$ is

$$\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & & \frac{\partial f_1}{\partial x_N} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & & \frac{\partial f_2}{\partial x_N} \\ & & & \\ \frac{\partial f_M}{\partial x_1} & \frac{\partial f_M}{\partial x_2} & & \frac{\partial f_M}{\partial x_N} \end{bmatrix}. \tag{A.1}$$

Two common vector functions that are encountered are

$$\mathbf{f}(\mathbf{x}) = \mathbf{A}\mathbf{x} \tag{A.2}$$

and

$$\mathbf{f}(\mathbf{x}) = \mathbf{x}^T\mathbf{A}\mathbf{x}. \tag{A.3}$$

For the first function, the definition of the partial derivative leads to

$$\frac{\partial(\mathbf{A}\mathbf{x})}{\partial \mathbf{x}} = \frac{\partial}{\partial \mathbf{x}}\left(\begin{bmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N \\ \vdots \\ a_{N1}x_1 + a_{N2}x_2 + \cdots + a_{NN}x_N \end{bmatrix}\right), \tag{A.4}$$

which becomes, upon performing the partial derivatives,

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1N} \\ a_{21} & a_{22} & \cdots & a_{2N} \\ \vdots & & & \vdots \\ a_{N1} & a_{N2} & \cdots & a_{NN} \end{bmatrix} = \mathbf{A}\,. \tag{A.5}$$

For the second function,

$$\mathbf{x}^T\mathbf{A}\mathbf{x} = x_1(a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N) + x_2(a_{21}x_1 + a_{22}x_2 + \cdots + a_{2N}x_N) + \cdots$$
$$+ x_N(a_{N1}x_1 + a_{N2}x_2 + \cdots + a_{NN}x_N) \tag{A.6}$$

Differentiating $\mathbf{x}^T\mathbf{A}\mathbf{x}$ with respect to x_1 yields

$$\frac{\partial(\mathbf{x}^T\mathbf{A}\mathbf{x})}{\partial x_1} = 2a_{11}x_1 + a_{12}x_2 + \cdots + a_{1N}x_N + x_2a_{21} + \ldots x_Na_{N1}\,. \tag{A.7}$$

Collecting like terms in x_i results in the equation,

$$\frac{\partial(\mathbf{x}^T\mathbf{A}\mathbf{x})}{\partial x_1} = 2a_{11}x_1 + (a_{12} + a_{21})x_2 + \cdots + (a_{1N} + a_{N1})x_N\,. \tag{A.8}$$

Similarly the partial derivative with respect to x_i is

$$\frac{\partial(\mathbf{x}^T\mathbf{A}\mathbf{x})}{\partial x_i} = (a_{1i} + a_{i1})x_1 + \cdots + 2a_{ii}x_i + \cdots + (a_{1N} + a_{N1})x_N\,. \tag{A.9}$$

If $\mathbf{A}$ is symmetric, then Equation (A.9) becomes

$$\frac{\partial(\mathbf{x}^T\mathbf{A}\mathbf{x})}{\partial x_i} = 2(a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{iN}x_N), \tag{A.10}$$

which is the i^{th} entry in the row vector forming the partial derivative. This quantity is twice as large as the i^{th} entry in the row vector $\mathbf{x}^T\mathbf{A}$. Thus, as long as $\mathbf{A}$ is symmetric,

$$\frac{\partial(\mathbf{x}^T\mathbf{A}\mathbf{x})}{\partial \mathbf{x}} = \mathbf{x}^T\mathbf{A} \tag{A.11}$$

holds.

A.2 Matrix Inversion Lemma

If the matrices A, C, and $C^{-1}+DA^{-1}B$ are nonsingular, then the matrix $A+BCD$ is nonsingular, and

$$(A+BCD)^{-1}=A^{-1}-A^{-1}B\left(C^{-1}+DA^{-1}B\right)^{-1}DA^{-1} \tag{A.12}$$

is true. This is shown by multiplication of $A+BCD$ by its inverse. This yields $(A+BCD)\left(A^{-1}-A^{-1}B\left(C^{-1}+DA^{-1}B\right)^{-1}DA^{-1}\right)$. The product of the first term of each multiplicand, which is the first term, is the identity matrix, I. The second term, which is the product of the first term of the first multiplicand and the second term of the second multiplicand, is $-B\left(C^{-1}+DA^{-1}B\right)^{-1}DA^{-1}$. The product of the second term of the first multiplicand and the first term of the second multiplicand is $BCDA^{-1}$, which is the third term. The fourth term, which is product of the second term of each multiplicand, is $-BCDA^{-1}B\left(C^{-1}+DA^{-1}B\right)^{-1}DA^{-1}$. The sum of the second and fourth term is

$$\begin{aligned}
&-B\left(C^{-1}+DA^{-1}B\right)^{-1}DA^{-1}-BCDA^{-1}B\left(C^{-1}+DA^{-1}B\right)^{-1}DA^{-1}\\
&=-B\left(\left(C^{-1}+DA^{-1}B\right)^{-1}+CDA^{-1}B\left(C^{-1}+DA^{-1}B\right)^{-1}\right)DA^{-1}\\
&=-BC\left(C^{-1}\left(C^{-1}+DA^{-1}B\right)^{-1}+DA^{-1}B\left(C^{-1}+DA^{-1}B\right)^{-1}\right)DA^{-1}\\
&=-BC\left(\left(C^{-1}+DA^{-1}B\right)\left(C^{-1}+DA^{-1}B\right)^{-1}\right)DA^{-1}=-BCDA^{-1}
\end{aligned}$$

The sum of the second and fourth term, thus cancels the third term, leaving only the first term, the identity matrix. Thus, $A+BCD$ and $A^{-1}-A^{-1}B\left(C^{-1}+DA^{-1}B\right)^{-1}DA^{-1}$ are inverses.

APPENDIX B:

LINEAR PREDICTION MODEL

A linear discrete-time system can be written as

$$A_p^*(q^{-1})y(k) = q^{-d}B_p^*(q^{-1})u(k), \tag{B.1}$$

where

$$A_p^*\left(q^{-1}\right) = 1 + a_1 q^{-1} + \cdots + a_n q^{-n} \tag{B.2}$$

and

$$B_p^*\left(q^{-1}\right) = q^{-d}\left(b_0 + b_1 q^{-1} + \cdots + b_m q^{-m}\right) = q^{-d}B'\left(q^{-1}\right). \tag{B.3}$$

as described in Equation 1.37. The output of the system at time $k+d$ can be written in the predictor form,

$$y(k+d) = \alpha\left(q^{-1}\right)y(k) + \beta\left(q^{-1}\right)u(k), \tag{B.4}$$

where

$$\alpha\left(q^{-1}\right) = \alpha_0 + \alpha_1 q^{-1} + \cdots + \alpha_{n-1}q^{-n+1} = G\left(q^{-1}\right) \tag{B.5}$$

and

$$\beta\left(q^{-1}\right) = \beta_0 + \beta_1 q^{-1} + \cdots + \beta_{m+d-1}q^{-m-d+1} = F\left(q^{-1}\right)B'\left(q^{-1}\right). \tag{B.6}$$

$F\left(q^{-1}\right)$ and $G\left(q^{-1}\right)$ are the unique polynomials that satisfy

$$1 = F\left(q^{-1}\right)A\left(q^{-1}\right) + q^{-d}G\left(q^{-1}\right), \tag{B.7}$$

with

$$F\left(q^{-1}\right) = f_0 + f_1 q^{-1} + \cdots + f_{d-1}q^{-d+1}, \tag{B.8}$$

and

$$G\left(q^{-1}\right) = g_0 + g_1 q^{-1} + \cdots + g_{n-1} q^{-n+1}. \tag{B.9}$$

The coefficients are computed in the following manner from Equation B.7.

$$f_0 = 1, \tag{B.10}$$

$$f_i = -\sum_{j=0}^{i-1} f_j a_{i-j}, \tag{B.11}$$

and

$$g_i = -\sum_{j=0}^{d-1} f_j a_{i+d-j}. \tag{B.12}$$

APPENDIX C

MATLAB CODE

Code for Classical Controller Simulation

Root Locus

```
Bq=[0.961e-4 0.9233e-4]; % Set uncompensated numerator

Aq=[1 -1.885 0.8869]; % Set uncompensated denominator

Ts=0.01; % Set sampling time

sysol=tf(Bq,Aq,Ts); % Formulate uncompensated open-loop system

Aqint=conv(Aq,[1 -1]); % Determine compensated denominator

Bqint=conv(Bq,[1 -0.98]); % Determine compensated numerator

sysint=tf(Bqint,Aqint,Ts); %Formulate compensated open-loop system

rlocus(sysint) % Perform root locus

zgrid(0.5169,0.07738) % Add contours of constant damping ratio and natural frequency

axis([0.8 1.2 -.2 .2]) % Specify graph size
```

Step Response

```
Kos=42.8;

Bq=[0.961e-4 0.9233e-4]; % Set uncompensated numerator

Aq=[1 -1.885 0.8869]; % Set uncompensated denominator

Ts=0.01; % Set sampling time

sysol=tf(Bq,Aq,Ts); % Formulate uncompensated open-loop system

Aqint=conv(Aq,[1 -1]); % Determine compensated denominator
```

```
Bqint=conv(Bq,[1 -0.98]); % Determine compensated numerator

sysint=tf(Bqint,Aqint,Ts); % Formulate compensatted open-loop system

syscl=feedback(Kos*sysint,1); % Formulate closed-loop system

step(syscl) % Find the closed-loop step response

title('Matching the Overshoot Criteria')
```

Self-Tuning Regulator Simulation

```
clear

clf

lambda=0.1; % Initialize forgetting factor

T=0.01; % Set sampling time

k=0:300; %Initialize sample index

N=length(k); % Determine the number of samples

uc=1*sign(sin((2*pi*k)/400)); % Formulate the reference signal

% Designate Plant

B=[96.1e-6 92.33e-6]; % Set uncompensated numerator

A=[1 -1.885 0.8869]; % Set uncompensated denominator

 % Extract transffer function coefficients

 br0=B(1);

 br1=B(2);

 ar1=A(2);

 ar2=A(3);

% Designate Desired Transfer Function

Bm=[0.1761 0]; % Set desired numerator
```

```
Am=[1 -1.92 0.9231]; %Set desired denominator
 % Extract desired coefficients
 bm0=Bm(1);
 bm1=Bm(2);
 am1=Am(2);
 am2=Am(3);
 a00=0;
% Initialize estimation variables
e=zeros(1,N);
K=zeros(4,N);
theta=zeros(4,N);
den=zeros(1,N);
% Set initial parameter estimate
theta(:,1)=[0;0;0.01;0.2];
% Perform first two iterations of estimation and output
theta(:,2)=theta(:,1);
P(:,:,2)=diag([100 100 100 100]);
u(1)=uc(1);
u(2)=uc(2);
y(1)=0;
y(2)=br0*u(1);
% Perform ramaining iterations of estimation, conttrol signal formulation
% and output determination
```

```
for m=3:N

   % Determine Output

   y(m)=-ar1*y(m-1)-ar2*y(m-2)+br0*u(m-1)+br1*u(m-2);

      % Estimate Parameters

   Pkm1=P(:,:,m-1);

   phikm1=[-y(m-1) -y(m-2) u(m-1) u(m-2)]';

   thetakm1=theta(:,m-1);

   e(m)=y(m)-phikm1'*thetakm1;

   K(:,m)=Pkm1*phikm1*inv(lambda+phikm1'*Pkm1*phikm1);

   P(:,:,m)=(diag([1 1 1 1])-K(:,m)*phikm1')*Pkm1/lambda;

   theta(:,m)=thetakm1+K (:,m)*e(m);

   a1=theta(1,m);

   a2=theta(2,m);

   b0=theta(3,m);

   b1=theta(4,m);

      % Formulate Control Signal

   den(m)=a2*b0*b0+b1*b1-a1*b0*b1;

   %Determine the S polynomial

   s0=(b0*(a00*a2+a2*am1-a1*a2-a00*am2)+b1*(a1*a1+am2+a00*am1-am1*a1-
a2))/den(m);

   s1=(b0*(a00*a2*am1+am2*a2-a00*a1*am2-a2*a2)+b1*(a1*a2+a00*am2-a00*a2-
a2*am1))/den(m);

   S(:,m)=[s0;s1];
```

```
    %Determine the R polynomial
    r0=1;
    r1=(b0*b0*am2*a00-b0*b1*(am2-a2+am1*a00)+b1*b1*(am1-a1+a00))/den(m);
    R(:,m)=[r0;r1];
    %Determine the T polynomial
    tt=((1+am1+am2)/(b0+b1));
    t0=tt;
    t1=a00*tt;
    Tmat(:,m)=[t0;t1];
    u(m)=-r1*u(m-1)+t0*uc(m)+t1*uc(m-1)-s0*y(m)-s1*y(m-1);
end
%Plot the system response
t=k*T;
stairs(t,uc,'r')
hold on
stairs(t,y,'b')
title('Output (blue) and Input (red) of a Self-Tuning Regulator without Zero Cancelation')
xlabel('Time (s)')
ylabel('Output')
```

Model-Reference Adaptive System Simulation Code

```
% MRAS Weighted
clear
clf
```

```
lambda=0.1; % Determine forgetting factor for estimation
lamc=0.0000001; % Determine small control signal weighting
T=0.01; % Set sampling time
k=0:300; % Create sampling index
N=length(k);
uc=sign(sin((2*pi*k)/400));
% Designate Plant
B=[0.961e-4 0.9233e-4]; % Set uncompensated numerator
A=[1 -1.885 0.8869]; % Set uncompensated denominator
 % Extract transfer function coefficients
 br0=B(1);
 br1=B(2);
 ar1=A(2);
 ar2=A(3);
% Designate Desired Output
Bm=[0.00291428474 0.0028375860]; % Set desiered numerator
Am=[1 -1.91736448 0.923111635]; % Set desired denominator
ym1=filter(Bm,Am,uc); % Determine the desired system response
ym=[ym1 ym1(N)]; % Increase the size of the desired output by one
%Initialize estimation variables
e=zeros(1,N);
K=zeros(4,N);
theta=zeros(4,N);
```

```
theta(:,1)=[-2;1;0;0];

% Perform the first two iterations of estimation, control signal

% formulation, and output determination

theta(:,2)=theta(:,1);

P=diag([100 100 100 100]);

u(1)=uc(1);

u(2)=uc(2);

y(1)=0;

y(2)=br0*u(1);




% At each iteration after the first

for m=3:N

  % Determine Output

  y(m)=-ar1*y(m-1)-ar2*y(m-2)+br0*u(m-1)+br1*u(m-2);

  % Estimate Parameters

  Pkm1=P;

  phikm1=[-y(m-1) -y(m-2) u(m-1) u(m-2)]';

  thetakm1=theta(:,m-1);

  e(m)=y(m)-phikm1'*thetakm1;

  K(:,m)=Pkm1*phikm1*inv(lambda+phikm1'*Pkm1*phikm1);

  P=(diag([1 1 1 1])-K(:,m)*phikm1')*Pkm1/lambda;
```

```
    theta(:,m)=thetakm1+K(:,m)*e(m);
    a1=theta(1,m);
    a2=theta(2,m);
    b0=theta(3,m);
    b1=theta(4,m);
    % Formulate Control Signal
    u(m)=b0*(ym(m+1)+a1*y(m)+a2*y(m-1)-b1*u(m-1))/(b0*b0+lamc);
end
% Plot data
t=k*T;
% Plot input and controlled output
subplot(211)
plot(t,uc,'r')
hold on
stairs(t,y,'b')
title('Output (blue) and Input (red) of a Weighted MRAS OSA-Controlled DC Motor for
\lambda = 100*10^-^9')
xlabel('Sample')
ylabel('Output')
subplot(212)
% Plot control signal
stairs(t,u)
title('Control Signal')
```

```
xlabel('Sample')

ylabel('uc(t)')
```