INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI[®]

Improvements to Stochastic Multiple Model Adaptive Control: Hypothesis Test Switching and a Modified Model Arrangement

by

Alexander S. Campbell, B.Sc.

A thesis submitted to

The Faculty of Graduate Studies and Research

in partial fulfilment of the requirements for the degree of

Master of Applied Science in Electrical Engineering

Ottawa-Carleton Institute of Electrical and Computer Engineering

Department of Systems and Computer Engineering

Carleton University

Ottawa, Ontario

August 14, 2005

©Copyright

2005, Alexander S. Campbell

Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

0-494-08368-9

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file *Votre référence*

ISBN:

Our file *Notre référence*

ISBN:

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

The undersigned hereby recommend to
The Faculty of Graduate Studies and Research
Acceptance of the thesis

**Improvements to Stochastic Multiple Model Adaptive Control:
Hypothesis Test Switching and a Modified Model Arrangement**

Submitted by
Alexander S. Campbell

In partial fulfillment of the requirements for the degree of
Master of Applied Science

Professor Howard M. Schwartz, Thesis Supervisor

Professor Rafik A. Goubran, Chair
Department of Systems and Computer Engineering

Carleton University
August 2005

Abstract

This work demonstrates the fusion of two concepts in switching systems, namely, hypothesis testing and multiple model adaptive control. A hypothesis test switching method is defined to detect parameter jumps in a stochastic environment and perform model selection. The control of a discrete-time stochastic system with rapidly time-varying parameters is simulated. Hypothesis test switching is compared to performance index switching, the most researched and popular switching method. It is found that the hypothesis test method is unique because it operates optimally, without user adjustment or *a priori* knowledge of the time-varying conditions and model placement. Furthermore, it provides more accurate switching and lower control error.

In addition, a major modification to the way multiple models are arranged is proposed. The change decouples stability and performance. As a result, stability is proven more easily, previously required assumptions can be relaxed, new switching methods can be applied, and performance increases are simulated using current switching methods.

Acknowledgements

I would like to thank Professor Howard M. Schwartz for his thesis supervision, first-rate guidance, and endless supply of ideas. It was in anticipation of working with his support that I came to Ottawa. I also would like to thank my parents and Meghan. Their support and love makes everything enjoyable.

Contents

Abstract	ii
Acknowledgements	iii
Contents	iv
List of Figures	ix
List of Tables	xiii
1 Introduction	1
1.1 Background	1
1.1.1 Multiple Model Adaptive Control	1
1.1.2 Hypothesis Test Parameter Jump Detection	6
1.2 Thesis Overview	9
1.3 Contribution of the Thesis	10
1.3.1 Hypothesis Test Switching	10
1.3.2 Modified Model Arrangement	11
1.3.3 Quantitative Method to Analyze Performance During Multiple Parameter Jumps	12

1.4	Statement of the Problem	13
2	Stochastic Single Model Adaptive Control	17
2.1	Introduction	17
2.2	Parameter Estimation	18
2.2.1	Selection of Algorithm	18
2.2.2	Definition of the Extended Least Squares Algorithm	20
2.3	Control Using Certainty Equivalence	23
2.4	Stability	25
2.5	Simulations	32
2.5.1	Conditions Common to All Simulations	33
2.5.2	Comparison of RLS and ELS Parameter Estimation Methods .	35
2.5.3	Selection of Forgetting Factor, λ , for ELS in Time-Varying Sys- tems	43
2.5.4	Demonstration of Persistency of Excitation Requirements for ELS	48
3	Stochastic Multiple Model Adaptive Control	53
3.1	Introduction	53
3.2	Classical Model Arrangement	55
3.3	Modified Model Arrangement	57
3.4	Performance Index Switching	59
3.5	Stability	61
3.5.1	Introduction	61
3.5.2	Stability of Two Adaptive Models	64

3.5.3	Stability of One Fixed Model and One Free-Running Adaptive Model with Performance Index A Switching	71
3.5.4	Stability of One Fixed Model and One Free-Running Adaptive Model with Performance Index B Switching	83
3.5.5	Stability of the Classical Model Arrangement with Performance Index B Switching	87
3.5.6	Stability of the Modified Model Arrangement	88
3.6	Simulations	89
3.6.1	Conditions Common to All Simulations	90
3.6.2	Demonstration of Convergence with Random Switching Between Two Adaptive Models	92
3.6.3	Comparison of Multiple and Single Model Control Performance	97
3.6.4	Comparing Performance of the Classical and Modified Model Arrangement with Performance Index B Switching	105
3.6.5	Comparing Performance of the Classical and Modified Model Arrangement with Incorrect Switching	108
4	Hypothesis Test Switching	112
4.1	Introduction	112
4.2	Definition of the Hypothesis Test	113
4.3	Stability	116
4.3.1	Stability of the Classical Model Arrangement with Hypothesis Test Switching	116
4.3.2	Stability of the Modified Model Arrangement with Hypothesis Test Switching	120

4.4	Simulations	120
4.4.1	Conditions Common to All Simulations	120
4.4.2	Detailed Demonstration of Hypothesis Test Switching	126
4.4.3	Hypothesis Test Switching with Poor Choices for Rejection Threshold, T_r	129
4.4.4	Comparing Performance Index B Switching and Hypothesis Test Switching in the Absence of Disturbances	136
4.4.5	Comparing Performance Index B and Hypothesis Test Switch- ing with Noise Variance $\sigma^2 = 0.025$	139
4.4.6	Comparing Performance Index B and Hypothesis Test Switch- ing with Noise Variance $\sigma^2 = 0.025$ and Unique Parameter Jumps and Models	150
4.4.7	Comparing Performance Index B and Hypothesis Test Switch- ing with Changing Noise Variance, σ^2	155
5	Conclusions and Recommendations	159
	References	161
	APPENDICES	
A	Mathematical Methods and Definitions	166
A.1	Modified MSE for Quantitative Analysis	166
A.2	Strictly Positive Real	167
A.3	Persistency of Excitation	168
A.4	Monotonically Increasing	168
A.5	Lemma 3 - Necessary for Proving Theorem 2	169

A.6	Cauchy-Schwarz Inequality	170
A.7	Kronecker Lemma	170
A.8	Modified Kronecker Lemma	171
B	Simulation Code	175
B.1	Comparison of RLS and ELS Parameter Estimation Methods	175
B.2	Selection of Forgetting Factor, λ , for ELS in Time-Varying Systems	179
B.3	Demonstration of Persistency of Excitation Requirements for ELS	180
B.4	Demonstration of Convergence with Random Switching Between Two Adaptive Models	181
B.5	Comparison of Multiple and Single Model Control Performance	183
B.6	Comparing Performance of the Classical and Modified Model Arrange- ment with Performance Index B Switching	184
B.7	Comparing Performance of the Classical and Modified Model Arrange- ment with Incorrect Switching	191
B.8	Detailed Demonstration of Hypothesis Test Switching	192
B.9	Hypothesis Test Switching with Poor Choices for Rejection Threshold, T_r	195
B.10	Comparing Performance Index B Switching and Hypothesis Test Switch- ing in the Absence of Disturbances	196
B.11	Comparing Performance Index B and Hypothesis Test Switching with Noise Variance $\sigma^2 = 0.025$	197

List of Figures

1.1	A simplified single model adaptive controller.	2
1.2	A simplified multiple model adaptive controller.	4
1.3	The “car in the desert” switching example.	5
2.1	RLS parameter estimation in the presence of a disturbance with variance $\sigma^2 = 0.5$	38
2.2	ELS parameter estimation in the presence of a disturbance with variance $\sigma^2 = 0.5$	39
2.3	Performing adaptive control with RLS parameter estimation and a disturbance, $w(k)$, of variance $\sigma^2 = 0.5$	40
2.4	Performing adaptive control with ELS parameter estimation and a disturbance, $w(k)$, of variance $\sigma^2 = 0.5$	41
2.5	Comparison of control error, $e_c(k)$, produced by both the RLS and ELS adaptive control methods.	42
2.6	Control error, $e_c(k)$ produced when using different values for forgetting factor, λ (parameter jump at time-step $k = 200$, disturbance of variance $\sigma^2 = 0$).	46

2.7	Control error, $e_c(k)$ produced when using different values for forgetting factor, λ (parameter jump at time-step $k = 200$, disturbance of variance $\sigma^2 = 0.3$).	47
2.8	Performing adaptive control when the control signal, $u(k)$, and disturbance, $w(k)$, are not persistently exciting.	50
2.9	The parameter estimation process when the control signal, $u(k)$, and disturbance, $w(k)$, are not persistently exciting.	51
2.10	The parameter estimation process when only the disturbance, $w(k)$, is not persistently exciting.	52
3.1	The classical model arrangement, N models and N controllers.	56
3.2	The modified model arrangement, excluding the controllers associated with the $N - 2$ fixed models (control signals $u_1 \dots u_{N-1}$ come from controller C_{reset}).	58
3.3	Chart showing the necessary extensions to prove stability of the classical and modified model arrangements (each box represents a stability proof and each arrow is an extension).	63
3.4	Model arrangement containing only two adaptive models (resetting model has no fixed models to make it reset).	65
3.5	The signal $i(k)$ randomly switching between the two adaptive models.	95
3.6	The convergence of the overall adaptive scheme, during the random model switching.	96
3.7	The disturbance, $w(k)$, reference, $y^*(k + 1)$, and change in plant states (used when simulating both the single and multiple model control cases).	101

3.8	Control using a single model, with a jump in parameters at time-step $k = 500$	102
3.9	Control using the classical multiple model arrangement, with a jump in parameters at time-step $k = 500$	103
3.10	Model selections according to performance index B switching (corresponding to multiple model control in Figure 3.9.)	104
3.11	The repetitive tuning process for both the classical and modified model arrangement.	107
3.12	The behavior of the classical model arrangement with arbitrary switching of signal $i(k)$ (reference signal shown as dashed line).	110
3.13	The behavior of the modified model arrangement with arbitrary switching of signal $i(k)$ (reference signal shown as dashed line).	111
4.1	The behavior of the desired plant output and the plant parameter vector during all of the Section 4.4 simulations.	125
4.2	A detailed example of hypothesis test switching (rejection threshold, T_r , shown as dashed line).	128
4.3	The classical model arrangement with $T_r = \infty$	132
4.4	The modified model arrangement with $T_r = \infty$	133
4.5	The classical model arrangement with $T_r = 0$	134
4.6	The modified model arrangement with $T_r = 0$	135
4.7	The classical model arrangement with performance index switching, $d_3 = 0$, and no disturbance, $w(k) = 0$	137
4.8	The classical model arrangement with hypothesis test switching, $T_r = 0$, and no disturbance, $w(k) = 0$	138

4.9	The classical model arrangement with tuned hypothesis test switching, $T_r = 1.4$, achieving the lowest possible MSE of control error.	144
4.10	The tuning process, for the four combinations of model arrangements and switching methods.	145
4.11	The classical model arrangement with tuned hypothesis test switching, $T_r = 3.1$, achieving the lowest possible <i>modified</i> MSE of control error.	146
4.12	The classical model arrangement with tuned performance index B switching, $d_3 = 0.32$, achieving the lowest <i>modified</i> MSE of control error.	147
4.13	The modified model arrangement with tuned hypothesis test switching, $T_r = 7.3$, achieving the lowest <i>modified</i> MSE of control error.	148
4.14	The modified model arrangement with tuned performance index B switching, $d_3 = 0.32$, achieving the lowest <i>modified</i> MSE of control error).	149
4.15	The tuning process for 50 unique control scenarios (unique $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$), Contour (a) is classical arrangement with index B switching, (b) is classical arrangement with hypothesis test switching, (c) is modified arrangement with index B switching, and (d) is modified arrangement with hypothesis test switching.	154
4.16	The tuning process for 50 control scenarios of different noise variance, σ^2 . Contour (a) is classical arrangement with index B switching, (b) is classical arrangement with hypothesis test switching, (c) is modified arrangement with index B switching, and (d) is modified arrangement with hypothesis test switching.	158

List of Tables

2.1	Comparing the steady-state control error, $e_c(k)$, produced by the RLS and ELS methods.	37
2.2	Comparing the MSE of control error, $e_c(k)$, produced using different forgetting factor values, λ , and a disturbance, $w(k)$, with variance $\sigma^2 = 0$	44
2.3	Comparing the MSE of control error, $e_c(k)$, produced using different forgetting factor values, λ , and a disturbance, $w(k)$, with variance $\sigma^2 = 0.3$	45

Chapter 1

Introduction

1.1 Background

1.1.1 Multiple Model Adaptive Control

Adaptive control combines on-line parameter estimation and control for a system with unknown parameters. Stochastic adaptive control is performed when the system is subject to a random disturbance. Should the disturbance not be completely random, it too can be modeled. A diagram of a simple adaptive control scheme is given in Figure 1.1. The parameter estimation algorithm can be implemented to discount old data and make the new data of greater significance. As a result, systems with time-varying parameters and disturbances can be tracked and controlled. An aircraft is an example of a system with time-varying parameters because its dynamics can change significantly with speed, altitude, and angle of attack. A large cargo ship is an example where disturbances are time-varying. The ship's steering autopilot must maintain a direct course by tracking, and compensating for, wind, waves, and current.

Figure 1.1: A simplified single model adaptive controller.

Because parameter estimation and control are performed at the same instant, large transient control error can occur while little is known about the system's parameters. The least is known about the system during initialization or when there are jumps in parameters. To better describe these two cases, the control of a robotic arm is considered. Initially, the adaptive controller has no knowledge concerning each links' mass, actuators' torque, or joints' viscosity. In an attempt to achieve a desired output behavior, a control signal is applied, but the arm does not behave as desired. By observing the correlations between the outputs and inputs, the adaptive controller can parameterize the arm. After significant parameterization, zero control error can be achieved. When this robotic arm is used to pick up a heavy object, a parameter jump occurs because the object is of unknown mass. The adaptive controller must learn the arm's new dynamics to again achieve zero control error. With proper implementation of a method called multiple model adaptive control, the above transient control error can be avoided. Multiple model adaptive control is also effective at removing transient control error due to system faults or subsystem failures (these are different sources of

parameter jumps). It will be made clear how such improvements are achieved when the multiple model control method is defined below.

The system to be controlled is called a plant. The adaptive controller's parameter estimation algorithm constructs a model of the plant on-line. This model's plant parameter estimates are used in an attempt to make a matched controller (this is one adaptive control method). The controller generates a control signal for the plant. With multiple model adaptive control, multiple plant estimating models are implemented. Many of the models can be fixed (not adaptive). Every model has a corresponding controller, as shown in Figure 1.2. Only one controller can be used at each instant to control the plant. When one model is considered to have the most accurate plant parameter estimates, its corresponding controller is used to control the plant. The problem of stochastic multiple model control is how to decide, on-line, which model should be selected. On-line model selection is called switching [1]. Due to the presence of noise, switching is not a trivial task. If an adaptive model is selected, it should be given sufficient time for its parameter estimates to converge. This necessity is called tuning [1]. If a fixed model is selected when adaptive models are also available, the fixed model should not be permanently selected. Model placement is the process of assigning initial or fixed parameter estimates to each model. With proper model placement and on-line switching, virtually all transient control error can be eliminated. Should the system be stochastic, minimum variance control can be achieved.

The use of multiple models can clearly provide increased performance. However, there are some circumstances when switching can be detrimental to performance and even cause instability. Consider the "car in the desert" example taken from [2] and illustrated in Figure 1.3. The car must return to the safety of the oasis and is given two

Figure 1.2: A simplified multiple model adaptive controller.

trajectories, a and b . If the decision to use “stable” trajectory a or b is constantly switched, then trajectory c is taken. This constant switching actually results in instability because the car is constantly driving away from its target. Although this example is very high-level, it makes it clear that stability analyses must incorporate switching.

The simultaneous use of multiple identification models for control was initially researched in the 1960s. Some early authors were Magill [3], Lainiotis [4], and Athans *et al.* [5]. They used multiple Kalman filters to improve accuracy of state estimation. Prior to 1995, this research was applied to target tracking [6], cardiac monitoring [7], air traffic control tracking [8], and failure detection [9]. With this method, sta-

Figure 1.3: The “car in the desert” switching example.

bility analyses were never performed and all the models’ estimates were merged to generate the control signal. Martensson [10] proposed switching for adaptive control in 1986. Early multiple model adaptive control schemes with switching used fixed model representations of the unknown plant. This was considered adaptive control because the model selection was made on-line. Narendra *et al.* have since provided many contributions to discrete-time multiple model control of linear systems [1, 11, 12, 13, 14, 15, 16, 17]. The highlight of this work was introduced in 1992 and was called multiple model switching and tuning (MMST) [14].

Extensions were made in 2000 [11] to control stochastic discrete-time systems. Because some parameter estimation algorithms are structurally different, each can be more suitable to a slightly different stochastic problem. In 2003 [12], MMST selects, on-line, the adaptive model (algorithm) that results in the best performance for a given disturbance. A more thorough discussion of MMST is given in the main body of this document, where it is given its own chapter.

This MMST methodology is similar to the Markovian Jump Linear Systems (MJLS) control methodology, where a recent research example is [18]. In MJLS,

the plant's parameters change with respect to an underlying Markov Chain. As a result, there must be *a priori* knowledge of when and how to switch. The MMST technique operates without requiring such switching information.

The MMST method employs a performance index switching technique for on-line model selection. One particular performance index is well-researched and popular [1, 15, 16, 19, 20] because of its flexibility and ability to select models in stochastic environments. This performance index is discussed later in this document, where it is given its own section.

A hysteresis switching logic is proposed by Hespanha *et al.* [19, 21, 22]. Analysis is given and the authors indicate successful implementation in the deterministic case. The switching method is not proven useful with stochastic disturbances or designed for use with discrete event systems. As a result, further analysis of the hysteresis-based switching method is not in the scope of this thesis.

Discrete-time is suitable for adaptive control because it allows for easy analysis of stochastic disturbances. Also, computer systems, which are inherently discrete, are used for most applications of adaptive control. Narendra *et al.* found discrete-time systems most appropriate for research because they aimed to apply their MMST to control nonlinear systems using artificial neural networks. Their most recent research in multiple model neural networks [13] makes discrete-time a necessity.

1.1.2 Hypothesis Test Parameter Jump Detection

In statistics research, parameter jumps are referred to as “the problem of switching statistics.” Kalman filtering and a hypothesis testing method are employed to detect power system faults in [23]. This approach assumes that a discrete Kalman filter is operating prior to any system fault (parameter jump) such that the sequence of

residuals in the filter equation is zero mean Gaussian. Each residual, $\tilde{Z}_k$, is scalar and normalized to yield a standard normal random number. This normalized residual, $\tilde{Z}_{nk}$, is used as the test statistic for the hypothesis test. A mathematical overview of how the normalized residual is calculated is shown below by (1.1)-(1.5).

The Kalman residual, $\tilde{Z}_k$, is given by the equation

$$\tilde{Z}_k = Z_k - H_k^T \hat{X}_{k-1}, \quad (1.1)$$

where Z_k is the observation, H_k is the known observation matrix, and $\hat{X}_k$ is the recursive state estimate. Using the Kalman gain, K_k , the state estimate, $\hat{X}_k$, is calculated using the equation

$$\hat{X}_k = \hat{X}_{k-1} + K_k \tilde{Z}_k. \quad (1.2)$$

The error covariance, Q_k , is updated recursively in the filter equations and is defined as

$$Q_k = E \left[(X_k - \hat{X}_k) (X_k - \hat{X}_k)^T \right], \quad (1.3)$$

where $E[\cdot]$ denotes conditional expectation. The covariance matrix of residual, $\tilde{Z}_k$, is represented as

$$Q_{\tilde{Z}_k} = E \left[\tilde{Z}_k \tilde{Z}_k^T \right] = H_k^T Q_{k-1} H_k + R_k, \quad (1.4)$$

and the Kalman residual, $\tilde{Z}_k$, is normalized as

$$\tilde{Z}_{nk} = (Q_{\tilde{Z}_k})^{-1/2} \tilde{Z}_k. \quad (1.5)$$

It is assumed that a jump in parameters will yield a biased residual.

The null hypothesis, H_0 , and alternative, H_1 , are declared as follows:

H_0 : No parameter jump has occurred.

H_1 : jump has occurred.

Because $\tilde{Z}_{nk}$ is a standard normal random number, the normal distribution tables are used to determine a rejection threshold. Using T_r , the following binary decision rule is used:

1. H_0 is rejected if $|\tilde{Z}_{nk}| > T_r$.
2. H_0 is accepted if $|\tilde{Z}_{nk}| \leq T_r$.

In the event that a jump is detected, the filter is re-initialized to improve convergence to the new parameter values. The possibility of using an n -sample test is considered in [23]. The n -samples represent the number of time steps that the null hypothesis can be rejected before the filter is actually re-initialized. Using a value $n > 1$ allows for more accurate detection but single-sample testing (*i.e.* $n = 1$) yields the fastest possible detection. Experimental results in [23] show that performance is best when single-sample testing is implemented.

In [24] the previously discussed Kalman filtering and hypothesis testing method is further developed into what is called the Combined Filter Algorithm (CFA). The null hypothesis and alternative hypothesis, used in [23], are used with the CFA, but a three-state decision rule is used. An acceptance threshold, T_a , and rejection threshold, T_r , (both obtained from statistical tables) are used to make the following ruling:

1. reject H_0 if $|\tilde{Z}_{nk}| > T_r$.
2. accept H_0 if $|\tilde{Z}_{nk}| < T_a$.

3. no decision if $T_a \leq |\tilde{Z}_{nk}| \leq T_r$.

When the null hypothesis, H_0 , is rejected a parameter jump is assumed to have occurred and the applied Kalman filter is re-initialized to improve convergence to the new parameter values. If H_0 is accepted, the parameters are assumed to be constant and the Kalman filter uses all past data available to perform estimates. If no accept/reject decision can be made, the parameters are assumed to be slowly time-varying and the Kalman filter is implemented to have a fading memory.

1.2 Thesis Overview

In the remainder of **Chapter 1**, the research contributions are highlighted and the control problem is stated.

Chapter 2 provides a review of how stochastic discrete-time single model control is performed. Justifications are made for using the extended least squares parameter estimation algorithm. The certainty equivalence principle, for generating control signals, is defined. A stability analysis and simulations are included.

Chapter 3 gives detail to how stochastic discrete-time multiple model adaptive control is performed. A very popular and flexible switching method is reviewed. The most commonly used model arrangement is presented as the classical model arrangement. Then a modified model arrangement is proposed. Extensive analyses are given to show the stability benefits given by the new model arrangement. Numerous simulations are then performed to show the differences between the classical and modified model arrangements.

Chapter 4 proposes the hypothesis test switching method. The stability of the hypothesis test switching method is analyzed when implemented with either the clas-

sical or modified model arrangement. Simulations are then used to show the many benefits gained by the hypothesis test switching method.

Finally, **Chapter 5** summarizes the thesis, presents conclusions, and recommends future research.

Following the main document are references and appendices. **Appendix A** contains many of the mathematical derivations and concepts used. **Appendix B** provides all of the simulation code used to generate the results achieved.

1.3 Contribution of the Thesis

This thesis demonstrates the fusion of two concepts in switching systems, namely, hypothesis testing and multiple model adaptive control. Separately a modification to the most commonly used model arrangement is proposed to give improvements to both, performance and stability. Also, a quantitative reporting method is proposed and demonstrated useful for analyzing control performance during parameter jumps.

1.3.1 Hypothesis Test Switching

A hypothesis test switching method is defined to detect parameter jumps in a stochastic environment and perform model selection. This method is intended to operate optimally, without user adjustment or *a priori* knowledge of plant parameter behavior and model placement. A performance index switching method is chosen for comparison to the proposed hypothesis test method. This performance index is well-researched and popular [1, 15, 16, 19, 20] because of its flexibility and ability to select models in stochastic environments. The performances of the two different switching methods are compared in Chapter 4. In Section 4.4.6, one benefit of the hypothesis

test switching method is demonstrated to be ease-of-implementation. Simulations show that the method can be implemented without *a priori* knowledge of the control environment and the multiple model controller will reliably yield small control error in the presence of parameter jumps. In contrast, the performance index switching method does not possess this ease-of-implementation attribute and requires repetitive tuning to achieve acceptable control error. For this reason, the performance index is a *heuristic* switching method. It is then demonstrated in Section 4.4.7 that, in general, the hypothesis test switching method yields lower control error than the performance index method.

1.3.2 Modified Model Arrangement

In this thesis, the commonly used model arrangement [1, 11, 12, 13, 14, 15, 16, 17, 19, 20] is referred to as the *classical model arrangement*. It is found to be, by far, the most popular way to implement more than one model and controller with a plant. Stability analysis for this model arrangement requires assumptions that restrict both switching performance and the permissible size of the disturbance present.

This thesis proposes a major modification to this well-researched model arrangement. This *modified model arrangement* decouples stability and performance, for any applied switching method. The stability argument does not require analysis of the applied switching logic because it is intrinsic that an adaptive model will be selected. This is not the case for the classical model arrangement. For this reason, the stability argument for the modified arrangement is considerably simpler. Also, the assumptions necessary for the stability of the classical arrangement are no longer necessary. New and existing switching methods, that would compromise stability of the classical arrangement, can be applied to the modified arrangement. The two model

arrangements' stabilities are analyzed and compared in Section 3.5.

The simulated behaviors and performances of the two model arrangements are compared in Section 3.6. The proposed modifications to the classical model arrangement are shown to cause no detrimental performance. And, should the switching method be improperly implemented, the modified model arrangement gives far superior performance.

When compared to the classical model arrangement, the computational complexity is the same, stability is proven more easily, and performance increases are realized.

1.3.3 Quantitative Method to Analyze Performance During Multiple Parameter Jumps

In the control simulations presented in this thesis, the plant has multiple parameter jumps. Repetitive tuning of the applied switching method is performed to achieve the lowest control error for a given control problem. Due to the number of simulations performed, it would take hundreds of pages to plot each simulation's control error and perform qualitative analysis and make comparisons between plots. A modified MSE calculation is proposed to analyze the control error during these parameter jumps. The method employs a user-defined threshold to identify, and apply a ceiling to outliers in the control error. The resulting control error MSE is without bias from extraordinarily large transient samples. Using a larger threshold places more quantitative value on the magnitude of transient error. Using a smaller threshold places more value on the speed at which the transient error converges. This quantitative reporting is proven useful for comparing the control error of simulations containing multiple jumps in parameters. More information regarding the modified MSE method

is given in Appendix A.1.

1.4 Statement of the Problem

A discrete-time, unity-delay, single-input-single-output plant is described by the difference equation

$$\begin{aligned} y(k+1) = & -a_1(k)y(k) - \cdots - a_{n_A}(k)y(k-n_A) \\ & + b_1(k)u(k) + \cdots + b_{n_B}(k)u(k-n_B+1) \\ & + c_1(k)w(k) + \cdots + c_{n_C}(k)w(k-n_C+1) + w(k+1) \end{aligned} \quad (1.6)$$

where $u(k)$ is the input, $y(k)$ is the output, and $w(k)$ is the disturbance. The terms n_A , n_B , and n_C represent the order of the plant's outputs, inputs, and disturbance, respectively.

The disturbance, $w(k)$, is zero mean Gaussian noise with a variance denoted by σ^2 . Equation (1.6) can also be represented as

$$y(k+1) = \theta^T(k)\phi_0(k) + w(k+1), \quad (1.7)$$

where the plant parameter vector, $\theta^T(k)$, and regression vector, $\phi_0(k)$, are defined as

$$\theta^T(k) = [a_1(k), \cdots, a_{n_A}(k), b_1(k), \cdots, b_{n_B}(k), c_1(k), \cdots, c_{n_C}(k)] \quad (1.8)$$

and

$$\phi_0(k)^T = [-y(k), \dots, -y(k - n_A), u(k), \dots, u(k - n_B + 1), w(k), \dots, w(k - n_C + 1)]. \quad (1.9)$$

Equations (1.6), (1.7), and (1.9) are only useful for theoretical analysis, as the disturbance signal, $w(k + 1)$, cannot be measured.

The plant can also be represented using the autoregressive moving average exogenous variables (ARMAX) model as

$$A(q^{-1})y(k) = B(q^{-1})u(k) + C(q^{-1})w(k), \quad (1.10)$$

where

$$A(q^{-1}) = 1 + a_1q^{-1} + \dots + a_{n_A}q^{-n_A},$$

$$B(q^{-1}) = b_1q^{-1} + \dots + b_{n_B}q^{-n_B},$$

$$C(q^{-1}) = 1 + c_1q^{-1} + \dots + c_{n_C}q^{-n_C},$$

and q^{-1} represents the backward shift operator.

Assumptions:

(A1) Using the ARMAX model, it is stated that plant polynomials $A(q^{-1})$, $B(q^{-1})$, and $C(q^{-1})$ are all relatively prime. In addition, $A(q^{-1})$ is monic (*i.e.* the coefficient of the highest order term is 1), $B(q^{-1})$ is stable (*i.e.* all zeros are inside the unit circle), $C(q^{-1})$ is monic, stable, and $[1/C(q^{-1}) - \frac{1}{2}]$ is strictly

positive real (SPR). This requires that the poles of $[1/C(q^{-1}) - \frac{1}{2}]$ are inside the unit circle (conditions to be SPR are further discussed in Appendix A.2).

- (A2) The order of the system is known (*i.e.* n_A , n_B , and n_C). This allows a parameter estimation method of sufficient order to be implemented. This also allows the reference signal to excite the system with sufficient order. Persistency of excitation is defined in Appendix A.3. In Section 2.5.4 the effects of insufficient excitation are demonstrated.
- (A3) The plant parameter b_1 , in (1.6), is greater than a positive lower bound, b_{min} . This condition assures that the certainty equivalence control signal is bounded, as discussed in Section 2.3.

Assumptions which can be conditionally relaxed:

- (A4) The plant's transfer function is minimum phase (*i.e.* the roots and poles are located inside the unit circle, $|z| = 1$).
- (A5) The plant parameter vector, $\theta^T(k)$, is time-invariant (*i.e.* it can be denoted θ^T).

Assumptions (A4) and (A5) simplify the stability analyses and are relaxed in many of the simulations. Explanation is given when the assumptions are relaxed. The plant parameter vector, $\theta^T(k)$, can change with time, either slowly and continuously or in a more abrupt, piece-wise constant, manner. The case is considered when parameter jumps result in unstable plant configurations.

The desired plant output, $y^*(k+1)$, is bounded and assumed to be known at time-step k . The control error, $e_c(k+1)$, is defined as

$$e_c(k+1) = y(k+1) - y^*(k+1) \quad (1.11)$$

The objective of the control problem is to determine a bounded input, $u(k+1)$, such that the output, $y(k+1)$, is bounded and tracks $y^*(k+1)$ with minimum variance.

These requirements for being bounded are stated analytically as

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N u^2(k) < \infty \quad (1.12)$$

and

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N y^2(k) < \infty. \quad (1.13)$$

The minimum variance control error is expressed using the mean or conditional expectation, $E[\cdot]$, of the control error, $e_c(k+1)$, given the data available at time step $k-1$,

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N E[e_c(k)|k-1]^2 = \sigma^2, \quad (1.14)$$

where σ^2 is the variance of the disturbance, $w(k)$, acting on the plant (1.6).

Chapter 2

Stochastic Single Model Adaptive Control

2.1 Introduction

Single model discrete-event control is the foundation on which multiple model control is defined in this thesis. As shown in Figure 1.1, a single estimation model uses the plant input and output signals to recursively estimate the plant's parameters. At every time-step, k , the certainty equivalence principle uses the most recent plant parameter estimates, $\hat{\theta}(k)$, to generate a plant control input, $u(k)$. The objective of the control problem is to make the plant output, $y(k)$, track a reference signal, $y^*(k)$, ensuring that the control objective and conditions described by (1.12)-(1.14) are satisfied.

The extended least squares (ELS) algorithm is chosen for online parameter estimation in this thesis. Arguments for using the ELS algorithm and a description of the other popular parameter estimation methods are given in Section 2.2.1. The

certainty equivalence control principle is defined in Section 2.3.

2.2 Parameter Estimation

2.2.1 Selection of Algorithm

There is no single estimation algorithm considered to be most suitable for all stochastic problems. The recursive least squares (RLS), extended least squares (ELS), output error with extended prediction model (OEEPM), and recursive maximum likelihood (RML) methods are each suitable for online parameter estimation of a stochastic linear system. The methods are recursive and have comparable computational efficiencies. The ELS, OEEPM, and RML algorithms are structurally different. In [12], it is shown that, as a result of their differences, each algorithm is most suitable to a slightly different stochastic problem.

The ELS algorithm is chosen for the control problem stated in Section 1.4. As discussed later in this chapter, the ELS algorithm has superior parameter estimation properties when compared to the RLS algorithm. The ELS algorithm is frequently used in stochastic adaptive control literature [11, 12, 26, 27, 28]. Because the ELS algorithm is an extension from the RLS algorithm, it is worth first discussing the least squares (LS) and RLS algorithms.

The LS algorithm was first proposed by Karl Gauss and it has become a very popular method for estimating unknown parameters from experimental data. The algorithm operates on the assumption that the parameters to be estimated are time-invariant. A matrix inversion is necessary to perform the parameter estimation. The dimension of this matrix is directly related to the number of new measurements taken. Thus, if new data were always arriving, the inversion process would become progres-

sively more time consuming. Intuitively, a computationally recursive implementation of LS, was developed. This RLS algorithm is also referred to as a sequential or online estimation method. Using RLS, the computational load remains constant while more data are arriving. The newest parameter estimate can be produced using the previous RLS estimate and just the newest arrival of data. As a result, this method has strong appeal for online parameter estimation and control applications. Plant parameter estimates are generated using plant inputs and outputs as experimental data. This parameter estimation method is extensively studied and frequently used for adaptive control.

Extensions to the RLS and ELS methods were developed to consider the case when plant parameters are time-varying. Two such methods are discussed in the following two paragraphs.

For the case when the parameters to be estimated are changing abruptly but infrequently, it is instinctive to periodically *reset* the covariance matrix, denoted by P [31]. Resetting the covariance matrix, P , involves setting $P = aI$, where a is a large number and I is an identity matrix. A large number is used for a to cause large changes in the next parameter estimate updates. Resetting is not an ideal parameter tracking solution because there is no way of knowing when to or how often to reset. Resetting “too often” will cause unnecessary estimation errors and resetting “not often enough” allows large periods of time where parameter changes are undetected.

For the case when the parameters being estimated are changing slowly and continuously, a method called exponential forgetting is used [11, 12, 17, 31]. In order to discount old data and make the new data of greater significance, a user-defined scalar forgetting factor is used to affect the recursive update procedure. A drawback to the exponential forgetting method is revealed when the parameter estimation process

is improperly excited. Because of the constant discounting of old data, it must be guaranteed that the new data contain sufficient correlation for estimation of all parameters. This condition, called *persistence of excitation*, is further defined in Appendix A.3. In the extreme case, when the plant input is unchanging, the covariance matrix, P , of the parameter estimation algorithm will grow exponentially. Upon the arrival of the first unique control input, the parameter estimates can become immediately unbounded (*i.e.* estimates are updated using a considerably overcompensating step-size). This undesirable effect, referred to as estimator windup, has been the focus of much investigation [29, 30, 31].

The ELS parameter estimation algorithm extends from RLS to use, in addition to control inputs and plant outputs, past estimation errors. As a result, any correlation in the estimation error can be parameterized. Correlation in estimation error occurs when the system is subject to a coloured Gaussian disturbance. Using ELS, this estimation error becomes asymptotically white because it is parameterized. A comparison of the RLS and ELS algorithms is performed using simulations in Section 2.5.2.

ELS parameter estimation with exponential forgetting is ideal for adaptive control of a plant with time-varying parameters and disturbances. Estimator windup is avoided if both the disturbance and control inputs are persistently exciting. The concept of estimator windup and the requirements for persistence of excitation are further discussed when demonstrated in Section 2.5.4.

2.2.2 Definition of the Extended Least Squares Algorithm

The plant parameter vector, $\theta^T(k)$ from Section 1.4, is defined as

$$\theta^T(k) = [a_1(k), \dots, a_{n_A}(k), b_1(k), \dots, b_{n_B}(k), c_1(k), \dots, c_{n_C}(k)]. \quad (2.1)$$

Terms n_A , n_B , and n_C represent the order of the plant's outputs, inputs, and disturbance, respectively. These orders are known, as stated by *Assumption (A2)*, but the values of the parameters are unknown. As a result, (2.1) is represented by a parameter estimate vector, $\hat{\theta}^T(k)$, defined as

$$\hat{\theta}^T(k) = [\hat{a}_1(k), \dots, \hat{a}_{n_A}(k), \hat{b}_1(k), \dots, \hat{b}_{n_B}(k), \hat{c}_1(k), \dots, \hat{c}_{n_C}(k)]. \quad (2.2)$$

The noise components, $w(k), \dots, w(k - n_C + 1)$, in the plant's regression vector,

$$\phi_0(k)^T = [-y(k), \dots, -y(k - n_A), u(k), \dots, u(k - n_B + 1), w(k), \dots, w(k - n_C + 1)], \quad (2.3)$$

cannot be measured and are replaced by the *a posteriori* estimation errors,

$$e(k), \dots, e(k - n_C + 1),$$

defined below by (2.8). The ELS regression vector, $\phi(k)$, is then defined as

$$\phi(k)^T = [-y(k), \dots, -y(k - n_A), u(k), \dots, u(k - n_B + 1), e(k), \dots, e(k - n_C + 1)]. \quad (2.4)$$

The *a priori* plant output estimate, $\hat{y}^0(k + 1)$, is expressed as

$$\hat{y}^0(k + 1) = \hat{\theta}^T(k)\phi(k). \quad (2.5)$$

Because $\hat{y}^0(k+1)$ is calculated using the parameter estimate vector at time-step k , it is called the *a priori* estimate of the plant output, $y(k+1)$. Using the parameter estimates at time-step $k+1$, the *a posteriori* estimate, $\hat{y}(k+1)$, is calculated as

$$\hat{y}(k+1) = \hat{\theta}^T(k+1)\phi(k). \quad (2.6)$$

The *a priori* and *a posteriori* estimation errors, $e^0(k+1)$ and $e(k+1)$, are defined as

$$e^0(k+1) = y(k+1) - \hat{y}^0(k+1) \quad (2.7)$$

and

$$e(k+1) = y(k+1) - \hat{y}(k+1). \quad (2.8)$$

The parameter estimate vector, $\hat{\theta}^T(k+1)$, is the first calculation to be performed at time-step $k+1$. Using the newest *a posteriori* estimation error, which is from the previous time-step, k , the parameter estimation vector is updated as

$$\hat{\theta}^T(k+1) = \hat{\theta}^T(k) + P(k+1)\phi(k)e(k). \quad (2.9)$$

The initialization of the parameter estimate vector, $\hat{\theta}^T(0)$, is known and the covariance matrix, $P(k+1)$, is calculated as

$$P(k+1) = \frac{1}{\lambda} \left(P(k) - \frac{P(k)\phi(k)\phi^T(k)P(k)}{\lambda + \phi^T(k)P(k)\phi(k)} \right), \quad (2.10)$$

where the user-defined forgetting factor, $\lambda \in (0, 1]$, specifies how quickly the algorithm discounts past sample information and is more thoroughly discussed in Section 2.2.1. The process of selecting a suitable value for λ is demonstrated in Section 2.5.3.

Assuming that the input, $u(k)$, is persistently exciting of order $n_A + n_B$, the estimated parameter vector, $\hat{\theta}^T(k)$, will converge asymptotically to the true parameter vector, $\theta^T(k)$. A definition of persistency of excitation is provided in Appendix A.3. In Section 2.5.4, it is demonstrated that estimator windup, discussed in Section 2.2.1, can be avoided if all the parameter estimates are properly excited. As a consequence, both the system inputs, $u(k)$ and $w(k)$, are required to be persistently exciting.

2.3 Control Using Certainty Equivalence

Generally, the certainty equivalence principle implements a decision policy by putting the uncertainty equal to its expected value. In this thesis, the certainty equivalence principle equates the desired plant output, $y^*(k+1)$, with the estimated plant output, $\hat{y}(k+1)$. The objective is to determine a bounded input, $u(k)$, such that the plant output, $y(k+1)$, is bounded and tracking the reference, $y^*(k+1)$, with minimum variance.

The control error is defined as

$$e_c(k+1) = y(k+1) - y^*(k+1), \quad (2.11)$$

which is rearranged to be

$$y(k+1) = y^*(k+1) + e_c(k+1). \quad (2.12)$$

The ELS plant output estimate,

$$\hat{y}(k+1) = \hat{\theta}^T(k+1)\phi(k),$$

is equated with the desired plant output, $y^*(k+1)$, and it follows that

$$\begin{aligned}
y(k+1) &= \hat{y}(k+1) + e_c(k+1), \\
&= \hat{\theta}^T(k+1)\phi(k) + e_c(k+1), \\
&= \hat{b}_1(k+1)u(k) + \hat{\theta}^T(k+1)\tilde{\phi}(k) + e_c(k+1),
\end{aligned} \tag{2.13}$$

where

$$\tilde{\phi}(k)^T = [-y(k), \dots, -y(k-n_A), 0, \dots, u(k-n_B+1), e(k), \dots, e(k-n_C+1)]. \tag{2.14}$$

The regression vector, $\tilde{\phi}(k)$, is the same as $\phi(k)$ from (2.4), except that $u(k)$ is replaced by a zero. With the certainty equivalence principle, the control error, $e_c(k)$, is equal to the *a posteriori* estimation error, $e(k)$. The control input, $u(k)$, is calculated by rearranging (2.13) to be

$$u(k) = \frac{y^*(k+1) - \hat{\theta}^T(k+1)\tilde{\phi}(k)}{\hat{b}_1(k+1)}. \tag{2.15}$$

To ensure that $u(k)$ is bounded, the ELS algorithm is modified to adjust the denominator parameter estimate, $\hat{b}_1(k+1)$, such that

$$\hat{b}_1(k+1) \geq b_{min} > 0. \tag{2.16}$$

When updating the parameter estimate vector, $\hat{\theta}^T(k+1)$, if the estimated parameter $\hat{b}_1(k+1)$ is less than b_{min} , it is set to $\hat{b}_1(k+1) = b_{min}$, so that condition (2.16) always holds.

It appears intuitive to modify (2.16) to encompass the more broad condition, $|\hat{b}_1(k+1)| \geq b_{min} \geq 0$, but this introduces a complication. Should the plant parameter b_1 jump from a positive value to a negative value, then the estimate, $\hat{b}_1$, will adapt and pass through zero. To avoid this complication, *Assumption (A3)*, in Section 1.4, states that the plant parameter b_1 is also bounded such that $b_1(k+1) \geq b_{min} \geq 0$. The simulations performed in this thesis apply this condition using $b_{min} = 0.001$.

2.4 Stability

Because this thesis focuses on contributing to multiple model adaptive control, the arguments for stability of single model control will not be thoroughly derived. It is important to note that the stability of single model control is central to arguments made for the multiple model case in Section 3.5.

As discussed in Section 1.4, by *Assumption (A1)*, the transfer function

$$\frac{1}{C(q^{-1}) - \frac{1}{2}}$$

is SPR. The definition of SPR is further discussed in Appendix A.2. The SPR condition is satisfied when using the forgetting factor value $\lambda = 1$ in the covariance matrix, $P(k)$, given by (2.10).

As given by *Assumption (A5)*, the plant parameter vector, θ^T , is time-invariant. This is necessary to simplify the proof of stability.

Two lemmas are required before proving stable adaptive control. Lemma 1 states how parameter estimates are guaranteed to converge and Lemma 2 states the same for the stochastic case.

Lemma 1 - Parameter convergence for the deterministic case:

Given the time-invariant plant θ from (2.1), ELS parameter estimates $\hat{\theta}^T(k)$ from (2.9), and the covariance matrix $P(k)$ with $\lambda = 1$ from (2.9), the following results are achieved:

$$\|\hat{\theta}(k) - \theta\| \leq m_1 \|\hat{\theta}(0) - \theta\|, \quad \forall k \geq 1, \quad (2.17)$$

$$\lim_{k \rightarrow \infty} \sum_{\tau=1}^k \frac{e^2(\tau)}{1 + \phi^T(\tau-1)\phi(\tau-1)} < \infty, \quad (2.18)$$

$$\lim_{k \rightarrow \infty} \frac{e(k)}{[1 + \phi^T(k-1)\phi(k-1)]^{1/2}} = 0, \quad (2.19)$$

$$\lim_{k \rightarrow \infty} \|\hat{\theta}(k) - \hat{\theta}(k - m_2)\| = 0, \quad (2.20)$$

for any finite m_1 and m_2 .

Proof of Lemma 1 is given in the textbook “Adaptive Filtering, Prediction and Control” [25].

Lemma 1 is the foundation on which all further stability is based in this document. The parameter estimate error is bounded as described by (2.17). Even if the regression vector grows in an unbounded fashion, (2.19) implies that the estimation error will grow at slower rate. The parameter values converge, as defined by (2.20), and become near constant.

Lemma 2 - Parameter convergence for the stochastic case:

The relationship between the magnitudes of the (2.7) *a priori* estimation error, $e^0(k)$, and the (2.4) regression vector, $\phi(k)$, is defined as

$$\lim_{k \rightarrow \infty} \sum_{k=1}^N \frac{(e^0(k) - w(k))^2}{r(k-1)} < \infty, \quad (2.21)$$

where $r(k-1)$ is the sum-of-squares of the norms of the regression vector. Using the initial condition $r(k) = 0$ at time-step $k = 0$, $r(k-1)$ is defined as

$$r(k-1) = r(k-2) + \phi^T(k-1)\phi(k-1) = \sum_{\tau=0}^{k-1} \phi^T(\tau)\phi(\tau). \quad (2.22)$$

Equation (2.22) is positive and monotonically increasing. The term monotonically increasing is defined in Appendix A.4.

The relationship between the magnitudes of the (2.8) *a posteriori* estimation error, $e(k)$, and the regression vector, $\phi(k)$, is defined as

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N \frac{(e(k) - w(k))^2}{r(k-1)} < \infty. \quad (2.23)$$

And the asymptotic convergence of the ELS parameter estimate vector can be stated analytically as

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N \left\| \hat{\theta}(k) - \hat{\theta}(k-m) \right\|^2 < \infty, \quad (2.24)$$

for any finite m .

Proof of Lemma 2 is given in textbook “Adaptive Filtering, Prediction and Control” [25].

Reference [12] provides the following proof of stable single model adaptive control. This proof uses the Kronecker lemma, which is included in Appendix A.7.

It is assumed that $r(k-1)$, from (2.22), is bounded, then (2.23) is bounded as follows

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N (e(k) - w(k))^2 < \infty. \quad (2.25)$$

Next, it is assumed that $r(k-1)$ is unbounded and monotonically increasing. The Kronecker lemma (in Appendix A.7) is necessary for further analysis. The term $r(k-1)$ suits the necessary requirements to be term b_k , which is defined within the Kronecker lemma. The terms b_k and x_k are equated to the following terms from (2.23):

$$b_k : r(k-1).$$

$$x_k : \frac{(e(k) - w(k))^2}{r(k-1)}.$$

The Kronecker lemma gives the result,

$$\lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N b_k x_k = 0, \quad (2.26)$$

thus, (2.23) becomes

$$\lim_{N \rightarrow \infty} \frac{1}{r(N-1)} \sum_{k=1}^N (e(k) - w(k))^2 = 0. \quad (2.27)$$

For use later in this proof, (*i.e.* (2.38)) equation (2.27) can be represented as

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N \frac{(1/N)(e(k) - w(k))^2}{r(N-1)/N} = 0. \quad (2.28)$$

The *a posteriori* ELS estimate, $\hat{y}(k+1)$, is the one-step-ahead estimation of the plant output, $y(k+1)$. The ELS estimate, calculated

$$\hat{y}(k+1) = \hat{\theta}^T(k+1)\phi(k), \quad (2.29)$$

can be represented using conditional expectation as

$$\hat{y}(k+1) = E[y(k+1)|k]. \quad (2.30)$$

Using the certainty equivalence principle, defined by (2.15), the minimum variance control error, $e_c(k)$, is bounded and equal to the estimation error, $e(k)$. The desired output, $y^*(k)$, and noise signal, $w(k)$, are mean-square bounded (given in Section 1.4), and it follows that

$$\frac{1}{N} \sum_{k=1}^N y^2(k) \leq \frac{\alpha_1}{N} \sum_{k=1}^N (e_c(k) - w(k))^2 + \alpha_2, \quad (2.31)$$

where $0 < \alpha_1, \alpha_2 < \infty$. This states analytically that the plant output, $y(k)$, and minimum variance control error, $[e_c(k) - w(k)]$, grow at the same, or at a comparable, rate. Equation (2.31) is intuitive because the desired plant output, $y^*(k)$, is defined in Section 1.4 to be bounded, and $e_c(k) = y(k) - y^*(k)$.

Because the plant is minimum phase (*i.e.* the polynomial $B(q^{-1})$ of (1.10) is stable), it follows that

$$\frac{1}{N} \sum_{k=1}^N u^2(k) \leq \frac{\alpha_3}{N} \sum_{k=1}^N y^2(k) + \alpha_4, \quad (2.32)$$

where $0 < \alpha_3, \alpha_4 < \infty$. This analytically states that the signals $u(k)$ and $y(k)$ grow at the same, or at a comparable, rate.

The *a priori* plant output estimate, $\hat{y}^0(k)$ from (2.5), a *posteriori* plant output estimate, $\hat{y}(k)$ from (2.6), and plant actual output, $y(k)$ from (2.29), also grow at identical or comparable rates. This is stated analytically using the relationships:

$$\frac{1}{N} \sum_{k=1}^N \hat{y}^2(k) \leq \frac{\alpha_5}{N} \sum_{k=1}^N y^2(k) + \alpha_6, \quad (2.33)$$

$$\frac{1}{N} \sum_{k=1}^N y^2(k) \leq \frac{\alpha_7}{N} \sum_{k=1}^N \hat{y}^2(k) + \alpha_8, \quad (2.34)$$

$$\frac{1}{N} \sum_{k=1}^N [\hat{y}^0]^2(k) \leq \frac{\alpha_9}{N} \sum_{k=1}^N y^2(k) + \alpha_{10}, \quad (2.35)$$

$$\frac{1}{N} \sum_{k=1}^N y^2(k) \leq \frac{\alpha_{11}}{N} \sum_{k=1}^N [\hat{y}^0]^2(k) + \alpha_{12}, \quad (2.36)$$

where $0 < \alpha_{5,\dots,12} < \infty$.

By definition of the regression vector, and observing (2.31)-(2.36), the regression vector also grows at a rate that is identical or comparable to the rate of the plant output, $y(k)$. Thus, $r(k-1)$, the sum-of-squares of the norms of the regression vector grows in the same manner,

$$\frac{r(N-1)}{N} \leq \frac{\alpha_{13}}{N} \sum_{k=1}^N (e_c(k) - w(k))^2 + \alpha_{14}, \quad (2.37)$$

where $0 < \alpha_{13}, \alpha_{14} < \infty$. The term $r(N-1)$ is defined by (2.22). By definition of the certainty equivalence principle, $e_c(k) = e(k)$ and (2.28) can be represented as

$$\lim_{N \rightarrow \infty} \frac{(1/N)}{r(N-1)/N} \sum_{k=1}^N (e_c(k) - w(k))^2 = 0. \quad (2.38)$$

Equations (2.37) and (2.38) are contradicting. A very informal analysis is used to make the contradiction more clear. The terms $L_1(N)$ and $L_2(N)$ are equated as follows:

$$L_1(N) : \quad \frac{r(N-1)}{N}.$$

$$L_2(N) : \quad \frac{1}{N} \sum_{k=1}^N (e_c(k) - w(k))^2.$$

Equations (2.37) and (2.38) become

$$L_1(N) \leq \alpha_{13}L_2(N) + \alpha_{14}, \quad (2.39)$$

where $0 < \alpha_{13}, \alpha_{14} < \infty$, and

$$\lim_{N \rightarrow \infty} \frac{L_2(N)}{L_1(N)} = 0, \quad (2.40)$$

respectively. It is clear that (2.37) and (2.38) are contradicting. These equations were achieved assuming that $r(N-1)$ is unbounded. Thus, this case cannot occur and $r(N-1)/N$ must be bounded. The bounded case was explored above and resulted in (2.25). The single model controller converges as follows

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N (e_c(k) - w(k))^2 = 0. \quad (2.41)$$

From (2.31) and (2.33) it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N y^2(k) < \infty \quad (2.42)$$

and

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N u^2(k) < \infty. \quad (2.43)$$

Equations (2.42) and (2.43) are identical to (1.12) and (1.13) in Section 1.4.

Note: The minimum variance of $e_c(k)$ is expressed by (1.14) in Section 1.4, using conditional expectation, as

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N \mathbb{E}[e_c(k)|k-1]^2 = \sigma^2, \quad (2.44)$$

where σ^2 is the variance of the disturbance, $w(k)$, acting on the plant (1.6). And note that the control error equation,

$$y(k) - y^*(k) = e_c(k), \quad (2.45)$$

can be expressed, using conditional expectation, as

$$E[(y(k) - y^*(k))^2 | k-1] = E[(e_c(k) - w(k))^2 | k-1] + E[w^2(k) | k-1]. \quad (2.46)$$

Using (2.41), it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N E[e_c^2(k) | k-1] = \sigma^2. \quad (2.47)$$

Equations (2.47) and (1.14) are identical. The exact goal of the adaptive control problem, as stated by (1.12)-(1.14) in Section 1.4, is achieved.

2.5 Simulations

The key concepts of Chapter 2 are demonstrated using simulation. The conditions common to all simulations are given in Section 2.5.1. In Section 2.5.2, the performances of the RLS and ELS parameter estimation methods are compared. In Section 2.5.3, experimentation is performed to find a forgetting factor, λ , suitable for use with the stochastic conditions considered throughout the thesis. A demonstration of estimator windup, which occurs when input signals are not persistently exciting, is given in Section 2.5.4.

2.5.1 Conditions Common to All Simulations

The stochastic plant to be controlled is described using the difference equation

$$\begin{aligned} y(k+1) = & a_1(k)y(k) + a_2(k)y(k-1) + b_1(k)u(k) + b_2(k)u(k-1) \\ & + c_1(k)w(k) + c_2(k)w(k-1) + w(k+1), \end{aligned} \quad (2.48)$$

where $w(k)$ is zero mean Gaussian noise with variance, σ^2 . The actual values for the plant parameter vectors and the variance, σ^2 , are uniquely defined in Section 2.5.2, 2.5.3, and 2.5.4. The plant's corresponding parameter vector, $\theta^T(k)$, is defined as

$$\theta^T(k) = [a_1(k) \quad a_2(k) \quad b_1(k) \quad b_2(k) \quad c_1(k) \quad c_2(k)]. \quad (2.49)$$

The desired plant output is described by

$$y^*(k+1) = \sin(4\pi k/50) + \sin(4\pi k/150), \quad (2.50)$$

which clearly has the initial value $y^*(0) = 0$. The RLS and ELS algorithm are implemented to begin with initial plant output estimate $\hat{y}_{RLS}(0) = 0$ and $\hat{y}_{ELS}(0) = 0$, respectively. In addition, the plant output has the initial value $y(0) = 0$. As a result, there is zero initial control error and estimation error, as defined by:

$$e_c(0) = y(0) - y^*(0) = 0,$$

$$e_{RLS}(0) = y(0) - \hat{y}_{RLS}(0) = 0,$$

$$e_{ELS}(0) = y(0) - \hat{y}_{ELS}(0) = 0.$$

The effects of initializing with zero error are discussed with the simulation results.

The parameter estimate vector for the RLS algorithm is defined as

$$\hat{\theta}_{RLS}^T(k) = [\hat{a}_1(k) \quad \hat{a}_2(k) \quad \hat{b}_1(k) \quad \hat{b}_2(k)], \quad (2.51)$$

with initial estimates

$$\hat{\theta}_{RLS}^T(0) = [0 \quad 0 \quad 1 \quad 0]. \quad (2.52)$$

This algorithm cannot parameterize the plant parameters $c_1(k)$ and $c_2(k)$, which are responsible for the correlation in disturbance, $w(k)$. The parameter estimate $\hat{b}_1$ is initialized as $\hat{b}_1 = 1$ to ensure that the condition for stable control using certainty equivalence, defined by (2.16), is satisfied.

The parameter estimate vector for the ELS algorithm is defined as

$$\hat{\theta}_{ELS}^T(k) = [\hat{a}_1(k) \quad \hat{a}_2(k) \quad \hat{b}_1(k) \quad \hat{b}_2(k) \quad \hat{c}_1(k) \quad \hat{c}_2(k)], \quad (2.53)$$

with initial estimates,

$$\hat{\theta}_{ELS}^T(0) = [0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0]. \quad (2.54)$$

Both algorithm's covariance matrices, $P_{RLS}(k)$ and $P_{ELS}(k)$, are respectively initialized as

$$P_{RLS}(0) = 10^5 I_4 \quad (2.55)$$

and

$$P_{ELS}(0) = 10^5 I_6. \quad (2.56)$$

These values are large to ensure a large step-size for initial parameter estimates.

Single model adaptive control is performed in Sections 2.5.2, 2.5.3, and 2.5.4 using the certainty equivalence principle, as defined by (2.15). The condition, $\hat{b}_1(k+1) \geq b_{min} \geq 0$, is enforced, with $b_{min} = 0.001$.

2.5.2 Comparison of RLS and ELS Parameter Estimation Methods

The RLS and ELS parameter estimation algorithms are independently simulated in identical control environments using the same assumptions concerning the plant. Both algorithms are implemented as described by (2.51)-(2.56) and use a forgetting factor value $\lambda = 1$.

The noise sequence, $w(k)$, is zero mean Gaussian with variance $\sigma^2=0.5$. In both simulations, the noise is generated using the same seed for random number generation. The time-invariant plant parameter vector, θ^T , is defined as

$$\theta^T = [0.6 \quad 0.2 \quad 1.0 \quad 0.5 \quad -0.9 \quad 0.9]. \quad (2.57)$$

These plant parameter values are arbitrarily chosen and satisfy *Assumptions (A1)-(A5)* in Section 1.4.

First, the behavior of each algorithm's parameter estimation process is analyzed. The convergence of the RLS and ELS parameter estimates is plotted in Figure 2.1 and 2.2, respectively. Plant parameters a_1 and a_2 are associated with the output of the plant and b_1 and b_2 are associated with the plant's input. It can be seen that the RLS parameter estimates did not converge to the actual plant parameter values. This was especially apparent for the plant output parameter estimates $\hat{b}_1$ and $\hat{b}_2$. This parameterization error was due to the correlation unaccounted for in the disturbance,

$w(k)$. The plant parameters c_1 and c_2 cause the Gaussian input, $w(k)$, to affect the plant, as a coloured Gaussian disturbance.

In Figure 2.2, it can be seen that the ELS algorithm parameterized the same disturbance sequence using parameter estimates $\hat{c}_1$ and $\hat{c}_2$. When compared to the RLS algorithm, the ELS algorithm yielded more accurate parameter estimates.

Next, the behavior of the control system is analyzed for the case when the RLS parameter estimates converge as already shown Figure 2.1. The control signal, $u(k)$, plant output, $y(k)$, and control error, $e_c(k)$, are plotted in Figure 2.3. The reference signal, $y^*(k)$, and plant output, $y(k)$, are initialized as zero. Correspondingly, there is zero initial control error ($e_c(k) = y(k) - y^*(k) = 0$). Transient control error and convergence occur once the reference signal excites the system (*i.e.* once it moves away from $y^*(0) = 0$). In Figure 2.4, the control system using ELS parameter estimation had similar transient control error.

In Figure 2.5, the RLS and ELS methods' control error, $e_c(k)$, is compared for the first 200 time-steps, k . The largest magnitude of transient control error produced using the RLS method, $e_c(k) = -8.66$, occurred at time-step $k = 5$. The ELS method's control error, $e_c(k)$, converged similarly, but with smaller magnitude. This is because of the ELS method's superior parameterization discussed above.

In addition, it is desirable to compare the algorithms' control error produced after the parameter estimates converge (steady-state). By observing Figures 2.1 and 2.2, it can be said that the RLS and ELS algorithms both had sufficiently converged at time-step $k = 400$. The mean control error and MSE of the control error is calculated for the period of time-steps after this, $400 \leq k \leq 1000$. The mean control error and the MSE of the control error are used as metrics for comparing steady-state performance in Table 2.1. The ELS parameter estimation method yielded a control error with

a smaller MSE and mean (zero mean Gaussian is desired). It can be said that ELS parameter estimation and control yielded preferable transient and steady-state control error.

Table 2.1: Comparing the steady-state control error, $e_c(k)$, produced by the RLS and ELS methods.

	Mean of $e_c(k)$ for $400 \leq k \leq 1000$	MSE of $e_c(k)$ for $400 \leq k \leq 1000$
Control using RLS	0.042	0.37
Control using ELS	0.037	0.30

Figure 2.1: RLS parameter estimation in the presence of a disturbance with variance $\sigma^2 = 0.5$.

Figure 2.2: ELS parameter estimation in the presence of a disturbance with variance $\sigma^2 = 0.5$.

Figure 2.3: Performing adaptive control with RLS parameter estimation and a disturbance, $w(k)$, of variance $\sigma^2 = 0.5$.

Figure 2.4: Performing adaptive control with ELS parameter estimation and a disturbance, $w(k)$, of variance $\sigma^2 = 0.5$.

Figure 2.5: Comparison of control error, $e_c(k)$, produced by both the RLS and ELS adaptive control methods.

2.5.3 Selection of Forgetting Factor, λ , for ELS in Time-Varying Systems

In this section, one value for the forgetting factor, λ , is chosen for use in all further simulations with stochastic disturbances in this document. Both, time-varying parameters and the presence of noise, make a problem stochastic. It is assumed that the plant parameters change either slowly or abruptly. This section considers the extreme case, where the plant's parameters are first constant and then abruptly jump. The parameter vector, $\theta^T(k)$, is piece-wise constant in time as follows:

$$\theta^T(k) = \begin{cases} \theta_1^T &= [0.6 \ 0.2 \ 1.0 \ 0.5 \ -0.5 \ 0.4], \quad k \leq 200, \\ \theta_2^T &= [-0.5 \ 0.2 \ 1.0 \ 0.2 \ -0.5 \ 0.4], \quad k > 200. \end{cases} \quad (2.58)$$

This plant is composed of two arbitrarily chosen stable plant states that satisfy *Assumptions (A1)-(A4)* in Section 1.4.

The one forgetting factor value must be suitable for the complete range of disturbances considered ($w(k)$ is zero mean Gaussian with variance values $\sigma^2 \in [0, 0.3]$). Testing is performed for the two extreme cases. In *Case (1)*, $\sigma^2 = 0$ (no noise) and in *Case (2)*, $\sigma^2 = 0.3$. In each case, five simulations are used to independently test five values for the forgetting factor, λ : 0.90, 0.95, 0.97, 0.98, and 0.99.

For *Case (1)*, the resulting control error, $e_c(k)$, is reported for each of the five simulations in Figure 2.6. It can be seen that each simulation's plot displays transient control error after the parameter jump at time-step $k = 200$. The axes in each plot within Figure 2.6 are fixed to make the settling time of the transient error qualitatively observable. As a result, the maximum values of the transient error cannot be seen. It is clear that the forgetting factor value $\lambda = 0.90$ yielded the shortest transient time.

The MSE of each simulation's control error is provided in Table 2.2 to allow error magnitudes to be separately analyzed.

Table 2.2: Comparing the MSE of control error, $e_c(k)$, produced using different forgetting factor values, λ , and a disturbance, $w(k)$, with variance $\sigma^2 = 0$.

	MSE of $e_c(k)$
$\lambda = 0.90$	0.01
$\lambda = 0.95$	0.03
$\lambda = 0.97$	$9.9e^+3$
$\lambda = 0.98$	$2.1e^{+10}$
$\lambda = 0.99$	$1.3e^{+13}$

Using the forgetting factor value $\lambda = 0.90$ yielded the smallest control error MSE (0.01). Using λ values 0.97, 0.98, and 0.99 yielded extremely large MSE values because of the transient control error. The large magnitudes cannot be seen because they are outside the axes. There is clearly a tradeoff when selecting a forgetting factor: reducing the transient time increases the magnitude of transient error. With half of this experimentation section complete, it can be said that selecting $\lambda = 0.97$ or 0.98 would yield an ideal compromise. This judgment is made by observing the simulations in Figure 2.7.

In *Case (2)*, the same experimental procedure is used as in *Case(1)*. The control error, $e_c(k)$, produced in each of the five simulations, is reported in Figure 2.7. Using $\lambda = 0.90$ made the parameter estimation algorithm forget “too fast,” and the control error never converged to any value. Table 2.3 is used to quantitatively compare the

simulations' control errors. The value $\lambda = 0.97$ yielded the lowest control error MSE.

Table 2.3: Comparing the MSE of control error, $e_c(k)$, produced using different forgetting factor values, λ , and a disturbance, $w(k)$, with variance $\sigma^2 = 0.3$.

	MSE of $e_c(k)$
$\lambda = 0.90$	0.25
$\lambda = 0.95$	0.17
$\lambda = 0.97$	0.16
$\lambda = 0.98$	0.18
$\lambda = 0.99$	1.25e^{+4}

Experimentation, using the extreme case values for noise variance σ^2 , suggests that the most suitable forgetting factor is $\lambda = 0.97$. This forgetting factor value is used in all the subsequent adaptive control problems. When performing multiple model adaptive control in Chapter 3 and 4, $\lambda = 0.97$ is the forgetting factor used for one of the adaptive models.

Figure 2.6: Control error, $e_c(k)$ produced when using different values for forgetting factor, λ (parameter jump at time-step $k = 200$, disturbance of variance $\sigma^2 = 0$).

Figure 2.7: Control error, $e_c(k)$ produced when using different values for forgetting factor, λ (parameter jump at time-step $k = 200$, disturbance of variance $\sigma^2 = 0.3$).

2.5.4 Demonstration of Persistency of Excitation Requirements for ELS

In this section, the ELS parameter estimation method is tested with a discontinuously exciting plant input and disturbance. When persistently exciting, the desired plant output, $y^*(k)$, operates as described by (2.50). The noise sequence $w(k)$ is zero mean Gaussian with variance $\sigma^2 = 0.1$. The time-invariant, plant parameter vector, θ^T , is defined by (2.57) in Section 2.5.2. The ELS algorithm is implemented using the forgetting factor value $\lambda = 0.97$.

In the first simulation, the reference signal, $y^*(k)$, and disturbance, $w(k)$, are forced to zero as follows:

$$y^*(k), w(k) = 0, \quad \text{for } 500 \leq k < 1800. \quad (2.59)$$

The plant input, $u(k)$, is no longer persistently exciting when the desired plant output, $y^*(k)$, is forced to zero (*i.e.* $u(k)$ is also forced to zero).

In Figure 2.8, the covariance matrix, $P(k)$, began to exponentially increase at time-step $k = 500$, when $u(k)$ and $w(k)$ were forced to zero. At time-step $k = 500$, no new information was available, but exponential forgetting continued. Upon the arrival of the first unique input, at time-step $k = 1800$, the covariance matrix, $P(k)$, immediately dropped to a low value. In Figure 2.9 (corresponding to the same simulation), the estimates of all parameters remained constant after time-step $k = 500$. Then, at time-step $k = 1800$, the arrival of new information caused there to be an explosive change in estimates. This caused the unbounded control signal, $u(k)$, and control error, $e_c(k)$, at time-step $k = 1800$.

A second simulation was performed to show the case when only the disturbance,

$w(k)$, is forced to zero. The disturbance, $w(k)$, was applied, as defined by (2.59), but the reference signal, $y^*(k)$, remained persistently exciting during the whole simulation. In Figure 2.10, it can be seen that the covariance matrix, $P(k)$, began to grow exponentially at time-step $k = 500$, despite the presence of the persistently exciting control input, $u(k)$. As a result, there was still an undesirably large control error at time-step $k = 1800$, when $w(k)$ was *persistently exciting* again. This example demonstrates that all parameters being estimated require appropriate excitation.

Figure 2.8: Performing adaptive control when the control signal, $u(k)$, and disturbance, $w(k)$, are not persistently exciting.

Figure 2.9: The parameter estimation process when the control signal, $u(k)$, and disturbance, $w(k)$, are not persistently exciting.

Figure 2.10: The parameter estimation process when only the disturbance, $w(k)$, is not persistently exciting.

Chapter 3

Stochastic Multiple Model Adaptive Control

3.1 Introduction

The method described here is similar to multiple model switching and tuning (MMST) method [1, 12, 13, 14, 16, 17]. Multiple model control is performed in this thesis using $N - 2$ fixed models and two adaptive models. This combination of fixed and adaptive models is considered a good compromise between computational complexity and performance [14, 15]. The greatest benefit is achieved when one of the adaptive models is free-running, and the other is strategically reset whenever a fixed model is selected. The resetting adaptive model uses initial parameter estimates taken from the newly selected fixed model. With proper model placement and switching, multiple model adaptive control provides improved stability and performance when compared to the single model case. The history and applications of multiple model adaptive control are discussed in Section 1.1.1.

The $N - 2$ fixed models and 2 adaptive models are designated as follows:

M_{fixed1}	–	fixed model 1,
$M_{fixed(N-2)}$	–	fixed model $N - 2$,
M_{reset}	–	adaptive resetting model $N - 1$,
M_{free}	–	adaptive free-running model N .

In [12], it was stated that “mathematically, fixed models only serve to provide better initial points from which the resetting model can adapt.” It is worth noting that the analyses performed in [11] and [12] showed that fixed models can also cause system instability. It is shown in Section 3.5.1 that the arguments for stability require assumptions and intricate analyses of the applied switching methods. These proofs are designed to show that the fixed models cannot be improperly, or permanently, selected. The most used model arrangement [1, 11, 12, 13, 14, 15, 16, 17, 19, 20] is named, in this thesis, the *classical model arrangement*. This thesis establishes the *modified model arrangement*. It is created with the intention that the statement, regarding fixed models [12] can be realized, not just at the mathematical level, but in application. The classical and modified model arrangements are further discussed in Section 3.2 and 3.3 respectively.

The applied performance index switching logic only requires the N models’ estimation errors, $e_1(k)$, $e_2(k)$, $\dots$, $e_N(k)$, to generate the model selection signal, $i(k) \in \{1 \dots N\}$. This is demonstrated in Section 3.4. As a result, the switching logic can be implemented identically for use with both the classical and modified model arrangement. In Section 3.5 the stability is discussed independently for both model arrangements. It is found that the proof of stability for the modified model arrangement is significantly simpler. In addition, the assumptions made in the clas-

sical model arrangement's proof are not needed in the modified model arrangement's proof. With the modified model arrangement, it is no longer necessary to analyze the applied switching logic's behavior. As a result, the problems of stability and performance are decoupled. New and existing switching methods that compromise the classical arrangement's stability can be blindly applied to the modified model arrangement. In Section 3.6 simulation studies demonstrate that both model arrangements have similar performance. This comparison is made while implementing both arrangements with the assumptions necessary for the classical model arrangement's stability.

3.2 Classical Model Arrangement

As shown in Figure 3.1, each of the N models has a corresponding matched controller. Model M_i , where $i \in \{1 \dots N\}$, is associated with controller C_i . When model M_i is selected, controller C_i is used to control the plant. This is a common approach to multiple model control [1, 11, 12, 13, 14, 15, 16, 17, 19, 20].

Controller C_i uses parameter estimates from model M_i and the desired plant output, $y^*(k)$, to generate the certainty equivalence control signal, $u_i(k)$. If M_i is an adaptive model, then the control signal, $u_i(k)$, is calculated using the procedure described in Section 2.3, which gives

$$u_i(k) = \frac{y^*(k+1) - \hat{\theta}_i^T(k+1)\tilde{\phi}_i(k)}{\hat{b}_{i1}(k+1)}. \quad (3.1)$$

The regression vector, $\tilde{\phi}_i(k)$, parameter estimate vector, $\hat{\theta}_i^T(k+1)$, and parameter estimate $\hat{b}_{i1}$ are associated with the adaptive model M_i . The certainty equivalence principle is more thoroughly discussed in Section 2.3.

 Figure 3.1: The classical model arrangement, N models and N controllers.

If M_i is a fixed model, then $u_i(k)$ is calculated as

$$u_i(k) = \frac{y^*(k+1) - \hat{\theta}_i^T \tilde{\phi}_i(k)}{\hat{b}_{i1}}, \quad (3.2)$$

where the regression vector, $\tilde{\phi}_i(k)$, fixed parameter estimate vector, $\hat{\theta}_i^T$, and fixed parameter estimate, $\hat{b}_{i1}$ are associated with the fixed model M_i .

As explained in Section 2.3, it is necessary to ensure that the parameter estimate

$\hat{b}_{i1} \geq b_{min} \geq 0$, when implementing (3.1) and (3.2). The simulations are performed using $b_{min} = 0.001$.

If the newly selected model, M_i , is a fixed model, then the resetting model, M_{reset} , will reinitialize with initial estimates taken from the parameter vector $\hat{\theta}_i^T$ of model M_i . The stability of the classical model arrangement with performance index switching is discussed in Section 3.5.

3.3 Modified Model Arrangement

The modified model arrangement excludes the controllers associated with the $N - 2$ fixed models. As shown in Figure 3.2, only the controllers corresponding to the two adaptive models are available to control the plant. The benefit of the fixed models is still realized, as they provide an initialization point from which model M_{reset} can adapt. The modified model arrangement is proposed by this thesis. The benefits to stability are shown Section 3.5.6. The resulting increase in performance is demonstrated using simulations in Section 3.6.

When either model, M_{free} or M_{reset} , is selected, it operates as previously defined by the classical model arrangement, and the corresponding controller, C_{free} or C_{reset} , is applied to control the plant. Both of these controllers generate a certainty equivalence control signal, as defined by (3.1). Selection of any fixed model M_i , where $i \in \{1 \dots N - 2\}$, causes the immediate re-initialization of model M_{reset} and application of controller C_{reset} . When model M_{reset} is re-initialized, it uses initial estimates taken from parameter vector $\hat{\theta}_i^T$ of the selected fixed model M_i . As a result, the fixed models are an extension to the adaptive model M_{reset} and only serve as strategic re-initialization points.

Figure 3.2: The modified model arrangement, excluding the controllers associated with the $N - 2$ fixed models (control signals $u_1 \dots u_{N-1}$ come from controller C_{reset}).

The following example illustrates how this modified model arrangement in Figure 3.2 would behave during two consecutive switches:

- At time-step k , the adaptive model M_{reset} is the selected model, and its corresponding controller, C_{reset} , is controlling the plant.
- At time-step $k + 1$, the switching logic selects the fixed model M_{fixed1} .
- In response, still at time-step $k + 1$, the adaptive model M_{reset} is reinitialized, taking initial estimates from parameter vector $\hat{\theta}_{fixed1}^T$ of model M_{fixed1} . Because

a fixed model was selected, controller C_{reset} remains applied and continues to control the plant (using the new M_{reset} parameters).

- At time-step $k + 2$, the switching logic selects the adaptive model M_{free} .
- In response, still at time-step $k + 1$, the controller C_{free} becomes the newly applied controller.

3.4 Performance Index Switching

The i th model's plant output estimate, $\hat{y}_i(k)$, is compared to the actual plant output to calculate the model's estimation error:

$$e_i(k) = y(k) - \hat{y}_i(k). \quad (3.3)$$

The estimation error, $e_i(k)$, is used in the performance index $J_i(k)$. The i th model is selected at any time-step, k , if

$$J_i(k) = \min_{j=1 \dots N} J_j(k).$$

A simple performance index, proposed in [11], is denoted by A, and defined

$$\text{A:} \quad J_i(k) = \frac{1}{k} \sum_{\tau=1}^k e_i^2(\tau). \quad (3.4)$$

The term, $1/k$, is incorporated to make an extension from the performance index

$$J_i(k) = \sum_{\tau=1}^k e_i^2(\tau),$$

which was intended for use in the deterministic case [11]. The term, $1/k$, ensures that performance index A, for a suitably selected model in bounded stochastic conditions, will not grow unbounded. A proof of stability for control with performance index A switching is provided in Section 3.5.3. This switching index is not suitable for multiple model control of a plant with time-varying conditions (parameters or disturbance), as it calculates each model's performance index $J_i(k)$ using all past estimation errors, $e_i(k)$.

Reference [1] provides the more versatile performance index denoted by B and defined

$$\text{B:} \quad J_i(k) = \frac{1}{k} \left[d_1 e_i^2(k) + d_2 \sum_{\tau=1}^k d_3^{k-\tau} e_i^2(\tau) \right], \quad (3.5)$$

where $d_3 \in (0, 1]$. The weights, d_1 , d_2 , and d_3 , are heuristically adjusted to achieve a desired switching behavior. This performance index is well suited for the stochastic and/or time varying case, as it integrates a fading memory with a sum-of-error. The compromise of switching “too often” and reacting to plant changes “too slowly,” is made by adjusted the ratio $d_1 : d_2$. As determined in [20], the ratio 1 : 2 yields suitable switching performance in most control scenarios. In this thesis, a new “control scenario” is considered to contain unique stochastic conditions (parameter jumps and disturbances) or fixed model placements. Weight d_3 determines the rate at which each model's past estimation errors are attenuated. Heuristic adjustment of d_3 is required for optimal switching, given a scenario's plant parameter behavior, disturbance, and model placement. A conditional proof of stability for control using performance index B switching is provided in Section 3.5.4.

3.5 Stability

3.5.1 Introduction

It is necessary to prove that both the classical model arrangement and modified model arrangement converge asymptotically with bounded signals when performance index B, (3.5), switching is used. As in the single model case, it is assumed that the transfer function $[1/C(q^{-1}) - \frac{1}{2}]$ is SPR. This requirement is more thoroughly stated by *Assumption (A1)* in Section 1.4. A definition of SPR is provided in Appendix A.2. As given by *Assumption (A5)*, the plant parameter vector, θ^T , is time-invariant. This assumption is necessary to simplify the proof of stability, but will be conditionally relaxed when considering switching.

An incremental approach is used to provide proofs of stability for the classical and modified model arrangements. The organization of this process is illustrated in Figure 3.3. In this diagram, a stability proof given in this thesis is shown as a box. The extensions necessary for each proof are shown using arrows. For the case when control is performed using a single adaptive model, stability is proven in Section 2.4 by making use of the Kronecker lemma. In Section 3.5.2, stability is shown for the case when control is performed using two adaptive models. A modified Kronecker lemma is necessary in this case. Stability is then proven, in Section 3.5.3, for the case when one fixed model and one free-running adaptive model are used. A fixed model introduces the possibility of there being an unstable closed loop controller-plant combination. As a result, the proof of stability requires analyses of applied switching methods to show that such a combination cannot become “frozen.” This condition, denoted as switching stability, is first shown when using the simple and well-researched performance index A, (3.4). Extensions are then made, in Section 3.5.4, to show that

switching stability can also be achieved using performance index B, (3.5). Due to the stochastic conditions, switching stability can only be guaranteed given assumptions regarding the amount of disturbance present and the weights used with performance index B. The case is then considered, in Section 3.5.5, when $N - 2$ fixed models and two adaptive models are used (this is the classical model arrangement). It is found that the same stability restrictions apply. In Section 3.5.2, stability is shown for the modified model arrangement. In this case, the analysis of any switching logic is not necessary for stability. The stability arguments used in Section 3.5.2, for the case of two adaptive models with arbitrary switching, allow for proof of stability of the modified model arrangement.

Figure 3.3: Chart showing the necessary extensions to prove stability of the classical and modified model arrangements (each box represents a stability proof and each arrow is an extension).

3.5.2 Stability of Two Adaptive Models

Two adaptive models form a multiple model controller as shown in Figure 3.4. Both models make the same assumptions concerning the plant and have random initial parameter estimates. This proof of stability assumes that switching is arbitrary or unknown. It shows that all the signals are mean-square bounded and that minimum variance control is achieved asymptotically. In [12], such a proof was given for the case when two adaptive models are implemented with different parameter estimation algorithms (the extended least squares (ELS) and output error with extended prediction model (OEEPM) algorithm). In this section, both models use ELS parameter estimation.

The two adaptive models produce their respective estimation errors, $e_1(k)$ and $e_2(k)$. The certainty equivalence control signal, $u(k)$, can be calculated using only one of these model's parameters at each time-step, k . As a result of the certainty equivalence principle, the control error, $e_c(k)$, is equal to only one of the estimation errors, $e_1(k)$ or $e_2(k)$, at any time-step, k . The Kronecker lemma, used in the single model case, cannot be applied. The modified Kronecker lemma is used in the proof, below, to show that all signals are bounded and growing at relatively equivalent rates. The modified Kronecker lemma is given in Appendix A.8.

It is assumed that plant parameters are not continuously jumping. Thus, it is shown that the overall scheme will converge after initialization or re-initialization of the adaptive models. The ensuing proof uses arguments adapted from [12].

Theorem 1:

The system, described above, has mean-square bounded plant inputs and outputs, and satisfies the minimum variance conditions given by (1.12)-(1.14) in Section 1.4.

Figure 3.4: Model arrangement containing only two adaptive models (resetting model has no fixed models to make it reset).

Proof of Theorem 1:

The model selected at time-step k is denoted by the model selection signal, $i(k) \in \{1, 2\}$. The selected model, $i(k)$, has a parameter estimation vector and regression vector denoted by $\hat{\theta}_i^T(k)$ and $\phi_i(k)$, respectively. The control error resulting from implementation of this model is denoted by $e_i(k)$. By definition of the certainty equivalence principle, the control error is equal to the estimation error of the selected model.

For the single model adaptive controller in Section 2.4, the relationship between the magnitudes of the *a posteriori* estimation error, $e(k)$, and the regression vector, $\phi(k)$, is represented as

$$\lim_{N \rightarrow \infty} \frac{1}{r(N-1)} \sum_{k=1}^N (e(k) - w(k))^2 = 0. \quad (3.6)$$

where $r(N-1)$ is the sum-of-squares of the norms of the regression vector, $\phi(N-1)$. Using the initial condition $r(k) = 0$ at $k = 0$, $r(N-1)$ is defined as

$$r(N-1) = r(N-2) + \phi^T(N-1)\phi(N-1) = \sum_{\tau=0}^{N-1} \phi^T(\tau)\phi(\tau). \quad (3.7)$$

Equation (3.7) is positive and monotonically increasing. The term monotonically increasing is defined in Appendix A.4. The rest of the proof of stability for the single model controller is shown in Section 2.4. Single model control was discussed above so that the ensuing discussion for the multiple model case is easier to understand.

For the multiple model controller considered in this chapter, the two adaptive models' *a posteriori* ELS estimation errors, $e_1(k)$ and $e_2(k)$, and regression vectors, $\phi_1(k)$ and $\phi_2(k)$, satisfy the inequalities

$$\lim_{N \rightarrow \infty} \frac{1}{r_1(N-1)} \sum_{k=1}^N (e_1(k) - w(k))^2 = 0 \quad (3.8)$$

and

$$\lim_{N \rightarrow \infty} \frac{1}{r_2(N-1)} \sum_{k=1}^N (e_2(k) - w(k))^2 = 0, \quad (3.9)$$

where $r_1(N-1)$ and $r_2(N-1)$ are the sums-of-squares of the norms of the regression vectors, $\phi_1(k)$ and $\phi_2(k)$, respectively.

Using the initial condition $r_1(k) = 0$ and $r_2(k) = 0$ for $k = 0$, $r_1(N-1)$ and $r_2(N-1)$ are defined as

$$r_1(N-1) = r_1(N-2) + \phi_1^T(N-1)\phi_1(N-1) = \sum_{\tau=0}^{N-1} \phi_1^T(\tau)\phi_1(\tau) \quad (3.10)$$

and

$$r_2(N-1) = r_2(N-2) + \phi_2^T(N-1)\phi_2(N-1) = \sum_{\tau=0}^{N-1} \phi_2^T(\tau)\phi_2(\tau). \quad (3.11)$$

As shown for the single model case, (2.37) in Section 2.4, the control error, $e_c(k)$, satisfies the inequality

$$\frac{r_i(N-1)}{N} \leq \frac{\alpha_{15}}{N} \sum_{k=1}^N (e_c(k) - w(k))^2 + \alpha_{16}, \quad (3.12)$$

where $0 < \alpha_{15}, \alpha_{16} < \infty$. As defined by the certainty equivalence principle in Section 2.3, the control error, $e_c(k)$, must equal the selected model's estimation error, $e_i(k)$, for any k . Similar to the single model case, specifically (2.38) in Section 2.4, it follows that

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N \frac{(e_c(k) - w(k))^2}{r_i(N-1)} < \infty. \quad (3.13)$$

The sum-of-squares of the norms of the two regression vectors, $r_1(k-1)$ and $r_2(k-1)$, are monotonically increasing functions of k (the term monotonically increasing is defined in Appendix A.4). But this cannot be guaranteed for the function $r_i(k-1)$ because the switching sequence, $i(k)$, is unknown and considered arbitrary in this stability analysis. The conditions required for use of the Kronecker lemma, stated in Appendix A.7, are not met. As a result, the Kronecker lemma cannot be used as it was for the single model case in Section 2.4.

In the following analysis, the modified Kronecker lemma will be used. First the conditions of the modified Kronecker lemma must be satisfied.

The output estimates $\hat{y}_1(k)$ and $\hat{y}_2(k)$, and the plant output, $y(k)$, are relatively equivalent as defined by (2.33)-(2.36) in Section 2.4. The output errors, $e_1(k)$ and $e_2(k)$, are defined as

$$e_1(k) = y(k) - \hat{y}_1(k)$$

and

$$e_2(k) = y(k) - \hat{y}_2(k).$$

The two regression vectors, $\phi_1(k)$ and $\phi_2(k)$, are defined as

$$\phi_1(k)^T = [-y(k), \dots, -y(k - n_A), u(k), \dots, u(k - n_B + 1), e_1(k), \dots, e_1(k - n_C + 1)]$$

and

$$\phi_2(k)^T = [-y(k), \dots, -y(k - n_A), u(k), \dots, u(k - n_B + 1), e_2(k), \dots, e_2(k - n_C + 1)].$$

Consequently, the sum-of-squares of the norms of the two regression vectors, $r_1(k-1)$ and $r_2(k-1)$, are monotonically increasing at identical, or comparable, rates.

The modified Kronecker lemma is used in the same manner as the Kronecker lemma is used in Section 2.4. The terms b_k and x_k , defined within the modified Kronecker lemma, are equated to the following terms from (3.13):

$$b_k : r_i(k-1).$$

$$x_k : \frac{(e_c(k) - w(k))^2}{r_i(k-1)}.$$

The modified Kronecker lemma, using b_k and x_k , gives

$$\lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N b_k x_k = 0, \quad (3.14)$$

which, by substitutions, corresponds to

$$\lim_{N \rightarrow \infty} \frac{1}{r_i(N-1)} \sum_{k=1}^N (e_c(k) - w(k))^2 = 0. \quad (3.15)$$

Equation (3.15) can be represented as

$$\lim_{N \rightarrow \infty} \frac{1}{r_i(N-1)/N} \sum_{k=1}^N (1/N) (e_c(k) - w(k))^2 = 0. \quad (3.16)$$

The modified Kronecker lemma is used in place of the Kronecker lemma to make the same argument. Equation (3.16) is very similar to (2.38) in Section 2.4. The proof of stability for two adaptive models is now performed in the same way as for the single model case in Section 2.4. It was assumed earlier in this Section that $r_i(N-1)$ grows unbounded, which results in (3.12) and (3.16) contradicting. A very informal analysis is used to make the contradiction more clear. The terms $L_1(N)$ and $L_2(N)$ are equated as follows:

$$L_1(N) : \frac{r_i(N-1)}{N}.$$

$$L_2(N) : \frac{1}{N} \sum_{k=1}^N (e_c(k) - w(k))^2.$$

Equations (3.12) and (3.16) become,

$$L_1(N) \leq \alpha_{15}L_2(N) + \alpha_{16}, \quad (3.17)$$

where $0 < \alpha_{15}, \alpha_{16} < \infty$, and

$$\lim_{N \rightarrow \infty} \frac{L_2(N)}{L_1(N)} = 0, \quad (3.18)$$

respectively. It is clear that (3.17) and (3.18) are contradicting. These equations were achieved assuming that $r_i(N-1)$ is unbounded. Thus this case cannot occur and $r_i(N-1)$ must be bounded (this case was explored above). The multiple model controller converges as follows

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N (e_c(k) - w(k))^2 = 0. \quad (3.19)$$

From (2.31) and (2.33), in Section 2.4, it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N y^2(k) = 0 \quad (3.20)$$

and

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N u^2(k) = 0. \quad (3.21)$$

Equations (3.20) and (3.21) are identical to (1.12) and (1.13) in Section 1.4.

Note that the minimum variance of $e_c(k)$ is expressed by (1.14) in Section 1.4, using conditional expectation, as

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N E[e_c(k)|k-1]^2 = \sigma^2, \quad (3.22)$$

where σ^2 is the variance of the disturbance, $w(k)$, acting on the plant defined by (1.6) in Section 1.4. The control error equation

$$y(k) - y^*(k) = e_c(k) \quad (3.23)$$

can be expressed, using conditional expectation, as

$$E[(y(k) - y^*(k))^2 | k-1] = E[(e_c(k) - w(k))^2 | k-1] + E[w^2(k) | k-1]. \quad (3.24)$$

Using (3.19) with (3.24), it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N E[e_c(k) | k-1]^2 = \sigma^2. \quad (3.25)$$

Equations (3.25) and (1.14) are identical. The exact goal of the adaptive control problem, as stated by (1.12)-(1.14) in Section 1.4, is achieved.

Proof of Theorem 1 is given and stability of the overall system is achieved. The proof required no analysis of the applied switching logic and is valid for arbitrary switching. As a result, the problems of performance and stability are decoupled.

3.5.3 Stability of One Fixed Model and One Free-Running Adaptive Model with Performance Index A Switching

When the analysis of stability for two adaptive models was performed, convergence was shown for random switching. A fixed model introduces the chance of there being an unstable closed loop controller-plant combination. Consequently, the proof of stability must incorporate analysis of performance index A, (3.4). Switching cannot become “frozen” with an unstable controller-plant combination. This section goes

further and proves that the switching scheme will converge, at some time-step, K , such that the adaptive model will be selected. In this thesis, this condition for stability is denoted as switching stability.

Note that performance index A is not desirable for use with this thesis (as discussed in Section 3.4). Extensions to this proof of stability are made in Section 3.5.4 to show stability for the chosen method, performance index B, (3.5).

The proof of stability, using performance index A, is adapted from [11] and presented as follows.

Theorem 2:

There exists a finite time-step K , and lower bound $\epsilon > 0$, such that

$$J_f(k) - J_a(k) > \epsilon > 0, \quad \forall k \geq K,$$

where $J_f(K)$ and $J_a(K)$ are the performance indices A associated with the fixed model and the single adaptive model, respectively.

Theorem 2 states that the switching scheme converges such that the adaptive model will be selected. In order to prove Theorem 2, the following two lemmas are necessary.

Lemma 3 - Necessary for proving Theorem 2:

If a_k is a sequence of real numbers such that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N a_k < \infty, \tag{3.26}$$

then

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N \frac{1}{k^2} a_k < \infty. \quad (3.27)$$

The proof of Lemma 3 is provided in Appendix A.5.

Lemma 4:

Let $X(k)$ be a random variable changing with respect to the time-steps, k . If $E[X(k)|k-l] = 0$, where l is a finite integer, and

$$\sum_{k=1}^{\infty} \frac{1}{k^2} E[X^2(k)|k-l] < \infty, \quad (3.28)$$

then

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N X(k) = 0. \quad (3.29)$$

Proof of Lemma 4 is trivial, as it is a direct result of the Kronecker lemma given in Appendix A.7.

Proof of Theorem 2:

The one fixed model's performance index and model estimation error is denoted by $J_f(k)$ and $e_f(k)$, respectively. The one adaptive model's performance index and model estimation error is denoted by $J_a(k)$ and $e_a(k)$, respectively. The model selection at time-step k is denoted by $i(k) \in \{1, 2\}$. The control error resulting from selection of this model is denoted by $e_i(k)$. This can be done because, by definition of the certainty equivalence principle, the control error is equal to the estimation error of the selected model.

By the definition of performance index A, (3.4), it follows that

$$\begin{aligned}
 J_f(N) &= \frac{1}{N} \sum_{k=1}^N e_f^2(k), \\
 &= \frac{1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2 + \\
 &\quad \frac{1}{N} \sum_{k=1}^N w^2(k) + \\
 &\quad \frac{2}{N} \sum_{k=1}^N (e_f(k) - w(k)) w(k),
 \end{aligned} \tag{3.30}$$

and

$$\begin{aligned}
 J_a(N) &= \frac{1}{N} \sum_{k=1}^N e_a^2(k), \\
 &= \frac{1}{N} \sum_{k=1}^N (e_a(k) - w(k))^2 + \\
 &\quad \frac{1}{N} \sum_{k=1}^N w^2(k) + \\
 &\quad \frac{2}{N} \sum_{k=1}^N (e_a(k) - w(k)) w(k).
 \end{aligned} \tag{3.31}$$

By definition of the certainty equivalence principle, the control error, $e_c(k)$, is equal to the model estimation error, $e_i(k)$, where $i \in \{1, 2\}$ denotes the model selected at time-step k . Consequently, it follows that

$$(e_c(k+1) - w(k+1))^2 = (e_i(k+1) - w(k+1))^2. \tag{3.32}$$

The switching scheme is proven to converge using two mutually exclusive but collectively exhaustive cases. First, in *Case (1)*, it is assumed that the plant output is bounded, such that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N y(k) < \infty.$$

Then, in *Case (2)*, the alternative assumption is made, where

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N y(k) = \infty,$$

and contradictions are found, which shows that *Case (2)* cannot occur.

Case (1) - The plant output, $y(k)$, is bounded:

The adaptive model is first analyzed, and it will be shown that the estimation error, $e_a(k)$, converges. Because a single adaptive model is considered the analysis used for single model adaptive control in Section 2.4 can be used. The Section 2.4 method proves convergence of the control error, but because the certainty equivalence principle is used, the convergence of estimation error for the one adaptive model is also proven. Following (2.31)-(2.36) in Section 2.4, all of the adaptive model's signals (*e.g.* $e_a(k)$ and $u_a(k)$) are bounded because $y(k)$ is bounded. Following (2.37), it can be stated that

$$\lim_{N \rightarrow \infty} \frac{r_a(N-1)}{N} < \infty, \quad (3.33)$$

and following how (2.41) is achieved, it is shown that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N (e_a(k) - w(k))^2 = 0. \quad (3.34)$$

The fixed model's behavior is analyzed next using three sub-cases. Because the adaptive model converges in all sub-cases, the stability of the overall multiple model controller is separately analyzed in each sub-case. The term L_3 is equated as follows:

$$L_3 : \lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2.$$

And the three sub-cases are defined as:

Case(1.1): Fixed model's estimation error, $e_f(k)$, converges ($L_3 = 0$).

Case(1.2): Fixed model's estimation error, $e_f(k)$, is bounded ($0 < L_3 < \infty$).

Case(1.3): Fixed model's estimation error, $e_f(k)$, is unbounded ($L_3 = \infty$)

(*proven to be a contradiction*).

These three sub-cases are independently analyzed below.

Case (1.1) - Fixed model's estimation error, $e_f(k)$, converges ($L_3 = 0$):

In this case, the fixed model has parameter identical to those of the plant, the model's corresponding controller is perfectly matched, and

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2 = 0.$$

From (3.32), it follows that

$$\frac{1}{N} \sum_{k=1}^N (e_c(k) - w(k))^2 = \frac{1}{N} \sum_{k=1}^N (e_i(k) - w(k))^2.$$

By previous assumptions and the stability analysis in Section 2.4, we have

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N (e_c(k) - w(k))^2 = 0.$$

Case (1.2) - Fixed model's estimation error, $e_f(k)$, is bounded ($0 < L_3 < \infty$):

In this case, the closed loop controller-plant combination is stable, all relevant signals are bounded, and

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2 < \infty.$$

From Lemma 4 defined above, it follows that

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N \frac{1}{k^2} (e_f(k) - w(k))^2 < \infty.$$

Given the conditional expectation relationship

$$E[(e_f(k) - w(k))w(k)|k-1] = (e_f(k) - w(k))E[w(k)|k-1] = 0, \quad (3.35)$$

it follows that

$$\sum_{k=1}^N \frac{1}{k^2} E[(e_f(k) - w(k))^2 w^2(k)|k-1] = \sum_{k=1}^N \frac{1}{k^2} [(e_f(k) - w(k))^2 w^2(k)] < \infty. \quad (3.36)$$

By using Lemma 3 defined above, it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N [(e_f(k) - w(k))w(k)] = 0. \quad (3.37)$$

Using the same approach for the adaptive model, it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N [(e_a(k) - w(k))w(k)] = 0. \quad (3.38)$$

Considering the above results and (3.30), the following can be stated for $J_f(N)$ as $N \rightarrow \infty$:

1. The 1st term is bounded below by a constant C .
2. The 2nd term remains constant.
3. The 3rd term tends to zero.

And considering equation (3.38) with (3.31), the following can be stated for $J_a(N)$ as $N \rightarrow \infty$:

1. The 1st term tends to zero.
2. The 2nd term remains constant.
3. The 3rd term tends to zero.

The convergence of the two model's performance indices can be stated as

$$J_f(N) = C + \frac{1}{N} \sum_{k=1}^N w^2(k) + 0, \quad \text{as } N \rightarrow \infty$$

and

$$J_a(N) = 0 + \frac{1}{N} \sum_{k=1}^N w^2(k) + 0, \quad \text{as } N \rightarrow \infty.$$

Thus, there will be a time-step K , where $J_a(K) < J_f(K)$ for all future time-steps. Switching will converge to the adaptive model (*i.e.* switching stability is achieved).

Case (1.3) - Fixed model's estimation error, $e_f(k)$, is unbounded ($L_3 = \infty$):

In this case the fixed model's estimation error is assumed to be unbounded as follows:

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N y(k) = \infty,$$

However, $C(q^{-1})$ from (1.10) in Section 1.4, and the fixed model's estimate of $C(q^{-1})$, have roots strictly inside the unit circle (from *Assumption (A1)* in Section 1.4). Taking (2.31) from Section 2.4, it follows that

$$\frac{1}{N} \sum_{k=1}^N y^2(k) \leq \frac{\alpha_1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2 + \alpha_2 \quad (3.39)$$

and

$$\frac{1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2 \leq \frac{\alpha_3}{N} \sum_{k=1}^N y^2(k) \alpha_4, \quad (3.40)$$

where $0 < \alpha_1, \alpha_2, \alpha_3, \alpha_4 < \infty$. This is a contradiction to the assumption necessary for *Case (1)*:

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N y(k) < \infty.$$

Consequently, *Case (1.3)* cannot occur.

Case (2) - The plant output, $y(k)$, is unbounded:

It is assumed that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N y(k) = \infty. \quad (3.41)$$

The adaptive model is first analyzed, and it will be shown that the estimation error, $e_a(k)$, grows unbounded.

Note that the term, $r_a(N-1)$, is the sum-of-squares of the norms of the adaptive model's regression vector, $\phi_a(k)$. Using the initial condition $r_a(k) = 0$, for $k = 0$, $r_a(N-1)$ is defined as

$$r_a(N-1) = r_a(N-2) + \phi_a^T(N-1)\phi_a(N-1) = \sum_{\tau=0}^{N-1} \phi_a^T(\tau)\phi_a(\tau) \quad (3.42)$$

Consequently, as the plant output grows unbounded as shown by (3.41), $r_a(N-1)$ grows unbounded as follows

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N \frac{r_a(N-1)}{N} = \infty. \quad (3.43)$$

Using the Kronecker lemma defined in Appendix A.7 and (2.27), it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{r_a(N-1)} \sum_{k=1}^N (e_a(k) - w(k))^2 = 0. \quad (3.44)$$

This shows that the adaptive model's regression vector grows faster than its estimation error. It is of interest to know how the following signal converges:

$$\lim_{N \rightarrow \infty} \frac{1}{r_a(N-1)} \sum_{k=1}^N (e_f(k) - w(k))^2. \quad (3.45)$$

Equation (3.41) is denoted by L_4 as:

$$L_4 : \lim_{N \rightarrow \infty} \frac{1}{r_a(N-1)} \sum_{k=1}^N (e_f(k) - w(k))^2.$$

The two possible behaviors for L_4 are analyzed using the sub-cases:

Case(2.1): The term $L_4 \geq l > 0$, where l is a scalar
(*proven to be a contradiction*).

Case(2.2): The term L_4 has no such lower bound l , $L_4 \geq l > 0$
(*proven to be a contradiction*).

Case (2.1) - The term $L_4 \geq l > 0$, where l is a scalar:

It is assumed that

$$\lim_{N \rightarrow \infty} \frac{1}{r_a(N-1)} \sum_{k=1}^N (e_f(k) - w(k))^2 \geq L_4 > 0, \quad (3.46)$$

which implies that adaptive model's regression vector grows slower than the fixed model's estimation error. Using (3.44) and (3.46), it follows that

$$\lim_{N \rightarrow \infty} \left(\frac{\sum_{k=1}^N (e_a(k) - w(k))^2}{\sum_{k=1}^N (e_f(k) - w(k))^2} \right) = 0. \quad (3.47)$$

This equation compares the growth rate of the two model estimation errors and shows that the adaptive model's output error grows more slowly than that of the fixed model. Using (3.43), it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2 = \infty,$$

and from the Schwarz inequality (in Appendix A.6) it can be shown that

$$\left| \frac{1}{N} \sum_{k=1}^N [(e_f(k) - w(k)) w(k)] \right| \leq \sqrt{\frac{1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2} \sqrt{\frac{1}{N} \sum_{k=1}^N w^2(k)}. \quad (3.48)$$

The terms of (3.48) are used to analyze how $J_f(N)$, defined by (3.30), behaves. The first term of $J_f(N)$,

$$\frac{1}{N} \sum_{k=1}^N (e_f(k) - w(k))^2,$$

will dominate the other terms. Thus, switching will stop at some time-step K and an adaptive model will be selected. This is a contradiction to the assumption that the plant output, $y(k)$, grows unbounded. As a result, *Case (2.1)* cannot occur.

Case (2.2) - The term L_4 has no such lower bound l , $L_4 \geq l > 0$:

Let there be a subsequence, $\{N_s\}$, where $(N_s = N_1, N_2, \dots)$, such that

$$\lim_{N_s \rightarrow \infty} \frac{1}{r_a(N_s - 1)} \sum_{k=1}^{N_s} (e_f(k) - w(k))^2 = 0, \quad (3.49)$$

which implies that adaptive model's regression vector grows faster than the fixed model's estimation error. From (3.32), it follows that

$$\frac{1}{r_a(N_s - 1)} \sum_{k=1}^{N_s} (e_c(k) - w(k))^2 = \frac{1}{r_a(N_s - 1)} \sum_{k=1}^{N_s} (e_i(k) - w(k))^2.$$

The adaptive model's estimation error, $e_a(k)$, converges as defined by (3.44). The fixed model's estimation error, $e_f(k)$, converges for the subsequence, N_s , as defined by (3.49). By definition of the certainty equivalence principle, the control error, $e_c(k)$, is equal to the i th's model estimation error, $e_i(k)$, where the model selected at time-step k is denoted by $i(k) \in \{1, 2\}$. Because both estimation errors converge, it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{r_a(N_s - 1)} \sum_{k=1}^{N_s} (e_c(k) - w(k))^2 = 0.$$

Using arguments similar to those for the single model case in Section 2.4, it follows that

$$\lim_{N_s \rightarrow \infty} \sum_{k=1}^N \frac{r_a(N_s - 1)}{N_s} < \infty. \quad (3.50)$$

This is a contradiction to (3.43). As a result, *Case (2.2)* cannot occur. Because the mutually exclusive cases, *Case (2.1)* and *Case (2.2)*, result in a contradiction, *Case (2)* cannot occur. The analysis performed in *Case (1)* shows that an adaptive model

will become selected. Stable control of a plant using one adaptive model, as in this case, is proven in Section 2.4. The plant inputs and outputs are bounded and the overall system is convergent.

3.5.4 Stability of One Fixed Model and One Free-Running Adaptive Model with Performance Index B Switching

It must be shown that the switching scheme will converge at some time-step, K , such that the adaptive model will be selected. This condition, denoted as switching stability, was proven in Section 3.5.3 for performance index A. Performance index B, (3.5), is used in this thesis for reasons stated in Section 3.4. Extensions to the proof for performance index A are made to show stability for B. Performance index B allows for better performance than A but stability is compromised. These compromises are made in the form of conditions, which are discussed at the end of this section.

Following the method of Section 3.5.3, there are only two models and performance indices, $J_f(k)$ or $J_a(k)$. It will be proven, by Theorem 3 below, that there exists a time-step K , where the switching will converge to the adaptive model (this is switching stability). The conditions for Theorem 3 will be given in the proof.

Theorem 3:

There exists a finite time-step K , and lower bound $\epsilon > 0$, such that

$$J_f(k) - J_a(k) > \epsilon > 0, \quad \forall k \geq K,$$

where $J_f(k)$ and $J_a(k)$ are the performance indices B, (3.5), associated with the fixed model and the single adaptive model, respectively.

Conditional Proof of Theorem 3:

Theorem 3 is very similar to Theorem 2 used in Section 3.5.3. An assumption regarding performance index B is necessary to simplify the proof of Theorem 3.

Assumption:

(B) The performance index B user-defined weights are assigned as: $d_1, d_2, d_3 = 1$.

Although this thesis uses values $d_1 = 1$ and $d_2 = 2$, relaxing *Assumption (B)* in this sense causes no difference to the proof. The effects of using other values for the forgetting factor weight, $d_3 \in (0, 1]$, will be discussed later. Using *Assumption (B)*, the performance index B becomes

$$J_i(k) = \frac{1}{k}e_i^2(k) + \frac{1}{k} \sum_{\tau=1}^k e_i^2(\tau). \quad (3.51)$$

The second term in (3.51), $(1/k) \sum_{\tau=1}^k e_i^2(\tau)$, is (3.4), performance index A. The convergence of this term is exhaustively shown in the Section 3.5.3. In this section, the effect of the added term, $e_i^2(k)/k$, is analyzed (even though it is intuitive that the term does not affect switching stability).

Equations (3.30) and (3.31) in Section 3.5.3 are modified to suit performance index (3.51), instead of A, as follows:

$$\begin{aligned}
 J_f(N) &= \frac{1}{N}e_f^2(N) + \frac{1}{N}\sum_{k=1}^N e_f^2(k), \\
 &= \frac{1}{N}e_f^2(N) + \\
 &\quad \frac{1}{N}\sum_{k=1}^N (e_f(k) - w(k))^2 + \\
 &\quad \frac{1}{N}\sum_{k=1}^N w^2(k) + \\
 &\quad \frac{2}{N}\sum_{k=1}^N (e_f(k) - w(k))w(k), \tag{3.52}
 \end{aligned}$$

$$\begin{aligned}
 J_a(N) &= \frac{1}{N}e_a^2(N) + \frac{1}{N}\sum_{k=1}^N e_a^2(k), \\
 &= \frac{1}{N}e_a^2(N) + \\
 &\quad \frac{1}{N}\sum_{k=1}^N (e_a(k) - w(k))^2 + \\
 &\quad \frac{1}{N}\sum_{k=1}^N w^2(k) + \\
 &\quad \frac{2}{N}\sum_{k=1}^N (e_a(k) - w(k))w(k). \tag{3.53}
 \end{aligned}$$

Equation (3.52) is different from (3.30) because it has the added term $(1/N)e_f^2(N)$, and (3.53) is different from (3.31) because it has the added term $(1/N)e_a^2(N)$. The convergence of the other terms in (3.52) and (3.53) has already been discussed in Section 3.5.3. For the extra terms, $(1/N)e_f^2(N)$ and $(1/N)e_a^2(N)$ in (3.52) and (3.53), respectively, it can be said that

$$\frac{1}{N}e_f^2(N) < \infty$$

and

$$\frac{1}{N}e_a^2(N) = 0,$$

for all cases, (1.1), (1.2), (1.3), (2.1), and (2.2).

As a result, (3.52) converges in the same way as (3.30), and (3.53) converges in the same way as (3.31). Theorem 3 is proven and, as shown in Section 3.5.3, there exists a time-step, K , where the adaptive model's performance index is less than that of the fixed model, for all time-steps $k > K$.

This proof of stability was made using $d_3 = 1$. Given the presence of noise, the convergence of the switching scheme will depend on the forgetting factor weight d_3 and the signal-to-noise ratio. From experimentation performed in [16], it was found that a large signal-to-noise ratio will allow switching to converge to the adaptive model for almost all $d_3 \in (0, 1]$, whereas a small signal-to-noise ratio will only ensure such convergence for d_3 values close to 1. Using performance index B instead of A allows for switching performance improvements at the cost of applying these conditions.

When stability is not guaranteed, there are two possible outcomes: a fixed model with some parameterization error will be permanently selected or switching will never converge to exclude fixed models (this is *chattering*).

3.5.5 Stability of the Classical Model Arrangement with Performance Index B Switching

The classical model arrangement is composed of $N - 2$ fixed models and two adaptive models. Stability is shown by making extensions from Section 3.5.4, where one fixed model and one adaptive model are used. Additional models do not adversely affect the stability of the overall system, as discussed in [11]. The analysis of stability is made using the same method as in the case of one fixed model and one adaptive model.

The performance indices for the $N - 2$ fixed models are denoted by

$$J_{fixed1}(k), J_{fixed2}(k), \dots, J_{fixed(N-2)}(k),$$

and the performance indices for the two adaptive models are denoted by

$$J_{reset}(k), \text{ and } J_{free}(k).$$

Theorem 4:

There exists a finite time-step K and lower bound $\epsilon > 0$, such that

$$\{J_{fixed1}(k), \dots, J_{fixed(N-2)}(k)\} - \{J_{reset}(k), J_{free}(k)\} > \epsilon > 0, \quad \forall k \geq K,$$

Proof of Theorem 4:

Theorem 4 is a modified version of Theorem 3 in Section 3.5.4. The convergence of performance index B, (3.5), for any fixed and adaptive model is demonstrated in

Theorem 3. It is shown that the performance index $J_{free}(k)$ will converge to become smaller than the performance indices for the fixed models, $J_{fixed1}(k)$, $J_{fixed2}(k)$, $\dots$, $J_{fixed(N-2)}(k)$. Once this convergence has happened, fixed models cannot become selected and the resetting adaptive model will become a 2nd free-running model. As a result, $J_{reset}(k)$ will converge in the same manner as $J_{free}(k)$. Because $J_{reset}(k)$ and $J_{free}(k)$ converge in the same manner, switching may continue between the two adaptive models. It is shown in Section 3.5.2 that the overall system is convergent even if the switching is allowed to continue between two adaptive models.

The same conditions to stability, described in more detail in Section 3.5.4, remain necessary. In the presence of noise, the convergence of the model's performance indices will depend on the forgetting factor weight, d_3 , and the signal-to-noise ratio. From experimentation performed in [16], it was found that a large signal-to-noise ratio will allow switching to converge to an adaptive model for almost all $d_3 \in (0, 1]$, whereas a small signal-to-noise ratio will only ensure such convergence for d_3 close to 1.

3.5.6 Stability of the Modified Model Arrangement

It is of interest to note that in [12] it was stated “mathematically, fixed models only serve to provide better initial points from which the resetting model can adapt.” With the modified model arrangement, it is intrinsic that any controller-plant combination will utilize an adaptive model's controller and be asymptotically convergent. As a result, this statement in [12] is truly realized (this is not the case for the classical model arrangement in Section 3.5.5).

The stability argument used for the arrangement of two adaptive models in Section 3.5.2 can be used to prove stability for the modified model arrangement. Theorem 1 in Section 3.5.2 demonstrates stability without requiring analysis of the applied switch-

ing logic. As a result, the multiple model controller's problems of performance and stability are decoupled. Theorem 1 states that all signals are mean-square bounded and minimum variance control is asymptotically achieved. These two adaptive models are both free-running. One of the modified model arrangement's adaptive models reinitializes every time a fixed model is selected. For the modified model arrangement's control error to be asymptotically convergent, it is required that the resetting model is not constantly reset. If the resetting model was constantly reset with initial parameters taken from a fixed model, and never allowed a small time T_{min} to converge, then it would essentially be a fixed model. This one condition to stability can be stated as follows: switching can only occur such that models are not switched before some small time T_{min} .

No matter what model is selected, there can be no unstable "frozen" controller-plant combination, and convergence will occur. As a result, accidental switching can cause performance degradation for the modified model arrangement, but it cannot compromise stability. This is demonstrated using simulations in Section 3.6.5.

3.6 Simulations

Independent sections are used to introduce each unique topic's simulations and analyses. The initial and operating conditions common to all of the simulations are described in Section 3.6.1. The convergence of multiple model control, using only two adaptive models, is demonstrated in Section 3.6.2. In Section 3.6.3, some advantages of multiple model control are demonstrated by making comparisons to the single model case. Two sections are then used to compare the classical model arrangement and the modified model arrangement. In Section 3.6.4, performance index B

is implemented and tuned to achieve ideal switching. It is shown that both model arrangements achieve similar performance. In Section 3.6.5, unintelligent switching is applied to both model arrangements and the modified model arrangement demonstrates its ability to preserve stability.

3.6.1 Conditions Common to All Simulations

The stochastic plant to be controlled is described by the difference equation,

$$\begin{aligned} y(k+1) = & a_1(k)y(k) + a_2(k)y(k-1) + b_1(k)u(k) + b_2(k)u(k-1) \\ & + c_1(k)w(k) + c_2(k)w(k-1) + w(k+1), \end{aligned} \quad (3.54)$$

where the noise signal, $w(k)$, is zero mean Gaussian with variance σ^2 . The plant's parameter vector, $\theta^T(k)$, is defined as

$$\theta^T(k) = [a_1(k) \quad a_2(k) \quad b_1(k) \quad b_2(k) \quad c_1(k) \quad c_2(k)]. \quad (3.55)$$

The actual values of the parameters in $\theta^T(k)$ and the noise variance, σ^2 , are specified within each of the experimental sections. The desired plant output is described by the persistently exciting function

$$y^*(k+1) = \sin(2\pi k/50) + \sin(2\pi k/150), \quad (3.56)$$

which clearly has the initial value $y^*(0) = 0$. The plant output initializes with $y(0) = 0$. As a result, there is zero initial control error, calculated

$$e_c(0) = y(0) - y^*(0) = 0. \quad (3.57)$$

Also, the resetting and free-running adaptive models are both initialized with an initial ELS plant output estimate of zero (*i.e.* $\hat{y}_{reset}(0), \hat{y}_{free}(0) = 0$). In any simulation containing fixed models, these models' initial plant output estimate are also zero (*i.e.* the i th fixed model has plant estimate $\hat{y}_i(0) = 0$). As a result, there are no initial model estimation errors. This is shown by the calculations:

$$e_{reset}(0) = y(0) - \hat{y}_{reset}(0),$$

$$e_{free}(0) = y(0) - \hat{y}_{free}(0),$$

$$e_i(0) = y(0) - \hat{y}_i(0).$$

The effects of initializing with zero control error and estimation errors are shown for the single model case in Section 2.5.2. It was demonstrated that transient control error and convergence occur once the reference signal, $y^*(k+1)$, excites the system (*i.e.* once it moves away from $y^*(0) = 0$).

For the multiple model case, the initialization of the switching algorithm must also be considered. As discussed above, the initial model estimation errors, $e_1(k)$, $e_2(k)$, $\dots$, $e_N(k)$, are zero. Each model's performance index B , (3.5), is based on the model's model estimation error. As a result, the N model performance indices, $J_1(k)$, $J_2(k)$, $\dots$, $J_N(k)$, are also equal and the minimum performance index, $J_i(k)$, where

$$J_i(k) = \min_{j=1 \dots N} J_j(k),$$

cannot be determined. As a result, it is necessary to force a model to be selected at time-step $k = 0$ so that the certainty equivalence control signal, $u(k)$, can be

calculated for time-step $k = 1$. The initial model selection signal, $i(0)$, is user-defined within each simulation section.

Let the resetting and free-running models' parameter estimate vectors be denoted by $\hat{\theta}_{reset}^T(k)$ and $\hat{\theta}_{free}^T(k)$, respectively. The adaptive models parameter estimates are updated by the ELS method described in Section 2.2.2. The parameter estimate vectors, $\hat{\theta}_{reset}^T(k)$ and $\hat{\theta}_{free}^T(k)$, estimate plant state (3.55) using the initial estimates

$$\hat{\theta}_{reset}^T(0), \hat{\theta}_{free}^T(0) = [0 \ 0 \ 1 \ 0 \ 0 \ 0]. \quad (3.58)$$

The parameter estimate condition $\hat{b}_1(k+1) \geq b_{min} \geq 0$ is enforced using $b_{min} = 0.001$. This condition, required for certainty equivalence control, is further explained in Section 2.3.

The two adaptive methods' associated ELS covariance matrices, $P_{reset}(k)$ and $P_{free}(k)$, are initialized as

$$P_{reset}(0), P_{free}(0) = 10^5 I_6. \quad (3.59)$$

These initial values are large to ensure that initial parameter estimate step-sizes are large.

Certainty equivalence control is used in all simulations. The certainty equivalence principle is further described in Section 2.3 and by (3.1) and (3.2) in Section 3.2.

3.6.2 Demonstration of Convergence with Random Switching Between Two Adaptive Models

The convergence of single model adaptive control is proven and demonstrated in Section 2.5. Convergence for the case of two adaptive models with arbitrary switching

is proven in Section 3.5.2. In this section, convergence is demonstrated by simulation.

Two adaptive models, M_{reset} and M_{free} , are implemented using the model arrangement in Figure 3.4. There are no fixed models, which if present and switched to, would cause re-initialization of model M_{reset} . Consequently, the resetting model, M_{reset} , actually operates as a 2nd free-running model.

The time-invariant plant parameter vector, θ^T , is defined as

$$\theta^T = [0.6 \quad 0.2 \quad 1.0 \quad 0.5 \quad -0.4 \quad 0.4]. \quad (3.60)$$

These plant parameters are arbitrarily chosen and satisfy *Assumptions (A1)-(A5)* in Section 1.4. The noise signal, $w(k)$, is zero mean Gaussian with a variance value $\sigma^2 = 0.1$.

The two adaptive models' parameter estimate vectors are randomly initialized. Using values $\delta_1, \dots, \delta_{10}$, uniformly distributed on the interval $[-1, 1]$, the parameter estimate vectors are initialized as

$$\hat{\theta}_{reset}^T(0) = [\delta_1 \quad \delta_2 \quad 1 \quad \delta_3 \quad \delta_4 \quad \delta_5] \quad (3.61)$$

and

$$\hat{\theta}_{free}^T(0) = [\delta_6 \quad \delta_7 \quad 1 \quad \delta_8 \quad \delta_9 \quad \delta_{10}]. \quad (3.62)$$

The parameter estimate $\hat{b}_1$ in (3.61) and (3.62) is initialized as $\hat{b}_1 = 1$ to ensure that the condition for stable certainty equivalence control is satisfied. This requirement is further explained in Section 2.3.

The adaptive models M_{reset} and M_{free} use forgetting factor values $\lambda_{reset} = 0.97$ and $\lambda_{free} = 1.00$, respectively.

The switching signal, $i(k)$, selects models M_{reset} and M_{free} arbitrarily, as plotted in Figure 3.5. The performance of the overall adaptive scheme during this random switching is shown in Figure 3.6. The signals are initialized at zero as follows: $y^*(0) = 0$, $y(0) = 0$, and as a result, $e_c(0) = 0$. Transient control error occurred once the reference signal, $y^*(k+1)$, excited the system (*i.e.* once it moved away from $y^*(0) = 0$). It is clear that all signals remained bounded and the overall system converged. The convergence occurred because both of the implemented models had convergence properties. Because both models' parameter estimates were accurate, it made no difference which estimates were selected for calculation of the certainly equivalence control input, $u(k)$.

Figure 3.5: The signal $i(k)$ randomly switching between the two adaptive models.

Figure 3.6: The convergence of the overall adaptive scheme, during the random model switching.

3.6.3 Comparison of Multiple and Single Model Control Performance

A multiple model controller and single model controller are simulated in an identical control scenario (*i.e.* the initial conditions and disturbances are identical). Both controllers make the same assumptions concerning the plant and are initialized as described in Section 3.6.1. The plant parameter vector, $\theta^T(k)$, is piece-wise constant in time as follows:

$$\theta^T(k) = \begin{cases} \theta_1^T &= [0.6 \ 0.2 \ 1.0 \ 0.5 \ -0.5 \ 0.4], \quad k \leq 500, \\ \theta_2^T &= [-0.5 \ 0.2 \ 1.0 \ 0.2 \ -0.5 \ 0.4], \quad k > 500. \end{cases} \quad (3.63)$$

This plant is composed of two arbitrarily chosen stable plant states that satisfy *Assumptions (A1)-(A5)* in Section 1.4. The noise signal, $w(k)$, is zero mean Gaussian with a variance of $\sigma^2 = 0.1$.

The single model controller is implemented as defined by (2.53)-(2.56) in Section 2.5. The ELS adaptive algorithm is used with a forgetting factor value $\lambda = 0.97$. This value for λ was found to be an optimal choice in Section 2.5.3.

The multiple model controller uses the classical model arrangement and performance index B switching. Along with the adaptive models, M_{reset} and M_{free} , two fixed models, M_{fixed1} and M_{fixed2} , are used. These models have respective fixed parameter estimate vectors, $\hat{\theta}_{fixed1}^T$ and $\hat{\theta}_{fixed2}^T$, defined to be in the neighborhood of the two (3.63) plant states, θ_1^T and θ_2^T , respectively. When a model is in the “neighborhood” of a plant state, this implies that the model’s parameters are similar to those of the plant state, with some parameterization error. Using the values $\delta_{11}, \dots, \delta_{22}$, uniformly distributed on the interval $[-1, 1]$, the two fixed models’ parameter estimate

vectors are defined as

$$\hat{\theta}_{fixed1}^T = \theta_1^T + 0.20[\delta_{11} \quad \delta_{12} \quad \delta_{13} \quad \delta_{14} \quad \delta_{15} \quad \delta_{16}], \quad (3.64)$$

and

$$\hat{\theta}_{fixed2}^T = \theta_2^T + 0.20[\delta_{17} \quad \delta_{18} \quad \delta_{19} \quad \delta_{20} \quad \delta_{21} \quad \delta_{22}]. \quad (3.65)$$

The plant states, θ_1^T and θ_2^T , are defined by (3.63). The resetting and free-running adaptive models' parameter estimate vectors initialize as

$$\hat{\theta}_{reset}^T(0) = \hat{\theta}_{fixed1}^T,$$

and

$$\hat{\theta}_{free}^T(0) = [0 \quad 0 \quad 1 \quad 0 \quad 0 \quad 0],$$

respectively. The resetting and free-running models use forgetting factor values $\lambda_{reset} = 0.97$ and $\lambda_{free} = 0.99$, respectively. In Section 2.5.3, it was found always optimal to set $\lambda = 0.97$ as an adaptive model's forgetting factor. A different λ is chosen for the 2nd adaptive model, allowing the two models to have different adaptive properties (*e.g.* convergence rate and accuracy of estimates), thus further exploiting the benefit of multiple models.

Performance index B, (3.5), uses weight values $d_1 = 1$ and $d_2 = 2$ as defined in Section 3.4. In this section $d_3 = 0.95$ is used and performance index B can be represented as

$$J_i(k) = \frac{1}{k} \left[e_i^2(k) + 2 \sum_{\tau=1}^k 0.95^{k-\tau} e_i^2(\tau) \right]. \quad (3.66)$$

At initialization, model M_{fixed1} is defined to be initially selected (*i.e.* $i(0) = 1$). The conditions affecting both simulations (*i.e.* the noise signal, $w(k)$, reference signal, $y^*(k+1)$, and the parameter jump) are shown in Figure 3.7.

The behavior of the single model control system can be seen in Figure 3.8. The signals are initialized as $y^*(0) = 0$, $y(0) = 0$, and $e_c(0) = 0$. Transient control error and convergence occurred once the reference signal, $y^*(k+1)$, excited the system (*i.e.* once it moved away from $y^*(0) = 0$). At time-step $k = 500$, the plant parameters jumped and the control error, $e_c(k)$, became large in magnitude. When compared to the simulation's initialization (the first convergence), convergence occurred significantly slower to the unknown parameters at time-step $k = 500$ (second convergence). This is because the covariance matrix was large during initialization, permitting large/fast changes in parameter estimates. Whereas at time-step $k = 500$, the covariance matrix was smaller and the parameter estimate calculations incorporate data observed prior to time-step $k = 500$. The prior data was not appropriate for use because the parameters had changed at $k = 500$.

The result of performing multiple model control in the same control scenario (same initial conditions and disturbances) is shown in Figure 3.9. The performance index model switching signal, $i(k)$, is plotted in Figure 3.10. Upon initialization, the fixed model M_{fixed1} was selected (Figure 3.10) and the adaptive model M_{reset} used (initial) parameters taken from model M_{fixed1} . Model M_{reset} was then selected at time-step $k = 1$. Frequent switching occurred between the models M_{reset} and M_{free} prior to $k = 500$. This was because they possessed parameter estimates similar to the plant's parameters, θ^T in (3.63), and thus equal to each other (*i.e.* $\hat{\theta}_{reset}^T(k) \approx \hat{\theta}_{free}^T(k) \approx \theta^T$). When models have similar parameter estimates, this translates to them having similar performance index values, and frequent switching. At time-step $k = 410$ the fixed

model M_{fixed1} was accidentally selected. There appeared to be no adverse affect on the control error at time-step $k = 410$ (in Figure 3.9).

At time-step $k = 501$, the effects of the parameter jump at $k = 500$ made the multiple model controller select fixed model M_{fixed2} . Having reinitialized with estimates taken from M_{fixed2} , model M_{reset} was selected next.

During this initialization, the control error, $e_c(k)$, had nearly imperceptible transient components (in Figure 3.9). This was not the case when simulating the single model controller for the same control scenario (in Figure 3.8). When compared to the single model case, the multiple model controller performed significantly better when subjected to the parameter jump at time-step $k = 500$. The resultant control error, $e_c(k)$ after $k = 500$ in Figure 3.9, was both smaller in magnitude and faster to converge. The fixed models clearly attributed to this desirable performance.

Figure 3.7: The disturbance, $w(k)$, reference, $y^*(k+1)$, and change in plant states (used when simulating both the single and multiple model control cases).

Figure 3.8: Control using a single model, with a jump in parameters at time-step $k = 500$.

Figure 3.9: Control using the classical multiple model arrangement, with a jump in parameters at time-step $k = 500$.

Figure 3.10: Model selections according to performance index B switching (corresponding to multiple model control in Figure 3.9.)

3.6.4 Comparing Performance of the Classical and Modified Model Arrangement with Performance Index B Switching

The classical model arrangement and modified model arrangement are independently tested in an identical control scenario (*i.e.* the initial conditions and disturbances are identical). Both arrangements switch models using performance index B, (3.5), and are initialized as described in Section 3.6.1.

The plant parameter vector, $\theta^T(k)$, is piece-wise constant in time as follows:

$$\theta^T(k) = \begin{cases} \theta_1^T &= [0.6 \ 0.2 \ 1.0 \ 0.5 \ -0.5 \ 0.4], \quad k \leq 300, \\ \theta_2^T &= [-0.5 \ 0.2 \ 1.0 \ 0.2 \ -0.5 \ 0.4], \quad 300 < k \leq 600, \\ \theta_3^T &= [1.7 \ 0.7 \ 1.0 \ -0.5 \ 0.5 \ 0.4], \quad 600 < k \leq 900. \end{cases} \quad (3.67)$$

This plant is composed of three arbitrarily chosen stable plant states that satisfy *Assumptions (A1)-(A4)* in Section 1.4. The noise signal, $w(k)$, is zero mean Gaussian with variance $\sigma^2 = 0.2$. The simulation terminates at time-step $k = 900$.

The classical and modified model arrangements use the same three fixed models: M_{fixed1} , M_{fixed2} , and M_{fixed3} . These models are implemented to be in the neighborhoods of the three plant states. Using the values $\delta_{23}, \dots, \delta_{40}$ uniformly distributed on the interval $[-1, 1]$, the fixed models' parameter estimate vectors are defined as:

$$\hat{\theta}_{fixed1}^T = \theta_1^T + 0.20[\delta_{23} \ \delta_{24} \ \delta_{25} \ \delta_{26} \ \delta_{27} \ \delta_{28}], \quad (3.68)$$

$$\hat{\theta}_{fixed2}^T = \theta_2^T + 0.20[\delta_{29} \ \delta_{30} \ \delta_{31} \ \delta_{32} \ \delta_{33} \ \delta_{34}], \quad (3.69)$$

$$\hat{\theta}_{fixed3}^T = \theta_3^T + 0.20[\delta_{35} \quad \delta_{36} \quad \delta_{37} \quad \delta_{38} \quad \delta_{39} \quad \delta_{40}]. \quad (3.70)$$

The resetting and free-running adaptive models are implemented and initialized as described in Section 3.6.3.

Both model arrangements are exhaustively tested with this one control scenario (initial conditions and disturbances), using all possible values for the performance index B tuning weight, d_3 . For each model arrangement, the one control scenario is independently simulated 100 times, each time using a different d_3 value. The d_3 values used are: 0.01, 0.02, ..., and then 1.00. In each simulation, the noise is generated using the same random number generation seed (thus the $w(k)$ sequence is the same). The MSE of each tuning attempt's resultant control error, $e_c(k)$, is recorded after each simulation. This tuning process is performed for both the classical and modified model arrangements independently. The 100 recorded MSE values for tuning the classical arrangement are plotted as a solid line in Figure 3.11. The 100 recorded MSE values for tuning the modified arrangement are plotted as a dashed line in Figure 3.11.

It is clear that both model arrangements had similar performance in this control scenario. Using small values for the performance index weight, d_3 , large MSE was obtained. As shown in Figure 3.11, both model arrangements yielded their lowest control error MSE when simulated using $d_3 = 0.8$.

Figure 3.11: The repetitive tuning process for both the classical and modified model arrangement.

3.6.5 Comparing Performance of the Classical and Modified Model Arrangement with Incorrect Switching

With proper model placement and switching logic implementation stability can be achieved using either the classical or modified model arrangement. This section demonstrates how each model arrangement performs when the switching logic is not ideal, but arbitrary. An arbitrary model switching sequence, $i(k)$, is applied to the classical model arrangement and modified model arrangement, using two separate simulations. The disturbance, plant parameter vector, adaptive models, and fixed models are implemented as described in Section 3.6.4. The simulation terminates at time-step $k = 250$. There are no parameter jumps because (3.63) defines the first jump to be at $k = 300$, after the simulation terminates. As a result, the plant parameter vector is time-invariant and defined $\theta^T = \theta_1^T$.

The behavior of the classical model arrangement is plotted in Figure 3.12. As indicated in the plot of the signal $i(k)$, model M_{reset} was initially chosen. The selection of model M_{reset} and application of its corresponding controller, C_{reset} , yielded bounded signals and convergence of the control error, $e_c(k)$. Model M_{fixed2} was arbitrarily selected as time-step $k = 25$, causing application of the unmatched controller C_{fixed2} . This controller-plant combination yielded unbounded control error, $e_c(k)$. The same outcome is observed at time-step $k = 50$, when model M_{fixed1} was selected. Throughout the entire simulation, only selection of model M_{reset} and M_{free} yielded bounded signals and convergence.

The behavior of the modified model arrangement in the same control scenario (same initial conditions, switching sequence, and disturbance) is plotted in Figure 3.13. Initially, the model M_{reset} was chosen, and its corresponding controller, C_{reset} ,

was applied to control the plant. This controller-plant combination yielded bounded signals and convergence of control error, $e_c(k)$. Model M_{fixed2} was selected at time-step $k = 25$, causing re-initialization of model M_{reset} . This switch to a fixed model did not affect the application of controllers (controller C_{reset} remained applied because this is the modified model arrangement). Transient control error occurred while model M_{reset} converged again. When model M_{fixed1} was selected as time-step $k = 50$, almost no error occurred. This was because model M_{reset} was reinitialized with new estimates, θ_{fixed1}^T , very close to the actual plant parameters, θ_1^T .

When an adaptive model was selected, the modified model arrangement and the classical model arrangement behaved the same way. When a fixed model was selected, the modified model arrangement maintained stable control, whereas the classical model arrangement became unstable. This demonstrates that the stability of the classical model arrangement is dependent on the model selection logic. There was no such stability constraint demonstrated by the modified model arrangement. These same conclusions were also made in the stability analyses performed in Section 3.5.

Figure 3.12: The behavior of the classical model arrangement with arbitrary switching of signal $i(k)$ (reference signal shown as dashed line).

Figure 3.13: The behavior of the modified model arrangement with arbitrary switching of signal $i(k)$ (reference signal shown as dashed line).

Chapter 4

Hypothesis Test Switching

4.1 Introduction

A hypothesis test is useful for parameter jump detection when coupled with a system identifying Kalman filter [23, 24]. Using the normalized Kalman residual as a Gaussian test statistic, a parameter jump is strategically responded to by reinitializing the filter with random initial estimates. This generally allows faster convergence to any new parameter values. A more thorough discussion of how the hypothesis test is used with the Kalman filter is given in Section 1.1.2.

In this paper, an n -sample hypothesis test is proposed to facilitate the model switching for stochastic multiple model control. When the hypothesis test detects a jump in plant parameters it enables a new model to be selected. The model with the least plant estimate error during the n -samples of the hypothesis test will become selected. This switching method is suitable for use with both the classical and modified model arrangements (defined in Chapter 3). In both arrangements, the resetting adaptive model uses initial parameter estimates taken from any selected fixed model.

The hypothesis test switching method is defined in Section 4.2. The stability of the switching method is then analyzed in Section 4.3. In Section 4.4, many simulations are used to demonstrate hypothesis test switching and compare it to performance index switching.

4.2 Definition of the Hypothesis Test

The null hypothesis is declared as:

H_0 : No plant parameter jump has occurred.

When the stochastic plant is controlled by means of a perfectly matched controller, the control error sequence, $e_c(k)$, calculated using (2.11) in Section 2.3, is a Gaussian statistic with zero mean. In the event of a parameter jump, the change in statistics can be observed in the t-test result $t_0(k)$, calculated as

$$t_0(k) = \frac{\sum_{\tau=0}^{n-1} |e_c(k - \tau)|}{S(k)/\sqrt{n}}. \quad (4.1)$$

The user-defined constant n is the sample size of past observed control error, $e_c(k)$. The running standard deviation, $S(k)$, is calculated as

$$S^2(k) = \frac{\sum_{\tau=0}^{r-1} (e_c(k - \tau) - \bar{e}_c(k))^2}{r - 1}, \quad (4.2)$$

where the sample mean control error, $\bar{e}_c(k)$, is calculated as

$$\bar{e}_c(k) = \frac{1}{r - 1} \sum_{\tau=0}^{r-1} e_c(k - \tau). \quad (4.3)$$

The running standard deviation sample size is denoted by r . The running sample size, r , increases by one at each time-step k . When the null hypothesis, H_0 , is rejected, r is reset to the hypothesis testing sample size, n ($r \geq n$ always). Without the rejection of H_0 , the standard deviation, $S(k)$, will saturate at the user-defined maximum, S_{sat} . The value of S_{sat} is defined for use with simulations in Section 4.4.

A binary decision rule is used to compare the t-test result, $t_0(k)$, to a user-defined rejection threshold, denoted by T_r . The value of T_r corresponds to a significance level, α , using statistical tables. This is a one tailed test of significance. The decision rule for the hypothesis test is stated as:

1. if $t_0(k) \leq T_r$, accept H_0 (no parameter jump).
2. if $t_0(k) > T_r$, reject H_0 (a parameter jump).

When H_0 is accepted, switching is not permitted and the resetting adaptive model, M_{reset} , continues parameter estimation (whether or not it is the currently selected model). The free-running adaptive model, M_{free} , is never affected by the result of the hypothesis test.

When H_0 is rejected, the model i with the smallest associated performance index C , $J_i(k)$, $i \in \{1 \dots N\}$, defined by (4.4), becomes selected. If the newly selected model is a fixed model, the resetting adaptive model will reinitialize as described in Section 3.2. In the event that the newly selected model is the same as the previously selected model, no change will have occurred (*i.e.* the resetting model will continue adapting as if H_0 was never rejected).

The model selection index enabled by rejection of the null hypothesis, performance index C , is defined as

$$\text{C: } J_i(k) = \frac{1}{n} \sum_{\tau=0}^n e_i^2(k - \tau), \quad (4.4)$$

where n is the number of samples used in the t-test (4.1). Index $J_i(k)$ utilizes the past n time-steps' model estimation errors, $e_i(k), e_i(k - 1), \dots, e_i(k - n)$. Using a larger sample size, n , will increase the accuracy of the hypothesis test, at the cost of reducing the speed in which a jump can be detected [23, 24].

The following statistical terms are defined in the context of this implementation of hypothesis testing:

- α — probability of falsely detecting a parameter jump (a Type I error).
- β — probability of missing a parameter jump (a Type II error).

With the proposed binary decision rule, a reduction in α will result in an increase in β . It is worth emphasizing that a falsely detected parameter jump enables, but does not force, a model switch. This is because the comparison of each model's performance index C, (4.4), could determine that the current model is still the most suitable model. The case where a falsely detected parameter jump causes no performance-detrimental switching is demonstrated in Section 4.4.2. As a result, it is best to allow a large α when forced to implement hypothesis test switching when there is little *a priori* knowledge concerning the control problem. *A priori* knowledge concerning the control problem is knowledge regarding the statistical properties of the noise disturbance, $w(k)$, the current and future values of the plant parameter vector, $\theta^T(k)$, and the placement of any employed fixed models.

4.3 Stability

It is necessary to prove that both the classical model arrangement and modified model arrangement converge asymptotically with bounded signals when hypothesis test switching is used. As in the single model case, it is assumed that the transfer function $[1/C(q^{-1}) - \frac{1}{2}]$ is SPR. This requirement is more thoroughly discussed by *Assumption (A1)* in Section 1.4. The definition of SPR is provided in Appendix A.2.

Because the hypothesis test serves to enable and disable switching according to a performance index, stability is shown by providing extensions to the arguments used for performance index switching in Section 3.5. The stability of the classical model arrangement and modified model arrangement is analyzed, independently, in different sections. In Section 4.3.1, the argument for stability is given when control is performed using the classical model arrangement with hypothesis test switching. The conditions for stability of the classical model arrangement, as defined in Section 3.5.5, are also found to be necessary for classical model arrangement with hypothesis testing. In Section 4.3.2, stability analysis is then performed when hypothesis test switching is employed with the modified model arrangement. In this case, analysis of the applied switching logic is no longer necessary and the short stability argument used in Section 3.5.6 gives proof of stability.

4.3.1 Stability of the Classical Model Arrangement with Hypothesis Test Switching

The purpose of the hypothesis test is to enable and disable switching according to performance index C , (4.4). Analysis is first performed for *Case (1)*, where the user-defined value for α is “too large.” This results in continual false alarms and

performance index C switching, (4.4), is always enabled. Then *Case (2)* is considered, where the chosen value for α is 'too small' and any parameter jumps are in jeopardy of being missed.

Case (1) - Large α :

When the significance level, α , is considerably large, the null hypothesis, H_0 , is always rejected and performance index C switching, (4.4), is always enabled to perform switching. In this case, extensions are made to arguments from Section 3.5.5, where analysis of performance index A was performed. The extensions are made to use the same arguments for performance index C.

Theorem 5:

There exists a finite time-step K , and lower bound $\epsilon > 0$, such that

$$J_f(k) - J_a(k) > \epsilon > 0, \quad \forall k \geq K,$$

where $J_f(k)$ and $J_a(k)$ are the performance indices C, (4.4), for the fixed models and adaptive models respectively.

Conditional Proof of Theorem 5:

Theorem 5 is very similar to Theorem 2 in Section 3.5.5. It is different only because it uses performance index C instead of A. Because performance index C, (4.4), incorporates a fixed window size of n past identification errors, it cannot be guaranteed that switching will converge to any single model. Condition: If this user-defined window size n , used in performance index C, included all past identification errors ($n = k$), then C, (4.4), becomes equivalent to A, (3.4). Convergence, using performance index A, is demonstrated in Section 3.5.3. Given this assumption, $n =$

k , Theorem 5 and 2 become identical, and switching stability is achieved for *Case (1)*. However, with the assumptions for *Case (1)* (α user-defined to be very large), hypothesis test switching reduces to a performance index based switching method, which is not the intended use of the switching method.

Case (2) - Small α :

When the applied significance level, α , is too small, a jump in plant parameters may be responded to “too slowly” or not at all. A jump in parameters could possibly create an unstable closed-loop controller-plant combination. Should the newly applied controller correspond to a fixed model, this unstable combination cannot be allowed to become “frozen.” The ensuing proof of stability serves to show that no unstable closed-loop controller-plant combination can become “frozen.” In other words, the null hypothesis, H_0 , will be rejected at some time-step, K , permitting a model switch. As defined in Section 4.2, H_0 is rejected whenever the t-test result, $t_0(k)$, is greater than the threshold, T_r (associated with the significance level, α).

Assumptions:

- (C1) An unstable controller-plant combination yields control error that is increasing in magnitude, such that $|e_c(k)| \rightarrow \infty$, as $|k| \rightarrow \infty$.
- (C2) After one or more rejections of the null hypothesis, H_0 , an adaptive model is selected.

Theorem 6:

There exists a finite time-step K , such that

$$t_0(K) \geq T_r, \quad 0 < K < \infty.$$

Proof of Theorem 6:

The proof is stated in steps as follows. With *Assumption (C1)*, the running standard deviation defined by (4.2), $S(k)$, increases as follows

$$S(k) \rightarrow S_{sat}, \text{ as } |e_c(k)| \rightarrow \infty,$$

where the user-defined maximum standard deviation, S_{sat} , is defined and discussed in Section 4.2.

With $S(k) = S_{sat}$ (user-defined: $S_{sat} < \infty$) the t-test (4.1) increases according to

$$\frac{\sum_{\tau=0}^{n-1} |e_c(k-\tau)|}{S_{sat}/\sqrt{n}} \rightarrow \infty, \quad \text{as } |e_c(k)| \rightarrow \infty,$$

for any user-defined sample size $n > 0$.

Thus, there exists some time-step, K , where the t-test result $t_0(K) \geq T_r$, and model switching is permitted. Hence, there can be no unstable “frozen” controller-plant combination and, upon the selection of an adaptive model (given by *Assumption (C2)*), the stability conditions for single model adaptive control, discussed in Section 2.4, can be achieved.

If the user-defined sample size, n , was large (*i.e.* $n = k$), switching convergence will occur. This is stated because, with $n = k$, performance index C is equivalent to A, for which convergence is shown in Section 3.5.3. Despite the presence of noise, convergence to a stable controller-plant combination (fixed or adaptive) is guaranteed, but convergence to an adaptive model can only be guaranteed for large signal-to-noise

ratios. This condition to stability of the classical model arrangement is discussed more thoroughly in Section 3.5.5.

4.3.2 Stability of the Modified Model Arrangement with Hypothesis Test Switching

In the design of the modified model arrangement, it is intrinsic that there can be no unstable “frozen” controller-plant combination. In Section 3.5.6, stability was shown for the modified model arrangement with any switching method. The overall system is convergent for an model selection. In other words, the problems of performance and stability are decoupled.

4.4 Simulations

Independent sections are used to introduce each unique topic for simulation and analysis. The initial and operating conditions common to all of the simulations performed are described in Section 4.4.1. A demonstration of hypothesis test switching with details of the switching logic is given in Section 4.4.2. The hypothesis test switching method is then compared to the flexible performance index B switching method in Sections 4.4.2-4.4.7. The multiple sections are necessary to demonstrate and analyze the many behavioral differences between the two switching methods.

4.4.1 Conditions Common to All Simulations

The stochastic plant, certainty equivalence controller, and adaptive models are implemented as defined for the simulation conditions in Section 3.6.1. The stochastic

plant to be controlled is described by (3.54) and (3.55). The multiple model control initial conditions described by (3.57)-(3.62) are used.

The desired plant output is described by the persistently exciting function,

$$y^*(k+1) = \sin(2\pi k/55) + \sin(2\pi k/165). \quad (4.5)$$

The plant parameter vector, $\theta^T(k)$, is piece-wise constant in time as shown in Figure 4.1 and defined by

$$\theta^T(k) = \begin{cases} \theta_1^T & \text{for } 0 < k \leq 200, \quad 600 < k \leq 800, \quad 1200 < k \leq 1400, \\ \theta_2^T & \text{for } 400 < k \leq 600, \quad 800 < k \leq 1000, \\ \theta_3^T & \text{for } 200 < k \leq 400, \quad 1000 < k \leq 1200, \end{cases} \quad (4.6)$$

where the three plant states are defined as:

$$\hat{\theta}_1^T = [\delta_{41} \quad \delta_{42} \quad 0.5 \quad \delta_{43} \quad \delta_{44} \quad \delta_{45}], \quad (4.7)$$

$$\hat{\theta}_2^T = [\delta_{46} \quad \delta_{47} \quad 1.0 \quad \delta_{48} \quad \delta_{49} \quad \delta_{50}], \quad (4.8)$$

$$\hat{\theta}_3^T = [\delta_{51} \quad \delta_{52} \quad 1.5 \quad \delta_{53} \quad \delta_{54} \quad \delta_{55}]. \quad (4.9)$$

The values, $\delta_{41}, \dots, \delta_{55}$, are uniformly distributed on the interval $[-1, 1]$. These three arbitrarily chosen plant states are stable and satisfy *Assumptions (A1)-(A4)* stated in Section 1.4. This is a relaxation of *Assumption (A5)*, which stated that the parameters were time-invariant.

As shown in Figure 4.1, (4.5) and (4.6) are defined such that no parameter jump occurs when the desired plant output is small (*e.g.* $y^*(k+1) = 0$). This is intended

to make any transient control error, due to a jump in plant parameters, large and easy to observe.

Both the classical model arrangement and modified model arrangement are implemented for testing in Sections 4.4.2-4.4.7. These two arrangements contain three fixed models, M_{fixed1} , M_{fixed2} , and M_{fixed3} . These models have their respective fixed parameter estimate vectors, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, $\hat{\theta}_{fixed3}^T$, defined to be in the neighborhood of the three plant states, θ_1^T , θ_2^T , and θ_3^T , from (4.6). When a model is in the “neighborhood” of a plant state, this implies that the model’s parameters are similar to those of the plant state, with some parameterization error. The maximum parameterization error allowed to exist between the fixed models and the actual plant states is user defined by the scalar μ , used in (4.10)-(4.12). Using the values $\delta_{56}, \dots, \delta_{73}$, uniformly distributed on the interval $[-1, 1]$, the three fixed models’ parameter estimate vectors are defined as:

$$\hat{\theta}_{fixed1}^T = \theta_1^T + \mu[\delta_{56} \quad \delta_{57} \quad \delta_{58} \quad \delta_{59} \quad \delta_{60} \quad \delta_{61}], \quad (4.10)$$

$$\hat{\theta}_{fixed2}^T = \theta_2^T + \mu[\delta_{62} \quad \delta_{63} \quad \delta_{64} \quad \delta_{65} \quad \delta_{66} \quad \delta_{67}], \quad (4.11)$$

$$\hat{\theta}_{fixed3}^T = \theta_3^T + \mu[\delta_{68} \quad \delta_{69} \quad \delta_{70} \quad \delta_{71} \quad \delta_{72} \quad \delta_{73}]. \quad (4.12)$$

The plant states θ_1^T , θ_2^T , and θ_3^T are defined in (4.7)-(4.9). The actual value of the maximum parameterization error, μ , is uniquely defined in each simulation section.

Using (4.6)-(4.12), unknown plant parameter jumps are generated and it is guaranteed that a fixed model will be in the neighborhood, with error μ , of each new plant state after a jump. Such new and unknown control scenarios are easily generated for

use in Sections 4.4.2-4.4.7. The applied switching methods are tested and tuned in these generated control scenarios. Although the parameters always jump the times stated by (4.6), the controllers do not assume that the jump times are known.

The initialization of the hypothesis test switching algorithm must also be considered. As discussed in Section 3.6.1, the initial (at the beginning of each simulation) model estimation errors, $e_1(k)$, $e_2(k)$, $\dots$, $e_N(k)$, are zero. As a result, the associated performance indices C , $J_1(k)$, $J_2(k)$, $\dots$, $J_N(k)$, are also equal and the minimum performance index, $J_i(k)$, where,

$$J_i(k) = \min_{j=1,\dots,N} J_j(k),$$

cannot be found. As a result, it is necessary to force a model to be selected at time-step $k = 0$ so that the certainty equivalence control signal, $u(k)$, can be calculated for time-step $k = 1$. The initial model selection signal, $i(0)$, is user-defined within each simulation section.

Single-sample hypothesis testing (*i.e.* $n = 1$) is implemented to provide the fastest possible detection. The user-defined maximum running standard deviation value, S_{sat} used with (4.2), is defined to be $S_{sat} = 0.20$.

In this thesis, the term “control scenario” is used to describe the initial and operating conditions of a control simulation. A unique control scenario has one or all of the following attributes being unique:

- disturbance sequence, $w(k)$,
- set of plant states, θ_1^T , θ_2^T , and θ_3^T , which corresponds to $\theta^T(k)$ being unique because plant states change as defined by (4.6),
- and placement of fixed models, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$.

Consequently, when a new control scenario is defined, $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$ change. This is done by using a unique seed for generating the disturbance, $w(k)$, and the random numbers, $\delta_{41}, \dots, \delta_{73}$, used in (4.7)-(4.12). In some simulation sections, similar control methods are being compared. This is performed by using same control scenario when simulating each method, so that two method's behavior can be more directly compared.

Figure 4.1: The behavior of the desired plant output and the plant parameter vector during all of the Section 4.4 simulations.

4.4.2 Detailed Demonstration of Hypothesis Test Switching

This straightforward example demonstrates how multiple model adaptive control is performed using hypothesis test switching with a rejection threshold value $T_r = 3.5$. The reference signal, $y^*(k+1)$, and predetermined distribution of parameter jumps are both defined in Section 4.4.1 and plotted in Figure 4.1. The simulation terminates at time-step $k = 400$. As a result, switching behavior is observed during initialization and one parameter jump. The disturbance signal, $w(k)$, is zero mean Gaussian with variance $\sigma^2 = 0.05$.

The classical model arrangement is used and model M_{fixed1} is defined to be initially selected (*i.e.* $i(0) = 1$). The maximum parameterization error allowed to exist between the three fixed plant models and plant states, μ in (4.10)-(4.12), is defined as $\mu = 0.3$.

The adaptive models M_{reset} and M_{free} use the forgetting factor values $\lambda_{reset} = 0.94$ and $\lambda_{free} = 0.97$, respectively. In Section 2.5.3, it was found always optimal to set $\lambda = 0.97$ as an adaptive model's forgetting factor. A different λ is chosen for the 2nd adaptive model, allowing the two models to have different adaptive properties (*e.g.* convergence rate and accuracy of estimates), thus further exploiting the benefit of multiple models.

The resulting control error, $e_c(k)$, and details of the hypothesis test switching logic are plotted in Figure 4.2. The initial control error is observed to be $e_c(0) = 0$ because the desired plant output and actual plant output are initialized as $y^*(0) = 0$ and $y(0) = 0$. Transient control error occurred once the reference signal $y^*(k+1)$ excited the system (*i.e.* once it moved away from $y^*(0) = 0$). The selected model M_{fixed1} was not sufficient to allow for zero mean Gaussian control error, $e_c(k)$. The running standard deviation of $e_c(k)$ increased with each time-step. The t-test compares $e_c(k)$

to the desired mean of zero, and the t-test result, $t_0(k)$, also increased in magnitude with each time-step. At time-step $k = 11$, the t-test result, $t_0(k)$, became larger than the magnitude of the user-defined rejection threshold value ($T_r = 3.5$, shown as a dashed line) and the null hypothesis, H_0 , was rejected. This allowed the adaptive model M_{reset} to become selected.

At time-step $k = 91$, the null hypothesis, H_0 , was rejected and performance index C switching determined that model M_{reset} was still the most suitable model. This demonstrated that a falsely detected parameter jump does not force performance-detrimental switching to occur, as discussed analytically in Section 4.2.

At time-steps $k = 124$ and $k = 182$, the null hypothesis was rejected. At $k = 124$, the adaptive model M_{free} was selected, and at $k = 182$, the adaptive model M_{reset} was selected again. Model M_{reset} was not reinitialized during these two switches because no fixed model was ever selected. Because both of the adaptive models had converged, prior to these two switches, to provide similar plant parameter estimates, the control error, $e_c(k)$, was unaffected by the switching.

At time-step $k = 200$, a jump in plant parameters occurred. The control error sample $e_c(200)$ caused the single-sample hypothesis test to reject H_0 , resulting in the selection of model M_{fixed2} . At the next time-step, $k = 201$, the null hypothesis was again rejected and model M_{reset} , having been reset using initial estimates taken from model M_{fixed2} , was again selected. This demonstrates the classical model arrangement's ideal response to a parameter jump [12]. This is an ideal response because a suitable fixed model was temporarily utilized, and the resetting adaptive model was next used to provide even better parameter estimates and control.

Figure 4.2: A detailed example of hypothesis test switching (rejection threshold, T_r , shown as dashed line).

4.4.3 Hypothesis Test Switching with Poor Choices for Rejection Threshold, T_r

This example demonstrates the system behavior due to two improper implementations of hypothesis test switching. The consequences of having a rejection threshold T_r that is “too large” and “too small” are demonstrated. A “too large” T_r value causes parameter jumps to go undetected and a “too small” T_r value causes the null hypothesis to be unnecessarily and frequently rejected. The two extreme choices for T_r are considered:

- $T_r = \infty$,
- $T_r = 0$.

The classical model arrangement and modified model arrangement are tested with these two poor rejection threshold values. The adaptive models M_{reset} and M_{free} use the forgetting factor values $\lambda_{reset} = 0.94$ and $\lambda_{free} = 0.97$, respectively. In Section 2.5.3, it was found always optimal to set $\lambda = 0.97$ as an adaptive model’s forgetting factor. A different λ is chosen for the 2nd adaptive model, allowing the two models to have different adaptive properties (*e.g.* convergence rate and accuracy of estimates), thus further exploiting the benefit of multiple models.

The disturbance, $w(k)$, is zero mean Gaussian with variance $\sigma^2 = 0.05$. The maximum parameterization error allowed to exist between the three fixed plant models and plant states, μ in (4.10)-(4.12), is defined as $\mu = 0.3$.

All simulations are performed using an identical control scenario (*i.e.* $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$ are the same in each simulation. The definition of a control scenario is provided with more detail in Section 4.4.1. In each simulation, model M_{fixed1} is forced to be initially selected (*i.e.* $i(0) = 1$).

Analysis when $T_r = \infty$:

The control error, $e_c(k)$, and hypothesis test switching signal, $i(k)$, produced by the classical model arrangement with $T_r = \infty$ are plotted in Figure 4.3. The same for the modified model arrangement is plotted in Figure 4.4. The model M_{fixed1} was initially selected and remained selected throughout these two arrangements' simulations because the t-test result could never become $t_0(k) = \infty$ for time-steps $k = 1$ to $k = 1400$ (end of simulation). Model M_{fixed1} did not have a parameter estimate vector, $\hat{\theta}_{fixed1}^T$, sufficiently close to plant parameter vector θ_1^T , θ_2^T , or θ_3^T . As a result, the classical model arrangements never achieved zero mean Gaussian control error. With the modified model arrangement, controller C_{reset} was applied when model M_{fixed1} was selected. As a result, there was very little initial transient control error and zero mean Gaussian control error was quickly achieved. At each parameter jump, large transient control error was observed because there was no switching or resulting re-initialization of model M_{reset} , which corresponds to controller C_{reset} .

Analysis when $T_r = 0$:

The control error, $e_c(k)$, and hypothesis test switching signal, $i(k)$, produced by the classical model arrangement with $T_r = 0$ are plotted in Figure 4.5. The same for the modified model arrangement is plotted in Figure 4.6. With $t_0(k) \leq 0$ for all k , the null hypothesis, H_0 , was constantly rejected and performance index C switching occurs unrestrained during all time-steps k . Performance index C, (4.4), was proposed in this thesis for use with the hypothesis test but it is similar to index A, (3.4), used in [11]. Both model arrangements had identical performance because the behavior of M_{reset} is what makes the model arrangements unique and M_{reset} was never allowed

sufficient time to adapt after it is selected. Switching did not converge to adaptive model M_{reset} or M_{free} during the simulated time-steps. It is clearly detrimental to the performance of the overall system when only performance index C is used for switching.

Figure 4.3: The classical model arrangement with $T_r = \infty$.

Figure 4.4: The modified model arrangement with $T_r = \infty$.

Figure 4.5: The classical model arrangement with $T_r = 0$.

Figure 4.6: The modified model arrangement with $T_r = 0$.

4.4.4 Comparing Performance Index B Switching and Hypothesis Test Switching in the Absence of Disturbances

The performance index B and hypothesis test switching methods are independently tested with the classical model arrangement. The performance index B switching method is implemented as defined in Section 3.6.1 and uses the weight value $d_3 = 0$. Hypothesis testing is implemented with rejection threshold $T_r = 0$. The maximum parameterization error, μ , allowed to exist between the three fixed plant models and plant states is defined as $\mu = 0.5$.

The adaptive models M_{reset} and M_{free} use the forgetting factor values $\lambda_{reset} = 0.94$ and $\lambda_{free} = 0.97$, respectively. In Section 2.5.3, it was found always optimal to set $\lambda = 0.97$ as an adaptive model's forgetting factor. A different λ is chosen for the 2nd adaptive model, allowing the two models to have different adaptive properties (*e.g.* convergence rate and accuracy of estimates), thus further exploiting the benefit of multiple models.

Both switching methods are simulated using an identical control scenario (*i.e.* $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$ are the same in each simulation. The definition of a control scenario is provided with more detail in Section 4.4.1. In each simulation, model M_{reset} is initially selected (*i.e.* $i(0) = 4$).

The performance index B switching signal, $i(k)$, and associated control error, $e_c(k)$, are plotted in Figure 4.7. The hypothesis test switching signal, $i(k)$, and associated control error, $e_c(k)$, are plotted in Figure 4.8. It is clear that both switching methods achieve ideal and identical performance when a stochastic disturbance is not present. The tuning of these switching methods becomes necessary when the stochastic disturbance is present.

Figure 4.7: The classical model arrangement with performance index switching, $d_3 = 0$, and no disturbance, $w(k) = 0$.

Figure 4.8: The classical model arrangement with hypothesis test switching, $T_r = 0$, and no disturbance, $w(k) = 0$.

4.4.5 Comparing Performance Index B and Hypothesis Test

Switching with Noise Variance $\sigma^2 = 0.025$

In this section the classical and modified model arrangements are independently simulated using both performance index B and then hypothesis test switching methods. Performance index B switching, (3.5), is implemented as defined in Section 3.6.1.

The adaptive models M_{reset} and M_{free} use the forgetting factor values $\lambda_{reset} = 0.97$ and $\lambda_{free} = 0.98$, respectively. In Section 2.5.3, it was found always optimal to set $\lambda = 0.97$ as an adaptive model's forgetting factor. A different λ is chosen for the 2nd adaptive model, allowing the two models to have different adaptive properties (*e.g.* convergence rate and accuracy of estimates), thus further exploiting the benefit of multiple models.

The disturbance, $w(k)$, is zero mean Gaussian with variance $\sigma^2 = 0.025$. The maximum parameterization error, μ , allowed to exist between the three fixed models and plant states is defined as $\mu = 0.5$. All simulations use an identical control scenario (*i.e.* $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$ are the same in each simulation). The definition of a control scenario is provided with more detail in Section 4.4.1. In each simulation, model M_{reset} is initially selected (*i.e.* $i(0) = 4$).

Due to the presence of noise, tuning of the switching methods is required to achieve *desirable control error*. When evaluating the control error sequence, $e_c(k)$, the control error attributes that are desirable are defined below by *Definition 1.1*, *1.2*, and *1.3*.

Definition 1 - Desirable control error behavioral attributes:

- 1.1. The MSE of the entire simulation's control error sequence, $e_c(k)$, should be as small as possible.
- 1.2. The control error's transient period (after each jump) should last a minimum

number of time-steps, k .

1.3. The steady state control error should be Gaussian with zero mean.

In the following simulations, it is shown that the desirable control attribute *Definition 1.1* does not guarantee *Definition 1.2* or *Definition 1.3*. The reason why these three attributes are decoupled is made clear below in the relevant simulations' are discussions.

It is shown in this section that extensive tuning of the performance index switching method is necessary to find desirable control error. Tuning of the hypothesis test switching method is also performed for the same goal. It is shown later, in Section 4.4.6, that the tuning of the hypothesis test switching method is not necessary.

Tuning of the hypothesis test switching method involves simulating once using a certain rejection threshold, T_r , recording the MSE, then iteratively trying again with an adjusted value for T_r . One hundred such tuning attempts are made, using values for T_r between 0.01 and 10.0. Tuning attempt j uses the rejection threshold value

$$T_r = 0.01j, \quad \text{where } j = 1, 2, \dots, 100 \quad (4.13)$$

Tuning the performance index B switching method involves adjusting the index's associated weight d_3 in the same methodical manner. One hundred tuning attempts are made, using values for d_3 between 0.01 and 1.0. Tuning attempt j uses weight

$$d_3 = 0.001j, \quad \text{where } j = 1, 2, \dots, 100 \quad (4.14)$$

This process of tuning was first performed using hypothesis test switching with the classical model arrangement. After the 100 tuning attempts, the value $T_r = 1.4$

produced the smallest control MSE and was determined to be the best rejection threshold value.

Using this tuned value, $T_r = 1.4$, the resulting hypothesis test switching signal, $i(k)$, and associated control error, $e_c(k)$, are plotted in Figure 4.9. This plotted $e_c(k)$ sequence yields the lowest MSE achievable by hypothesis test switching of the classical model arrangement (desirable control error attribute *Definition 1.1*, defined above). Zero mean Gaussian control error should have been achieved between time-steps $k = 200$ and $k = 400$ (desirable control error attribute *Definition 1.3*). It is observed that desirable control error was not achieved. This tuning process was unsuccessful at finding desirable control error because it was only tuning to satisfy the desirable control error attribute *Definition 1.1*. The MSE calculation was biased by the large magnitudes of transient errors. As a result, the coloured Gaussian control error behavior between time-steps $k = 200$ and $k = 400$ could not be tested for.

The above tuning process was performed again using a modified control MSE calculation. The modified control MSE calculation is proposed and discussed in Appendix A.1. This method employs a user-defined outlier threshold $T_{e_c} = 1$ to identify, and apply a ceiling to, each single sample of transient control error, $e_c(k)$. The resulting MSE is without bias from extraordinarily large control error samples because they are bounded by $T_{e_c} = 1$.

This modified control error MSE method was also used for tuning the classical model arrangement with performance index B switching and the modified model arrangement with both switching methods. In other words, the modified MSE is used when tuning the following four combinations of switching methods and model arrangements:

1. Performance index B switching of the classical model arrangement.

2. Hypothesis test switching of the classical model arrangement.
3. Performance index B switching of the modified model arrangement.
4. Hypothesis test switching of the modified model arrangement.

The tuning process, for each of these four combinations of model arrangements and switching methods, is plotted in Figure 4.10.

The lowest MSE for the classical model arrangement with hypothesis test switching was found at tuning attempt $j = 31$ (*i.e.* $T_r = 3.1$). The resulting switching signal, $i(k)$, and associated control error, $e_c(k)$, are plotted in Figure 4.11. The lowest MSE for the classical model arrangement with performance index B switching was found at tuning attempt $j = 32$ (*i.e.* $d_3 = 0.32$). The resulting switching signal, $i(k)$, and associated control error, $e_c(k)$, are plotted in Figure 4.12. The lowest MSE for the modified model arrangement with hypothesis test switching was found at tuning attempt $j = 73$ (*i.e.* $T_r = 7.3$). The resulting switching signal, $i(k)$, and associated control error, $e_c(k)$, are plotted in Figure 4.13. The lowest MSE for the modified model arrangement with performance index B switching was found at tuning attempt $j = 32$ (*i.e.* $d_3 = 0.32$). The resulting switching signal, $i(k)$, and associated control error, $e_c(k)$, are plotted in Figure 4.14.

When compared to Figure 4.9, it is clear that Figures 4.11-4.14 contain more desirable control error sequences, $e_c(k)$, according to *Definition 1*.

In Section 3.6.4 it was shown that the classical and modified model arrangements have similar behavior when performance index B switching is used. This can be seen again when comparing Figures 4.12 and 4.14. This can also be seen during their entire tuning process, in Figure 4.10. The steady state control error, $e_c(k)$, produced by the two arrangements with performance index B switching (using $d_3 = 0.32$), represent

the most desirable control error found by tuning. These two best case outcomes are unsatisfactory because switching never converged to a single model before the next parameter jump occurred. Also, small control errors were introduced every time a fixed model was selected. As discussed in Section 3.5.4, using a value for d_3 closer to 1 would allow switching to converge to a single model but this would also slow the switching response (allowing for a longer transient period).

The modified model arrangement with tuned hypothesis test switching (Figure 4.13) did not switch models at time-step $k = 1200$. This is because the parameter jump at time-step $k = 1200$ did not cause sufficient control error, $e_c(k)$, to reject H_0 and warrant a change in models. The model applied at time-step $k = 1199$ remained applied after the parameter jump, and it converged to the new parameter values.

The modified model arrangement with hypothesis test switching produced the most desirable control error when compared to the other three simulated combinations of switching methods and model arrangements. In other words, its desirable switching attributes, stated in *Definition 1*, were the most satisfying.

Figure 4.9: The classical model arrangement with tuned hypothesis test switching, $T_r = 1.4$, achieving the lowest possible MSE of control error.

Figure 4.10: The tuning process, for the four combinations of model arrangements and switching methods.

Figure 4.11: The classical model arrangement with tuned hypothesis test switching, $T_r = 3.1$, achieving the lowest possible *modified* MSE of control error.

Figure 4.12: The classical model arrangement with tuned performance index B switching, $d_3 = 0.32$, achieving the lowest *modified* MSE of control error.

Figure 4.13: The modified model arrangement with tuned hypothesis test switching, $T_r = 7.3$, achieving the lowest *modified* MSE of control error.

Figure 4.14: The modified model arrangement with tuned performance index B switching, $d_3 = 0.32$, achieving the lowest *modified* MSE of control error).

4.4.6 Comparing Performance Index B and Hypothesis Test Switching with Noise Variance $\sigma^2 = 0.025$ and Unique Parameter Jumps and Models

This section is nearly identical to Section 4.4.5, where the four combinations of switching methods and model arrangements were tuned using the modified MSE calculation. In Section 4.4.5, only the same control scenario was used for all simulations. The definition of a control scenario is provided with more detail in Section 4.4.1. For this one control scenario the switching methods' tuning processes are depicted in Figure 4.10.

The purpose of this section is to show how the tuning process changes when plant parameter jumps and fixed model placements are different (*i.e.* when the control scenario changes). New control scenarios are defined so that $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$ are unique in each new control scenario. This is done by using a unique seed for generating the disturbance, $w(k)$, and the random numbers, $\delta_{41}, \dots, \delta_{73}$ (used in (4.7)-(4.12) for defining $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$). Fifty unique control scenarios are generated and the tuning procedure defined in Section 4.4.5 is performed independently with each control scenario.

Instead of independently plotting the tuning process for each control scenario (as depicted in Figure 4.10 for the single control scenario) a two-dimensional filled contour plot is used. Using one contour plot, the tuning process for all 50 control scenarios can be observed. Following this paragraph is an explanation of how to interpret the contour. The tuning process of the classical model arrangement with performance index B is plotted, as a contour, in Figure 4.15 (a). The tuning process of the classical model arrangement with hypothesis testing is plotted, as a contour,

in Figure 4.15 (b). The tuning process of the modified model arrangement with performance index B is plotted, as a contour, in Figure 4.15 (c). The tuning process of the modified model arrangement with hypothesis testing is plotted, as a contour, in Figure 4.15 (d). The contours in Figure 4.15 have information presented as follows:

- The unique control scenario numbers are labeled on the left vertical axis, as $\#1, \dots, \#50$. Each control scenario number in (a) is the same control scenario number in (b), (c), and (d) (*e.g.* scenario $\#1$ has the same $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$ in (a), (b), (c), and (d)).
- The value being changed during the 100 tuning attempts (d_3 for performance index B or T_r for the hypothesis test method) is labeled on the bottom horizontal axis.
- The outcome of each tuning attempt (*i.e.* for each control scenario $\#$ and switching method's value, d_3 or T_r) is plotted on the contour using colour to represent the resulting modified MSE value.
- The colours are interpreted as actual modified MSE values using the vertical legend on the right side of the figure.

The behavior of the classical model arrangement with performance index B, Figure 4.15 (a), will first be discussed. The tuning process was unique for each unique control scenario. For example:

- In control scenario $\#30$ (labeled on the vertical axis) a low MSE was achieved (dark blue colour) using $d_3 = 0.4$.
- In control scenario $\#29$, using $d_3 = 0.4$ resulted in a large MSE (red colour) and $d_3 = 0.02$ was found to produce lower MSE.

Because there is no way of knowing what value for d_3 is best given a new control scenario, without performing tuning, the performance index switching method is heuristic.

In Section 3.6.4 it was shown that the classical model arrangement and modified model arrangement have similar behavior when using performance index B switching. This can be seen again by observing how similar the contours of Figure 4.15 (a) and (c) appear. The contours are similar because, for each control scenario, the tuning process found the lowest modified MSE value using the same weight d_3 , in both contours. For example, in control scenario #30 low MSE values are achieved (dark blue colour) using $d_3 < 0.5$ (labeled on the vertical axis) in both Figures, (a) and (c).

The behavior of the classical model arrangement with hypothesis test switching, Figure 4.15 (b), will be discussed next. The tuning process was very similar in each unique control scenario. For example:

- In control scenario #1 (labeled on the left vertical axis) a low MSE (dark blue colour) was achieved using $T_r \leq 9$.
- In control scenario #2, using $T_r \leq 9$ also produced a low MSE.
- For all control scenarios a low MSE was achieved using $T_r = 5$. This is illustrated in (b) by a vertical dashed line.

The vertical line was drawn in Figure 4.15 (b) to illustrate how a low control MSE could be achieved for all control scenarios without adjustment of the threshold T_r (no such line could be drawn in Figures 4.15 (a) and (c) because heuristic switching was used). This trend line demonstrates that the hypothesis test switching method actually does not require tuning when plant parameter jumps and model placements

are unique or unknown. For example, if a control scenario #51 was generated, the sequence $w(k)$ and the parameters of $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$, would be unknown, but $T_r = 5$ would produce a low control error MSE. Because it is known that T_r is best for a new control scenario, without *a priori* knowledge, this switching method is not heuristic. The benefits to the hypothesis test switching method are further discussed in the concluding remarks, Chapter 5.

Similar observations can be made for the contour shown in Figure 4.15 (d) (hypothesis test switching of the modified model arrangement). For all control scenarios a low MSE was achieved using $T_r = 5$ (shown as a vertical dashed line). By comparing Figures 4.15 (b) and (d), it can be seen that both contours appear somewhat different. The difference occurs because in these two figures different model arrangements are used (classical *vs.* modified), and the hypothesis test switching method does not have the same behavior with both. This difference was discussed in more depth in Section 4.4.3.

Figure 4.15: The tuning process for 50 unique control scenarios (unique $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$). Contour (a) is classical arrangement with index B switching, (b) is classical arrangement with hypothesis test switching, (c) is modified arrangement with index B switching, and (d) is modified arrangement with hypothesis test switching.

4.4.7 Comparing Performance Index B and Hypothesis Test Switching with Changing Noise Variance, σ^2

This section is very similar to Section 4.4.6, where the four combinations of model arrangements and switching methods were independently tuned for 50 unique control scenarios. Each control scenario was unique because $w(k)$, $\theta^T(k)$, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, and $\hat{\theta}_{fixed3}^T$ were unique. But, in each of these control scenarios the noise variance was always $\sigma^2 = 0.025$. In this section, experimentation is performed using control scenarios with unique noise variance values, σ^2 . The plant parameter vector, $\theta^T(k)$, and fixed model parameter estimate vectors, $\hat{\theta}_{fixed1}^T$, $\hat{\theta}_{fixed2}^T$, $\hat{\theta}_{fixed3}^T$, are forced to be the same in each control scenario. The initial conditions for the simulations in this section are defined as in Section 4.4.5. The definition of a control scenario is provided with more detail in Section 4.4.1.

Fifty unique control scenarios are generated and the tuning procedure defined in Section 4.4.5 is performed independently with each control scenario. The 50 control scenarios use values for σ^2 between 0.0 and 0.3. In other words, scenario l uses the noise variance value

$$\sigma^2 = 0.006l, \quad \text{where } l = 1, 2, \dots, 50. \quad (4.15)$$

Instead of independently plotting the tuning process for each control scenario (as depicted in Figure 4.10 for the single control scenario) a two-dimensional filled contour plot is used. Using a contour plot, the tuning process for all 50 control scenarios can be observed. Following this paragraph is an explanation of how to interpret the actual contours. The tuning process of the classical model arrangement with performance index B is plotted as a contour in Figure 4.16 (a). The tuning process of the classical

model arrangement with hypothesis testing is plotted as a contour in Figure 4.16 (b). The tuning process of the modified model arrangement with performance index B is plotted as a contour in Figure 4.16 (c). The tuning process of the modified model arrangement with hypothesis testing is plotted as a contour in Figure 4.16 (d). The contours in Figure 4.16 have information presented as follows:

- The unique control scenario's variance, σ^2 , is labeled on the left vertical axis.
- The value being changed during the 100 tuning attempts (d_3 for performance index B or T_r for the hypothesis test method) is labeled on the bottom horizontal axis.
- The outcome of each tuning attempt (*i.e.* for each control scenario's variance, σ^2 , and switching method's value, d_3 or T_r) is plotted on the contour using colour to represent the resulting modified MSE value.
- The colour values can be interpreted as actual modified MSE values using the vertical legend on the right side of the contours.

In Section 3.6.4 it was shown that the classical model arrangement and modified model arrangement have similar behavior when using performance index B switching, (3.5). This can be seen again by observing how similar the contours in Figure 4.16 (a) and (c) appear. By comparing Figures 4.16 (b) and (d), it can be seen that both contours appear somewhat different. The difference occurs because in these two figures different model arrangements are used (classical *vs.* modified), and the hypothesis test switching method does not have the same behavior with both. This difference was discussed in Section 4.4.3.

The plots in Figure 4.16 show that the hypothesis test switching method achieves a lower modified MSE in the presence of larger noise variances, when compared to the

performance index switching method. This can be seen, for example, when comparing Figures 4.16 (c) and (d):

- At the control scenario with noise variance $\sigma^2 = 0.30$ (shown on the vertical axis), it can be seen in (c) that all tuning attempts ($d_3 = 0.5$ to 1.0) yielded an MSE larger than 1.00 (red colour).
- In this same control scenario in (d), the rejection threshold value $T_r = 9$ produced an MSE much less than 1.0 (yellow colour).

The hypothesis test switching method had similar behavior when implemented with the classical model arrangement, in Figure 4.16 (b). This example demonstrated that the hypothesis test switching method, when compared to performance index B, achieves lower control error MSE in presence of large disturbances. As a result the hypothesis test switching method is more suitable for model selection in stochastic environments.

Figure 4.16: The tuning process for 50 control scenarios of different noise variance, σ^2 . Contour (a) is classical arrangement with index B switching, (b) is classical arrangement with hypothesis test switching, (c) is modified arrangement with index B switching, and (d) is modified arrangement with hypothesis test switching.

Chapter 5

Conclusions and Recommendations

A performance index was chosen for comparison to the proposed hypothesis test switching method. This performance index is well-researched and popular because of its flexibility and ability to select models in stochastic environments. The proposed hypothesis test switching method allowed for better switching and less control error when compared to the performance index based switching method. Simulations showed that hypothesis test switching could be implemented, without *a priori* knowledge of the control environment, and the multiple model controller would reliably yield small control error. The performance index switching method, in comparison, did not possess this ease-of-implementation attribute and required repetitive tuning to achieve acceptable control error. As a result, the performance index was considered to be a heuristic switching method. In general, the hypothesis test switching method yielded lower control error than the performance index method. Future work should compare the hypothesis test switching method to other switching methods. In particular, it could be found similar to a hysteresis-based switching method, which has yet to be proposed for discrete-event systems. Furthermore, each type of switching method's

computational complexities could be investigated and compared.

The model arrangement used in other research was referred to as the classical model arrangement. It is the only current method for combining more than one model for control with switching. Narendra *et al.* [12], stated that “mathematically, fixed models only serve to provide better initial points from which the resetting model can adapt,” however, stability analyses in Section 3.5.5 and simulations in Section 3.6.5 demonstrated that this was, in fact, not the case. The arguments for stability required performance crippling assumptions and intricate analyses of any applied switching method. When the switching logic was not properly tuned, instabilities occurred. This thesis proposed a modification to the classical arrangement. The modified model arrangement’s stability analysis was significantly simpler and required no performance crippling assumptions or analyses of switching methods. The modified arrangement served to decouple stability and performance. As a result, switching methods that would compromise the classical arrangement’s stability could be blindly applied. In future work, such new and existing switching methods should be implemented to benefit from the modified model arrangement. When comparing the modified model arrangement to the classical model arrangement, computational complexity was the same, stability was improved, and performance increases were demonstrated.

Both, hypothesis test switching and the modified model arrangement contribute to improve the control of time-varying stochastic systems. The most significant performance and stability improvements were demonstrated when the two methods were combined to form a completely new method of multiple model adaptive control.

References

- [1] K. S. Narendra, O. A. Driollet, M. Feiler, and K. George, "Adaptive control using multiple models, switching and tuning," *International Journal of Adaptive Control and Signal Processing*, vol. 14, no. 2, pp. 87-102, 2003.
- [2] R. N. Shorten and K. S. Narendra, "On the stability and existence of common Lyapunov functions for linear stable switching systems," in *Proceedings 37th Conference on Decision and Control*, 1998.
- [3] D. T. Magill, "Optimal adaptive estimations of sampled stochastic processes," *IEEE Transactions on Automatic Control*, vol. 10, no. 4, pp. 434-439, 1965.
- [4] D. G. Lainiotis, "Partitioning: a unifying framework for adaptive systems, I: estimation," in *Proceedings of the IEEE*, vol. 64, pp. 1126-1143, 1976.
- [5] M. Athans, D. Castanon, K. P. Dunn, C. S. Greene, W. H. Lee, N. R. Sandell, and A. S. Willsky, "The stochastic control of the F-8C aircraft using the multiple model adaptive control method, part 1: equilibrium flight," *IEEE Transactions on Automatic Control*, vol. 22, no. 5, pp. 768-791, 1977.

- [6] R. L. Moose, H. F. Van Landingham, and D. H. McCabe, "Modeling and estimation for tracking maneuvering targets," *IEEE Transactions on Aerospace Electrical Systems*, vol. 15, no. 2, pp. 448-456, 1979.
- [7] R. J. Roy, H. Kaufman, and B. W. Bequette, "Multiple-model adaptive predictive control of mean arterial pressure and cardiac output," *IEEE Transactions on Biomedical Engineering*, vol. 39, no. 8, pp. 765-778, 1992.
- [8] X. R. Li and Y. Bar-Shalom, "Design of an interacting multiple model algorithm for air traffic control tracking," *IEEE Transactions on Control Systems Technology*, vol. 1, no. 3, pp. 186-194, 1993.
- [9] D. W. Lane and P. S. Maybeck, "Multiple model adaptive estimation applied to the LAMBDA URV for failure detection and identification," in *IEEE Conference on Decision and Control*, Dec. 1994.
- [10] B. Martensson, "Adaptive stabilization," PhD Dissertation, Lund Institute of Technology, Lund, Sweden, 1986.
- [11] K. S. Narendra and C. Xiang, "Adaptive control of discrete-time systems using multiple models," *IEEE Transactions on Automatic Control*, vol. 45, no. 9, pp. 1669-1686, 2000.
- [12] K. S. Narendra and O. A. Driollet, "Stochastic adaptive control using multiple models for improved performance in the presence of random disturbances," *International Journal of Adaptive Control and Signal Processing*, vol. 15, no. 3, pp. 287-317, 2001.

- [13] L. Chen and K. S. Narendra, "Intelligent control using multiple neural networks," *International Journal of Adaptive Control and Signal Processing*, vol. 17, pp. 417-430, 2003.
- [14] K. S. Narendra and J. Balakrishnan, "Improving transient response of adaptive control systems using multiple models and switching," *IEEE Transactions on Automatic Control*, vol. 39, no. 9, pp. 1861-1866, 1994.
- [15] K. S. Narendra and J. Balakrishnan, "Adaptive control using multiple models," *IEEE Transactions on Automatic Control*, vol. 42, no. 2, pp. 171-187, 1997.
- [16] C. Xiang and K. S. Narendra, "Design issues in stochastic adaptive control of discrete-time systems using multiple models," in *IEEE Conference on Decision and Control*, Dec. 2002.
- [17] K. S. Narendra and O. A. Driollet, "Stochastic adaptive control using multiple estimation models," in *Proceedings of the American Control Conference*, Jun. 2001.
- [18] O. L. V. Costa and M. D. Fragoso, "Stability results for discrete-time linear systems with markovian jumping parameters," *International Journal of Adaptive Control and Signal Processing*, vol. 6, pp. 557-579, 1993.
- [19] J. P. Hespanha, D. Liberzon, A. S. Morse, B. D. O. Anderson, T. S. Brinsmead, and F. (De) Bruyne, "Multiple Multiple model adaptive control. Part 2: switching," *International Journal of Adaptive Control and Signal Processing*, vol. 11, no. 5, pp. 479-496, 2001.

- [20] T. Autenrieth and E. Rogers, "Performance enhancements for a class of multiple model adaptive control schemes," *International Journal of Adaptive Control and Signal Processing*, vol. 13, no. 2, pp. 105-127, 2003.
- [21] J. P. Hespanha, "Logic-based switching algorithms in control," PhD Thesis, Department of Electrical Engineering, Yale University, 1998.
- [22] J. P. Hespanha, D. Liberzon, and A. S. Morse "Hysteresis-based switching algorithms for supervisory control of uncertain systems," *Automatica*, vol. 39, pp. 263272, 2003.
- [23] F. N. Chowdury, J.N. Christensen, and J.L. Aravena, "Power system fault detection and state estimation using Kalman filter with hypothesis testing," *IEEE Transactions on Power Delivery*, vol. 6, no. 3, pp. 1025-1030, 1991.
- [24] F. N. Chowdury, "Kalman filter with hypothesis testing: a tool for estimating uncertain parameters," *Circuits, Systems, and Signal Processing*, vol. 15, no. 3, pp. 291-311, 1996.
- [25] G. C. Goodwin and K. Sin, *Adaptive Filtering, Prediction and Control*, New Jersey: Prentice-Hall, 1984.
- [26] J. P. Hespanha, D. Liberzon, A. S. Morse, B. D. O. Anderson, T. S. Brinsmead, and F. (De) Bruyne, "A weighted adaptive one-step-ahead minimum variance controller based on the ELS algorithm," *International Journal of Adaptive Control and Signal Processing*, vol. 11, no. 6, pp. 461-474, 1997.

- [27] L. Guo and H. F. Chen “The Astrom-Wittenmark self tuning-regulator revisited and ELS-based adaptive trackers,” *IEEE Transactions on Automatic Control*, vol. 37, no. 7, pp. 802-812, 1991.
- [28] T. L. Lai and C. Z. Wei “Extended least squares and their applications to adaptive control and prediction in linear systems,” *IEEE Transactions on Automatic Control*, vol. 31, no. 10, pp. 898-906, 1986.
- [29] J. K. Park, C. H. Chio, and H. Choo, “Dynamic anti-windup method for a class of time-delay control systems with input saturation,” *International Journal of Robust and Nonlinear Control*, vol. 14, pp. 665-682, 2004
- [30] S. Tarbouriech, J. M. Gomes Da Silva Jr., and G. Garcia “Delay-dependent anti-windup strategy for linear systems with saturating inputs and delayed outputs,” *International Journal of Robust and Nonlinear Control*, vol. 10, pp. 457-488, 2000
- [31] K. J. Åström and B. Wittenmark, *Adaptive Control*, 2nd ed., New York: Addison-Wesley, 1995.
- [32] J. Gathen and J. Gerhard, *Modern Computer Algebra*, New York: Cambridge University Press, 1999.
- [33] M. Loeve, *Probability Theory*, 2nd ed., New Jersey: Princeton, 1960.
- [34] C. T. Abdallah, P. Dorato, and S. Karni, “Continuous and discrete-time SPR design using feedback,” in *Proceedings of the American Control Conference*, 1991.

Appendix A

Mathematical Methods and Definitions

A.1 Modified MSE for Quantitative Analysis

Switching should be fast and accurate to yield transient control error that is small in both magnitude and duration. Because control error samples, $e_c(k)$, can be of very large magnitude when a parameter jump yields a temporarily unstable system, they are observed data outliers. To quantitatively analyze the control error $e_c(k)$, produced during an m time-step simulation, a modified MSE is calculated as:

$$\text{MSE} = \frac{1}{m} \sum_{i=1}^m x_i^2, \quad \text{where } x_i = \begin{cases} e_c(i) & : e_c(i) \leq T_{e_c} \\ T_{e_c} & : e_c(i) > T_{e_c} \end{cases}$$

This modified MSE calculation is proposed in this thesis. It employs a user-defined threshold, T_{e_c} , to identify, and apply a ceiling to, outliers. The resulting MSE is without bias from extraordinarily large control error samples. This statistic is

more useful for comparing the control error of simulations containing multiple jumps in parameters. Using a larger threshold, T_{e_c} , places more quantitative value on the magnitude of transient error. Using a smaller T_{e_c} places more value on the speed at which the transient error converges.

A.2 Strictly Positive Real

This definition of the strictly positive real (SPR) condition is taken from [34]. Consider the discrete-time multi-input multi-output linear time-invariant system,

$$x(k+1) = Ax(k) + Bu(k),$$

$$y(k) = Cx(k) + Du(k),$$

where $x(k)$ is an n vector, $u(k)$ is an m vector, $y(k)$ is a p vector, and A , B , C , and D are of the appropriate dimensions. The corresponding transfer function matrix is

$$P(z) = C(zI - A)^{-1} + Du.$$

It is assumed that the system has an equal number of inputs and outputs (*i.e.* $p = m$) and define the relative degree n^* as follows: $n^* = 0$ if $D \neq 0$ but $CB = 0$. The Hermitian part of $T(z)$ is denoted by

$$He[T(z)] = \frac{1}{2}[T(z) + T^T(z^*)],$$

where z^* is the complex conjugate of z and $z = \rho^{jw} = 0$.

An $m \times m$ matrix $T(z)$ of real rational functions is SPR if :

1. All elements of $T(z)$ are analytic on and outside the unit circle (*i.e.* in the region $z \geq 1$), and
2. The matrix $He[T(\rho^{jw})](z)$ is positive definite Hermitian for all real w .

Note that a necessary condition for $T(z)$ to be SPR is that it is relative degree $n^* = 0$.

A.3 Persistency of Excitation

For convergence of the all parameter estimates in a given system, the estimation process must be persistently excited. To parameterize a 2nd order SISO system, four parameters (two numerator and two denominator coefficients) must be found. A sinusoidal signal is of order 2. By exciting the system with a single sinusoid, there is no way to determine all four parameters. This input would not be persistently exciting. A signal is persistently exciting if it contains sufficient frequencies to enable it to continuously update all parameter estimates. In other words, all the “modes” of the system must be excited.

In this thesis, two unique sinusoidal signals are used together to excite the estimation process for a 2nd order plant. Because the ELS algorithm is used, there are additional parameters to be estimated. These are excited also because white noise (order n) is injected into the system).

A.4 Monotonically Increasing

A sequence $\{f(y)\}$, $f(y) \in R$ is called a monotonically increasing sequence if

$$x > y \text{ implies } f(x) \geq f(y).$$

A.5 Lemma 3 - Necessary for Proving Theorem 2

Let $\{a_k\}$ be a sequence of real numbers such that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N a_k < \infty,$$

then

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N \frac{a_k}{k^2} < \infty.$$

Proof of Lemma 3:

Let

$$A_N \sum_{k=1}^N a_k,$$

and set $A_0 = 0$, then:

$$\begin{aligned} \sum_{k=1}^N \frac{a_k}{k^2} &= \sum_{k=1}^N \frac{A_k}{k^2} - \frac{A_{k-1}}{k^2} \\ &= \frac{A_N}{N^2} + \sum_{k=1}^{N-1} \left(\frac{A_k}{k^2} - \frac{A_{k-1}}{(k+1)^2} \right) \\ &= \frac{A_N}{N^2} + \sum_{k=1}^{N-1} \frac{A_k(2k+1)}{k^2(k+1)^2} \\ &= \frac{A_N}{N^2} + \sum_{k=1}^{N-1} \frac{A_k}{k} \left(\frac{2}{(k+1)^2} - \frac{1}{k(k+1)^2} \right) \end{aligned}$$

Since A_N/N is bounded, it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{N} \sum_{k=1}^N \frac{a_k}{k^2} < \infty.$$

A.6 Cauchy-Schwarz Inequality

For any two vectors, $x = (x_1, \dots, x_n)$, $y = (y_1, \dots, y_n)$, $\in \mathbb{R}^n$, it follows [32] that

$$\sum_{k=1}^n x_k^2 \sum_{k=1}^n y_k^2 = (\|x\|^2) (\|y\|^2) \geq x^2 * y^2 = \left(\sum_{k=1}^n x_k y_k \right)^2.$$

A.7 Kronecker Lemma

Let

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N x_k = s,$$

and let the term b_N be increasing monotonically such that $b_N \rightarrow \infty$. Then it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N b_k x_k = 0.$$

The ensuing proof of the Kronecker Lemma is adapted from [33].

Proof of Kronecker Lemma:

The following values are assigned:

$$b_0 = 0$$

$$a_k = b_k - b_{k-1}$$

$$s_{N+1} = \sum_{k=1}^N x_k.$$

Then it can be obtained that:

$$\begin{aligned} \lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N b_k x_k &= \lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N b_k (s_{k+1} - s_k), \\ &= \lim_{N \rightarrow \infty} \left(s_{N+1} - \frac{1}{b_N} \sum_{k=1}^N a_k s_k \right), \\ &= s - s, \\ &= 0. \end{aligned} \tag{A.1}$$

A.8 Modified Kronecker Lemma

The term x_k is defined to be a sequence of real numbers such that

$$x_k \geq 0$$

and

$$\lim_{N \rightarrow \infty} \sum_{k=1}^N x_k = s < \infty.$$

The term b_N is defined to be positive and increasing such that $b_N \rightarrow \infty$. Let there be a value $K > 0$ such that

$$\frac{b_l}{b_N} \leq K,$$

for all $l \leq N$. Then it follows that

$$\lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N b_k x_k = 0.$$

Proof of Modified Kronecker Lemma:

The proof of the modified Kronecker Lemma is taken from [12]. The following values are assigned:

$$b_0 = 0,$$

$$a_k = b_k - b_{k-1},$$

$$b_N = \sum_{k=1}^N a_k,$$

$$s_{N+1} = \sum_{k=1}^N x_k.$$

Note that s_N is monotonically increasing since $x_k \geq 0$. It follows that

$$\frac{1}{b_N} \sum_{k=1}^N b_k x_k = s_{N+1} - \frac{1}{b_N} \sum_{k=1}^N a_k s_k.$$

Substituting the term $s_k = s_k - s + s$, the second term in the r.h.s. becomes

$$-\frac{1}{b_N} \sum_{k=1}^N a_k (s_k - s) - \frac{1}{b_N} \sum_{k=1}^N a_k s.$$

It follows that

$$\frac{1}{b_N} \sum_{k=1}^N b_k x_k = (s_{N+1} - s) - \frac{1}{b_N} \sum_{k=1}^N a_k (s_k - s).$$

The term z_k is defined as $x_k = s - s_k$, with $z_k > 0$ and decreasing monotonically such that $z_k \rightarrow 0$. It follows that

$$\lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N b_k x_k = (s - s) + \lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N \frac{a_k}{b_N} z_k.$$

Since $z_k \rightarrow 0$, for any $\epsilon > 0$ there exists $\eta(\epsilon)$ such that $z_k < \epsilon$ for all $N \geq \eta(\epsilon)$. This can be stated as

$$\left| \frac{1}{b_N} \sum_{k=1}^N a_k x_k \right| \leq \frac{1}{b_N} \left| \sum_{k=1}^{\eta(\epsilon)} a_k z_k \right| + \frac{1}{b_N} \left| \sum_{k=\eta(\epsilon)+1}^N a_k z_k \right|.$$

This equation is composed of one l.h.s term and two r.h.s terms (each with a summation). The l.h.s term tends to zero. The first r.h.s. term can be expressed as

$$\frac{1}{b_N} \left| \sum_{k=\eta(\epsilon)+1}^N a_k z_k \right| \leq \frac{1}{b_N} \left| \sum_{k=\eta(\epsilon)+1}^N \frac{b_k - b_{k-1}}{b_N} z_k \right|.$$

Considering that

$$\frac{b_l}{b_N} \leq K,$$

for all $l \leq N$. And considering that z_k decreases monotonically and $|z_k| < \epsilon$, the second term in the r.h.s. can be shown to be less than $C\epsilon$, where C is a constant. Since ϵ is arbitrary,

$$\lim_{N \rightarrow \infty} \frac{1}{b_N} \sum_{k=1}^N b_k x_k = 0,$$

and the lemma is proved.

The Kronecker lemma and modified Kronecker lemma are very similar, except for the fact that b_k is not sign definite in the Kronecker lemma and non negative in the modified Kronecker lemma. Also, b_k is a monotonically increasing positive sequence in the Kronecker lemma and only positive in the modified Kronecker lemma. In both lemmas, $b_N \rightarrow \infty$. The requirement for the modified Kronecker lemma is that for any N , $b_l/b_N \leq K$ for all $l \leq N$, where K is a positive constant.

Appendix B

Simulation Code

B.1 Comparison of RLS and ELS Parameter Estimation Methods

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoRlsVsEls.m (by: Alex Campbell)
% This file performs simulations for Section 2.5.2, titled:
% "Comparison of RLS and ELS Parameter Estimation Methods"
% Uses files:
%       simRlsSystem.m
%       simElsSystem.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List of terms:
%   simL      - number of time-steps to simulate for
%   myNoise    - the disturbance sequence
%   lambda    - estimation forgetting factor
%   p         - plant parameter vector
%   plantState - plant parameter vector sequence during 'simL' time-steps
%   myRef     - the reference signal sequence
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%% Setup initial conditions
close all; clear all; clc;
simL = 1000;
noiseSeed = mod(sum(clock)*1000,100);
randn('seed',noiseSeed); % SAME SEED EVERY TIME
myNoise = 0.5*randn(1,simL); %Generate Normal Gaussian Noise
lambda = 1;
p = [ 0.6, 0.2, 1.0, 0.5, -0.9, 0.9]; % Define Plant parameters
```



```

for k = 1:simL
    myRef(k) = sin(4*pi*k/50) + sin(4*pi*k/150);
    plantState(:,k) = p;
end
save initialCond %values saved as initialCond.mat

%%%%% First Simulate Control using Recursive Least Squares (RLS)
simRlsSystem;
save saveRLSsim %results saved as saveRLSsim.mat for plotting later

%%%%% NOW Simulate Control using Extended Least Squares (ELS)
clear all;
load initialCond
simElsSystem;
save saveELSSim %results saved as saveELSSim.mat for plotting later

% end of file: demoRlsVsEls.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: simRlsSystem.m (by: Alex Campbell)
% This file performs RLS estimation and control of an unknown plant.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This file is used by a parent file which has defined the following:
%
% double:                simL - number of time-steps to simulate for.
% double:                lambda - forgetting factor.
% <1xsimL> double:        myRef - the reference signal.
% <1xsimL> double:        myNoise - the disturbance present.
% <6xsimL> double:        plantState - the plant parameter vector of 6 parameters
%                               over 'simL' time-steps.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List of terms:
% iP - initial values in trace of 'P'
% P - covariance matrix
% Q - parameter estimate vector
% O - regression vector
% u - most recent plant input
% ud - previous 'u' value (delay 1)
% Y - most recent plant output
% Yd - previous 'Y' value (delay 1)
% wd - most recent noise sample
% wd - previous 'w' value (delay 1)
% wdd - previous 'wd' value (delay 1)
% Yr - most recent reference signal value
% Yrd - previous 'Yr' value
% k - most recent time-step
% Yest - the adaptive model's estimated plant output 'Y'
% er - prediction error
% Ec - control error
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% Setup initial conditions
iP = 1000000;
P = iP*eye(4);
Q = [0 0 1 0]';

```

```

ud = 0; u = 0;
Yd = 0; Y = 0;
Yrd = 0; Yr = 0;
wdd = 0; wd = 0; w = 0;

% Begin simulation:
for k = 1:simL
    udd = ud; ud = u;
    wdd = wd; wd = w;
    w = myNoise(k);
    Yrd = Yr;
    Yr = myRef(k);
    p = plantState(:,k);
    Ydd = Yd; Yd = Y;
    Y = p(1)*Yd + p(2)*Ydd + p(3)*ud + p(4)*udd + p(5)*wd + p(6)*wdd + w;

% Perform RLS parameter estimation
    Q = [Yd; Ydd; ud; udd];
    K = P*Q*inv(lambda*eye(1) + Q'*P*Q);
    P = (eye(4) - K*Q')*P/lambda;
    Yest = Q'*Q;
    er = Y - Yest;
    Ec = Y - Yrd;
    Q = Q + K*er; %Parameter vector updated

% Generate certainty equivalence control signal
    if Q(3)<0.01 % Enforce a b_min, as discussed in text
        Q(3) = 0.01;
    end
    Qc = [Q(1) Q(2) Q(3) Q(4)]';
    Qc = [Y Yd 0 ud];
    u = (Yr-Qc*Qc)/Qc(3);

% Save results of time-step k
    saveY(k) = Y;
    saveYr(k) = Yrd;
    saveU(k) = u;
    saveW(k) = w;
    saveE(k) = Ec;
    saveQ(:,k) = Q;
    saveP(:,k) = p';
end

% end of file: simRlsSystem.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: simElsSystem.m (by: Alex Campbell)
% This file performs ELS estimation and control of an unknown plant.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This file is used by a parent file which has defined the following:
%
% double:          simL - number of time-steps to simulate for.
% double:          lambda - forgetting factor.
% <1xsimL> double:  myRef - the reference signal.
% <1xsimL> double:  myNoise - the disturbance present.
% <6xsimL> double: plantState - the plant parameter vector of 6 parameters
%                  over 'simL' time-steps.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List of terms:
%   iP - initial values in trace of 'P'

```

```

% P - covariance matrix
% Q - parameter estimate vector
% O - regression vector
% u - most recent plant input
% ud - previous 'u' value (delay 1)
% Y - most recent plant output
% Yd - previous 'Y' value (delay 1)
% wd - most recent noise sample
% wd - previous 'w' value (delay 1)
% wdd - previous 'wd' value (delay 1)
% Yr - most recent reference signal value
% Yrd - previous 'Yr' value
% k - most recent time-step
% Yest- the adaptive model's estimated plant output 'Y'
% er - prediction error
% Ec - control error
% edr - previous 'er' value (delay 1)
% eddr - previous 'edr' value (delay 1)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Setup initial conditions
iP = 1000000; % Initial Trace of P
P = iP*eye(6)
Q = [0 0 1 0 0 0]';
ud = 0; u = 0;
Yd = 0; Y = 0;
Yrd = 0; Yr = 0;
wdd = 0; wd = 0; w = 0;
edr = 0; er = 0;
% Begin simulation:
for k = 1:simL
    udd = ud; ud = u;
    wdd = wd; wd = w;
    w = myNoise(k);
    Yrd = Yr;
    Yr = myRef(k);
    p = plantState(:,k);
    Ydd = Yd; Yd = Y;
    Y = p(1)*Yd + p(2)*Ydd + p(3)*ud + p(4)*udd + p(5)*wd + p(6)*wdd + w;
% Perform ELS parameter estimation
    eddr = edr; edr = er; % Past Prediction Errors
    O = [Yd; Ydd; ud; udd; edr; eddr];
    K = P*O*inv(lambda*eye(1) + O'*P*O);
    P = ( eye(6) - K*O' ) * P / lambda ;
    Yest = O'*Q;
    er = Y - Yest;
    Ec = Y - Yrd;
    Q = Q + K*er; %Parameter vector updated
% Generate certainty equivalence control signal
    if Q(3)<0.01 % Enforce a b_min, as discussed in text
        Q(3) = 0.01;
    end
    Qc = [Q(1) Q(2) Q(3) Q(4) Q(5) Q(6)]';
    Oc = [Y Yd 0 ud er edr];
    u = (Yr-Oc*Qc)/Qc(3);
% Save results of time-step k

```

```

    saveY(k) = Y;
    saveYr(k) = Yrd;
    saveU(k) = u;
    saveW(k) = w;
    saveE(k) = Ec;
    saveQ(:,k) = Q;
    saveP(:,k) = p';
    saveTrace(k) = trace(P);
end
% end of file: simElsSystem.m

```

B.2 Selection of Forgetting Factor, λ , for ELS in Time-Varying Systems

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoSelectLambda.m (by: Alex Campbell)
% This file performs simulations for Section 2.5.3, titled:
% "Selection of Forgetting Factor for ELS in Time-Varying Systems"
% Uses file:
%       simElsSystem.m (defined with code for Section 2.5.2)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List of terms:
%   simL      - number of time-steps to simulate for
%   myNoise    - the disturbance sequence
%   lambda     - estimation forgetting factor
%   p1         - initial plant parameter vector
%   p2         - plant parameter vector after jump
%   plantState - plant parameter vector sequence during 'simL' time-steps
%   myRef      - the reference signal sequence
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Setup initial conditions
close all; clear all; clc;
simL = 400;
noiseSeed = mod(sum(clock)*1000,100);
randn('seed',noiseSeed); % reproducible normal rand numbers
myNoise = 0.3*randn(1,simL); %Generate Normal Gaussian Noise
lambda = .97;
p1 = [ 0.6, 0.2, 1.0, 0.5, -0.5, 0.4]; % Define Plant parameters
p2 = [ -0.5, 0.2, 1.0, 0.2, -0.5, 0.4]; % Define Plant parameters
for k = 1:simL
    myRef(k) = sin(4*pi*k/50) + sin(4*pi*k/150);
    plantState(:,k) = p1;
    if k > 200
        plantState(:,k) = p2;
    end
end
save initialCond %values saved as initialCond.mat
%%%% Now simulate control using ELS
clear all; load initialCond;
lambda = .99;

```

```

simElsSystem; save saveELSSim1
clear all; load initialCond;
lambda = .98;
simElsSystem; save saveELSSim2
clear all; load initialCond;
lambda = .97;
simElsSystem; save saveELSSim3
clear all; load initialCond;
lambda = .95;
simElsSystem; save saveELSSim4
clear all; load initialCond;
lambda = .90;
simElsSystem; save saveELSSim5
% end of file: demoSelectLambda.m

```

B.3 Demonstration of Persistency of Excitation Requirements for ELS

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoPersistency.m (by: Alex Campbell)
% This file performs simulations for Section 2.5.4, titled:
% "Demonstration of Persistency of Excitation Requirements for ELS"
% Uses file:
%       simElsSystem.m (defined with code for Section 2.5.2)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List of terms:
%   simL       - number of time-steps to simulate for
%   myNoise     - the disturbance sequence
%   lambda     - estimation forgetting factor
%   p1         - initial plant parameter vector
%   p2         - plant parameter vector after jump
%   plantState - plant parameter vector sequence during 'simL' time-steps
%   myRef      - the reference signal sequence
%   YrList     - the exciting reference signal
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Setup initial conditions
close all; clear all; clc;
simL = 2000;
noiseSeed = mod(sum(clock)*1000,100);
randn('seed',noiseSeed); % reproducible normal rand numbers
myNoise = 0.1*randn(1,simL); %Generate Normal Gaussian Noise
lambda = .97;
p1 = [ 0.6, 0.2, 1.0, 0.5, -0.5, 0.4]; % Define Plant parameters
p2 = [ -0.5, 0.2, 1.0, 0.2, -0.5, 0.4]; % Define Plant parameters
for k = 1:simL
    myRef(k) = sin(4*pi*k/50) + sin(4*pi*k/150);
    if (k > 500) && (k < 1800) %change these values observe
        myRef(k) = 0; % this makes the reference stop moving
        myNoise(k) = 0; % this makes zero noise
    end
end

```

```

    plantState(:,k) = p1;
    if k > 9999 % change this values to cause a parameter jump
        plantState(:,k) = p2;
    end
end
%%%% Now simulate control using ELS
load initialCond;
simElsSystem;
% end of file: demoPersistency.m

```

B.4 Demonstration of Convergence with Random Switching Between Two Adaptive Models

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoTwoModel.m (by: Alex Campbell)
% This file performs the simulation for Section 3.6.2, titled:
% "Demonstration of Convergence with Random Switching Between
% Two Adaptive Models"
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List terms not defined in previous lists:
%   Qr - parameter estimate vector for resetting adaptive model
%   Qf  parameter estimate vector for free-running adaptive model
%   lambdaReset - forgetting factor for resetting adaptive model
%   lambdaFree  - forgetting factor for free-running adaptive model
%   Pr - covariance matrix for resetting adaptive model
%   Pf - covariance matrix for free-running adaptive model
%   Ymr - estimated plant output for resetting model
%   Ymf - estimated plant output for free-running model
%   er  - estimate error for resetting model
%   ef  - estimate error for free-running model
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%% Setup initial conditions
close all; clear all; clc;
simL = 150;
noiseSeed = mod(sum(clock)*1000,100);
randn('seed',noiseSeed); % reproducible normal rand numbers
myNoise = 0.1*randn(1,simL); %Generate Normal Gaussian Noise
p1 = [ 0.6, 0.2, 1.0, 0.5, -0.5, 0.4]; % Define Plant parameters
zpk(tf([p1(3), p1(4)], [1 -p1(1), -p1(2)])) %display transfer function 'p1'
for k = 1:simL
    myRef(k) = sin(2*pi*k/20) + sin(2*pi*k/60);
    plantParameters(:,k) = p1;
end
lambdaReset = .97;
lambdaFree = 1;
iP = 1000000;
Pr = iP*eye(6); %
Pf = iP*eye(6); %
ModSel = 1; %Initially Model 1 is selected

```

```

%Random initial parameter estimates:
Qr = [(2*rand-1) (2*rand-1) 1 (2*rand-1) (2*rand-1) (2*rand-1)]';
Qf = [(2*rand-1) (2*rand-1) 1 (2*rand-1) (2*rand-1) (2*rand-1)]';
Ymr = 0; Ymf = 0;
Yrdd = 0; Yrd = 0; Yr = 0;
Ydd = 0; Yd = 0; Y = 0;
wdd = 0; wd = 0; w = 0;
udd = 0; ud = 0; u = 0;
Ecdd = 0; Ecd = 0; Ec = 0;
edr = 0; er = 0;
edf = 0; ef = 0;
Yr = 0;
Ec = 0;
%%%% Now Perform Simulation
for k = 1:simL
    udd = ud;
    ud = u;
    p = plantParameters(:,k); %gets current plant parameters
    wdd = wd;
    wd = w;
    w = myNoise(k);
%---Update plant:
    Ydd = Yd;
    Yd = Y;
    Y = p(1)*Yd + p(2)*Ydd + p(3)*ud + p(4)*udd + p(5)*wd + p(6)*wdd + w;
    Yrd = Yr; %ref
    Yr = myRef(k);
    Ecdd = Ecd; Ecd = Ec;
    Ec = abs(Yrd - Y);
    Ecs = Yrd - Y;
    Ymrd = Ymr;
    Ymfd = Ymf;
%---update resetting model's estimates
    eddr = edr; edr = er;
    O = [Yd; Ydd; ud; udd; edr; eddr];
    K = Pr*O*inv(lambdaReset*eye(1) + O'*Pr*O);
    Pr = ( eye(6) - K*O' )*Pr/lambdaReset ;
    Yest = O'*Qr;
    er = Y - Yest;
    Qr = Qr + K*er;
    if Qr(3)<0.01 % Enforce a b_min, as discussed in text
        Qr(3) = 0.01;
    end
    %Output estimate of resetting model:
    Ymr = Qr(1)*Yd + Qr(2)*Ydd + Qr(3)*ud + Qr(4)*udd;
%---update free-running model's estimates
    eddf = edr; edf = ef;
    O = [Yd; Ydd; ud; udd; edf; eddf];
    K = Pf*O*inv(lambdaFree*eye(1) + O'*Pf*O);
    Pf = ( eye(6) - K*O' )*Pf/lambdaFree ;
    Yest = O'*Qf;
    ef = Y - Yest;
    Qf = Qf + K*ef;
    if Qf(3)<0.01 % Enforce a b_min, as discussed in text
        Qf(3) = 0.01;
    end
    %Output estimate of free-running model:
    Ymf = Qf(1)*Yd + Qf(2)*Ydd + Qf(3)*ud + Qf(4)*udd;

```

```

%---Randomly switch models
ModSel = ceil(2*rand)
saveModSel(k) = ModSel;
if ModSel == 1, Q = Qr; end
if ModSel == 2, Q = Qf; end
%---Generate certainty equivalence control signal
Qu = [Q(1) Q(2) Q(3) Q(4)]';
Ou = [Y Yd 0 ud];
u = (Yr-Ou*Qu)/Qu(3);
%---Save results for time-step k
saveY(k) = Y;
saveYr(k) = Yrd;
saveU(k) = u;
saveW(k) = w;
saveE(k) = Ecs;
saveQ(:,k) = Q;
end % END OF LOOP
% end of file: demoTwoModel.m

```

B.5 Comparison of Multiple and Single Model Control Performance

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoMMvsSM.m (by: Alex Campbell)
% This file performs the simulations for Section 3.6.3, titled:
% "Comparison of Multiple and Single Model Control Performance"
% Uses file:
%   simClassicalMMAC.m (this file provided with code for Section 3.6.4)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List terms not defined in previous lists:
%   myNoise      - the disturbance sequence
%   lambda       - estimation forgetting factor
%   p            - plant parameter vector
%   plantState   - plant parameter vector sequence during 'simL' time-steps
%   myRef        - the reference signal sequence
%   modelPlacementSeed - seed used for generating random model
%                   parameterization error
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% Setup initial conditions
close all; clear all; clc;
simL = 1000;
noiseSeed = mod(sum(clock)*1000,100);
randn('seed',noiseSeed); % reproducible normal rand numbers
myNoise = 0.1*randn(1,simL); %generate normal gaussian noise
p1 = [ 0.6, 0.2, 1.0, 0.5, -0.5, 0.4]; % Define Plant parameters
p2 = [-.5, 0.2, 4.0, -0.2, -0.5, 0.4]; % Define Plant parameters
for k = 1:simL
    myRef(k) = sin(2*pi*k/20) + sin(2*pi*k/60);
    plantState(k) = 1;
    plantParameters(:,k) = p1;

```



```

    if k > 500
        plantState(k) = 2;
        plantParameters(:,k) = p2;
    end
end
save initialCond %values saved as initialCond.mat
%%%%% Now Simulate Control using single model ELS
clear all;
load initialCond;
lambda = .97;
simSMAC;
save saveSimSMAC
clear all;
load initialCond;
%%%%% Initialize additional values for the multiple model control sim.
lambdaReset = .97;
lambdaFree = .99;
d3 = .9;
numbFixedModels = 2; % makes fixed model 1 & 2 enabled for use
                    % affects model selection in
                    % "selectModel_Index.m"
modelPlacementSeed = mod(sum(clock)*1000,100);
rand('seed',modelPlacementSeed); % reproducible normal rand numbers
model1 = p1 +
0.15*[(2*rand-1) (2*rand-1) (2*rand-1) (2*rand-1) (2*rand-1) (2*rand-1)];
model2 = p2 +
0.15*[(2*rand-1) (2*rand-1) (2*rand-1) (2*rand-1) (2*rand-1) (2*rand-1)];
model3 = [ 0 0 1 0 0 0]; % not used
%%%%% Now Simulate Control using single model ELS
simClassicalMMAC;
save saveSimClassicalMMAC
% end of file: demoMMvsSM.m

```

B.6 Comparing Performance of the Classical and Modified Model Arrangement with Performance Index B Switching

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoCompareArrangements.m (by: Alex Campbell)
% This file performs the simulations for Section 3.6.4, titled:
% "Comparing Performance of the Classical and Modified Model Arrangement
% with Performance Index B Switching"
% Uses file:
%       simClassicalMMAC.m
%       simModifiedMMAC.m
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%% Setup initial conditions
close all; clear all; clc;
simL = 900;

```

```

noiseSeed = mod(sum(clock)*1000,100);
randn('seed',noiseSeed); % reproducible normal rand numbers
myNoise = 0.2*randn(1,simL); %Generate Normal Gaussian Noise
p1 = [ 0.6, 0.2, 1.0, 0.5, -0.5, 0.4]; % Define Plant parameters
p2 = [ -0.5, 0.2, 1.0, 0.2, -0.5, 0.4]; % Define Plant parameters
p3 = [ 1.7, 0.7, 1.0, -0.5, 0.5, -0.4]; % Define Plant parameters
for k = 1:simL
    myRef(k) = sin(2*pi*k/20) + sin(2*pi*k/60);
    plantState(k) = 1;
    plantParameters(:,k) = p1;
    if k > 300
        plantState(k) = 2;
        plantParameters(:,k) = p2;
    if k > 600
        plantState(k) = 3;
        plantParameters(:,k) = p3;
    end
end
end
lambdaReset = .97;
lambdaFree = .99;
numbFixedModels = 3; % makes all 3 fixed models enabled
                    % affects "selectModel_Index.m"
modelPlacementSeed = mod(sum(clock)*1000,100);
rand('seed',modelPlacementSeed);
model1 = p1 + 0.2*[-rand -rand -rand rand -rand rand];
model2 = p2 + 0.2*[-rand -rand -rand rand -rand rand];
model3 = p3 + 0.2*[-rand -rand -rand rand -rand rand];
save initialCond %values saved as initialCond.mat
%%%% Now Perform Tuning by simulating 100 times.
clear all;
for tuneAttempt = 1:100
    load initialCond;
    d3 = 0.0 + tuneAttempt/100;
    saveD3(tuneAttempt) = d3;
    simClassicalMMAC;
    saveClassicalMSE(tuneAttempt) = mse(saveAbContE);
    simModifiedMMAC;
    saveModifiedMSE(tuneAttempt) = mse(saveAbContE);
end
save saveTuningProcess;
%%%% Find the lowest MSE obtained using classical model arrangement
load initialCond;
[value index] = min(saveClassicalMSE)
d3 = saveD3(index)
simClassicalMMAC;
save saveSimClassicalMMAC
%%%% Find the lowest MSE obtained using modified model arrangement
load initialCond;
[value index] = min(saveModifiedMSE)
d3 = saveD3(index)
simModifiedMMAC;
save saveSimModifiedMMAC
% end of file: demoCompareArrangements.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: simClassicalMMAC.m (by: Alex Campbell)
% This file is used by parent classes throughout Chapter 3. It performs

```

```

% multiple model adaptive control using the CLASSICAL MODEL ARRANGEMENT.
% It uses the following files mutually exclusively:
%   selectModel_Index.m (performance index switching)
%   selectModel_Htest.m (switching using hypothesis testing)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This file is used by a parent files which have defined the following:
%
% double:          simL - number of time-steps to simulate for.
% <1xsimL> double:  myRef - the reference signal.
% <1xsimL> double:  myNoise - the disturbance present.
% <6xsimL> double:  plantState - the plant parameter vector of 6 parameters
%                  over 'simL' time-steps.
% double:          lambdaReset - forgetting factor for resetting model
% double:          lambdaFree - forgetting factor for free-running model
% double:          numbFixedModels - number of fixed models used
% <1x6> double:     model1 - fixed model 1 parameters
% <1x6> double:     model2 - fixed model 2 parameters
% <1x6> double:     model3 - fixed model 3 parameters
% double:          d3 - performance index B weight 'forgetting factor'
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List terms not defined in previous lists:
%   Q1, Q2, Q3 - parameter vector of fixed model 1, 2, 3, respectively
%   Ym1,Ym2,Ym3 - model output of fixed model 1, 2, 3, respectively
%   ModSel - the model selection signal (denoted 'i' in text)
%   d1 - performance index B weight
%   d2 - performance index B weight
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Setup initial conditions
iP = 1000000;
Pr = iP*eye(6);
Pf = iP*eye(6);
ModSel = 4; %Initially Model 4m the resetting model, is selected
Q1 = model1';
Q2 = model2';
Q3 = model3';
Qr = model1'; %Resetting model initializes from Model 1
Qf = [0 0 1 0 0 0]';
Ym1 = 0; Ym2 = 0; Ym3 = 0; Ymr = 0; Ymf = 0;
Yrdd = 0; Yrd = 0; Yr = 0;
Ydd = 0; Yd = 0; Y = 0;
wdd = 0; wd = 0; w = 0;
udd = 0; ud = 0; u = 0;
Ecdd = 0; Ecd = 0; Ec = 0;
edr = 0; er = 0;
edf = 0; ef = 0;
sum1 = 0; sum2 = 0; sum3 = 0; sum4 = 0; sum5 = 0;
Yr = 0;
Ec = 0;
d1 = 1;
d2 = 2;
%%% Now Perform Simulation
for k = 1:simL
    udd = ud;
    ud = u;
    p = plantParameters(:,k); %gets current plant parameters
    wdd = wd;

```

```

    wd = w;
    w = myNoise(k);
%---Update plant:
    Ydd = Yd;
    Yd = Y;
    Y = p(1)*Yd + p(2)*Ydd + p(3)*ud + p(4)*udd + p(5)*wd + p(6)*wdd + w;
    Yrd = Yr;
    Yr = myRef(k);
    Ec = Yrd - Y;
%---update fixed model's outputs
    Ym1d = Ym1; Ym2d = Ym2; Ym3d = Ym3; Ymrd = Ymr; Ymfd = Ymf;
    % these past values of plant outputs are used if hypothesis
    % testing is used
    Ym1 = Q1(1)*Yd + Q1(2)*Ydd + Q1(3)*ud + Q1(4)*udd;
    Ym2 = Q2(1)*Yd + Q2(2)*Ydd + Q2(3)*ud + Q2(4)*udd;
    Ym3 = Q3(1)*Yd + Q3(2)*Ydd + Q3(3)*ud + Q3(4)*udd;
%---update resetting model's estimates and outputs
    eddr = edr; edr = er;
    O = [Yd; Ydd; ud; udd; edr; eddr];
    K = Pr*O*inv(lambdaReset*eye(1) + O'*Pr*O);
    Pr = ( eye(6) - K*O' ) * Pr / lambdaReset ;
    Yest = O'*Qr;
    er = Y - Yest;
    Qr = Qr + K*er;
    if Qr(3)<0.01 % Enforce a b_min, as discussed in text
        Qr(3) = 0.01;
    end
    Ymr = Qr(1)*Yd + Qr(2)*Ydd + Qr(3)*ud + Qr(4)*udd;
%---update free-running model's estimates and outputs
    eddf = edr; edf = ef;
    O = [Yd; Ydd; ud; udd; edf; eddf];
    K = Pf*O*inv(lambdaFree*eye(1) + O'*Pf*O);
    Pf = ( eye(6) - K*O' ) * Pf / lambdaFree ;
    Yest = O'*Qf;
    ef = Y - Yest;
    Qf = Qf + K*ef;
    if Qf(3)<0.01 % Enforce a b_min, as discussed in text
        Qf(3) = 0.01;
    end
    Ymf = Qf(1)*Yd + Qf(2)*Ydd + Qf(3)*ud + Qf(4)*udd;
%---Switchg models according to either:
    % "selectModel_Index.m" (performance index switching)
    % "selectModel_Htest.m" (hypothesis test switching)
    oldModSel = ModSel;
    selectModel_Index; %removed: selectModel_Htest;
    saveModSel(k) = ModSel;
%---Using the selected model ModeSel decide how to perform control
%---This method below is the classical model arrangement method
    if ModSel ~= oldModSel
        if ModSel == 1
            Qr = Q1; Pr = iP*eye(6);
            edr = 0; er = 0;
        end
        if ModSel == 2
            Qr = Q2; Pr = iP*eye(6);
            edr = 0; er = 0;
        end
        if ModSel == 3

```

```

        Qr = Q3; Pr = iP*eye(6);
        edr = 0; er = 0;
    end
end
% Select theta (denoted Q) for performing certainty equivalence control
if ModSel == 1, Q = Q1; end
if ModSel == 2, Q = Q2; end
if ModSel == 3, Q = Q3; end
if ModSel == 4, Q = Qr; end
if ModSel == 5, Q = Qf; end
%---Generate certainty equivalence control signal
Qu = [Q(1) Q(2) Q(3) Q(4)]';
Ou = [Y Yd 0 ud];
u = (Yr-Ou*Qu)/Qu(3);
%---Save results for time-step k
saveY(k) = Y;
saveYr(k) = Yrd;
saveU(k) = u;
saveW(k) = w;
saveE(k) = Ec;
saveQ(:,k) = Q;
if Ec^2 >= 4, saveAbContE(k) = 4; end %
if Ec^2 < 4, saveAbContE(k) = Ec^2; end
end % END OF LOOP
% end of file: simClassicalMMAC.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: simModifiedMMAC.m (by: Alex Campbell)
% This file is used by parent classes throughout Chapter 3. It performs
% multiple model adaptive control using the MODIFIED MODEL ARRANGEMENT.
% The modified model arrangement is an adjustment to the classical model
% arrangement. As a result this file is IDENTICAL to 'simClassical.m,'
% except for how controllers are applied. The modifications to the
% classical model arrangement are explained in Chapter 3, Section 3.3. The
% modifications are highlighted below by using the asterix '*' on the same
% line.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List terms not defined in previous lists:
%   ContSel - This is the controller to be applied (1,2,3,4, or 5). It
%               is unique from ModSel (the selected model) because this is
%               the modified model arrangement.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Setup initial conditions
iP = 1000000;
Pr = iP*eye(6);
Pf = iP*eye(6);
ModSel = 4; %Initially Model 4m the resetting model, is selected
Q1 = model1';
Q2 = model2';
Q3 = model3';
Qr = model1'; %Resetting model initializes from Model 1
Qf = [0 0 1 0 0 0]';
Ym1 = 0; Ym2 = 0; Ym3 = 0; Ymr = 0; Ymf = 0;
Yrdd = 0; Yrd = 0; Yr = 0;
Ydd = 0; Yd = 0; Y = 0;
wdd = 0; wd = 0; w = 0;
udd = 0; ud = 0; u = 0;

```

```

Ecdd = 0; Ecd = 0; Ec = 0;
edr = 0; er = 0;
edf = 0; ef = 0;
sum1 = 0; sum2 = 0; sum3 = 0; sum4 = 0; sum5 = 0;
Yr = 0;
Ec = 0;
d1 = 1;
d2 = 2;
%%% Now Perform Simulation
for k = 1:simL
    udd = ud;
    ud = u;
    p = plantParameters(:,k); %gets current plant parameters
    wdd = wd;
    wd = w;
    w = myNoise(k);
%---Update plant:
    Ydd = Yd;
    Yd = Y;
    Y = p(1)*Yd + p(2)*Ydd + p(3)*ud + p(4)*udd + p(5)*wd + p(6)*wdd + w;
    Yrd = Yr;
    Yr = myRef(k);
    Ec = Yrd - Y;
%---update fixed model's outputs
    Ym1d = Ym1; Ym2d = Ym2; Ym3d = Ym3; Ymrd = Ymr; Ymfd = Ymf;
    % these past values of plant outputs are used if hypothesis
    % testing is used
    Ym1 = Q1(1)*Yd + Q1(2)*Ydd + Q1(3)*ud + Q1(4)*udd;
    Ym2 = Q2(1)*Yd + Q2(2)*Ydd + Q2(3)*ud + Q2(4)*udd;
    Ym3 = Q3(1)*Yd + Q3(2)*Ydd + Q3(3)*ud + Q3(4)*udd;
%---update resetting model's estimates and outputs
    eddr = edr; edr = er;
    O = [Yd; Ydd; ud; udd; edr; eddr];
    K = Pr*O*inv(lambdaReset*eye(1) + O'*Pr*O);
    Pr = ( eye(6) - K*O' ) * Pr / lambdaReset ;
    Yest = O'*Qr;
    er = Y - Yest;
    Qr = Qr + K*er;
    if Qr(3)<0.01 % Enforce a b_min, as discussed in text
        Qr(3) = 0.01;
    end
    Ymr = Qr(1)*Yd + Qr(2)*Ydd + Qr(3)*ud + Qr(4)*udd;
%---update free-running model's estimates and outputs
    eddf = edr; edf = ef;
    O = [Yd; Ydd; ud; udd; edf; eddf];
    K = Pf*O*inv(lambdaFree*eye(1) + O'*Pf*O);
    Pf = ( eye(6) - K*O' ) * Pf / lambdaFree ;
    Yest = O'*Qf;
    ef = Y - Yest;
    Qf = Qf + K*ef;
    if Qf(3)<0.01 % Enforce a b_min, as discussed in text
        Qf(3) = 0.01;
    end
    Ymf = Qf(1)*Yd + Qf(2)*Ydd + Qf(3)*ud + Qf(4)*udd;
%---Switchg models according to either:
    % "selectModel_Index.m" (performance index switching)
    % "selectModel_Htest.m" (hypothesis test switching)
    oldModSel = ModSel;

```

```

        selectModel_Index; %removed: selectModel_Htest;
        saveModSel(k) = ModSel;
%---Using the selected model ModeSel decide how to perform control
%---This method below is the MODIFIED model arrangement method
        if ModSel ~= oldModSel %***
            if ModSel == 1 %***
                ContSel = 4; %***
                Qr = Q1; Pr = iP*eye(6); %***
                edr = 0; er = 0; %***
                sum1=0; sum2=0; sum3=0; sum4=0; sum5=0; %***
            end %***
            if ModSel == 2 %***
                ContSel = 4; %***
                Qr = Q2; Pr = iP*eye(6); %***
                edr = 0; er = 0; %***
                sum1=0; sum2=0; sum3=0; sum4=0; sum5=0; %***
            end %***
            if ModSel == 3 %***
                ContSel = 4; %***
                Qr = Q3; Pr = iP*eye(6); %***
                edr = 0; er = 0; %***
                sum1=0; sum2=0; sum3=0; sum4=0; sum5=0; %***
            end %***
            if ModSel == 4 %***
                ContSel = 4; %***
            end %***
            if ModSel == 5 %***
                ContSel = 5; %***
            end %***
        end %***
% Select theta, denoted Q, for performing certainty equivalence control
        if ContSel == 1, Q = Qr; end
        if ContSel == 2, Q = Qr; end
        if ContSel == 3, Q = Qr; end
        if ContSel == 4, Q = Qr; end
        if ContSel == 5, Q = Qf; end
        saveContSel(k) = ContSel;
%---Generate certainty equivalence control signal
        Qu = [Q(1) Q(2) Q(3) Q(4)]';
        Ou = [Y Yd 0 ud];
        u = (Yr-Ou*Qu)/Qu(3);
%---Save results for time-step k
        saveY(k) = Y;
        saveYr(k) = Yrd;
        saveU(k) = u;
        saveW(k) = w;
        saveE(k) = Ec;
        saveQ(:,k) = Q;
        if Ec^2 >= 4, saveAbContE(k) = 4; end
        if Ec^2 < 4, saveAbContE(k) = Ec^2; end
    end % END OF LOOP
% end of file: simModifiedMMAC.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: selectModel_Index.m (by: Alex Campbell)
% This file is used by parent classes throughout Chapter 3.
% It perform performance index B switching.

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This file is used by a parent files which have defined the following:
%
% double: em1,em2,em3,em4,em5 - the most recent model error sample for
%                               model 1,2,3,4, and 5, respectively.
% double: sum1,sum2,sum3,sum4,sum5 -
%                               the sum of all past modeling errors for the 5
%                               models, initialized at 0 by parent class.
% double: numbFixedModels - the number of fixed models present
% double: d1 and d2          - performance index B weights defined by parent.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List terms not defined in previous lists:
%   J1,J2,J3,J4,J5 - performance index B values for models 1-5.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
em1 = (Ym1-Y)^2;
em2 = (Ym2-Y)^2;
em3 = (Ym3-Y)^2;
em4 = (Ymr-Y)^2;
em5 = (Ymf-Y)^2;
sum1 = d3*sum1 + em1;
sum2 = d3*sum2 + em2;
sum3 = d3*sum3 + em3;
sum4 = d3*sum4 + em4;
sum5 = d3*sum5 + em5;
J1 = d1*em1 + d2*sum1;
J2 = d1*em2 + d2*sum2;
J3 = d1*em3 + d2*sum3;
J4 = d1*em4 + d2*sum4;
J5 = d1*em5 + d2*sum5;
if numbFixedModels < 3 % numbFixedModels selects how many fixed models to
    J3 = inf;          % use, in some simulations not all 3 are used.
    if numbFixedModels < 2
        J2 = inf;
        if numbFixedModels < 1
            J1 = inf;
        end
    end
end
[scrap ModSel] = min([J1 J2 J3 J4 J5]);
% end of file: selectModel_Index.m

```

B.7 Comparing Performance of the Classical and Modified Model Arrangement with Incorrect Switching

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: selectRandModel.m (by: Alex Campbell)
% This file is used by parent classes
% simClassicalMMAC.m and simModifiedMMAC.m

```



```
% This file is used in place of selectModel_Index.m (performance index
% switching) to instead perform arbitrary switching.
% This file performs the simulations for Section 3.6.5, titled:
% "Comparing Performance of the Classical and Modified Model Arrangement
% with Incorrect Switching"
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
switchSeed = 55;
randn('seed',switchSeed); % reproducible randn numbers
ModSel = ceil(5*rand);
% end of file: selectRandModel.m
```

B.8 Detailed Demonstration of Hypothesis Test Switching

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoDetailedHTest.m (by: Alex Campbell)
% This file is used to create the simulations for Section 4.4.2, titled:
% "Detailed Demonstration of Hypothesis Test Switching"
% This file uses:
%   getRandPlantsAndModels.m
%   (randomly assigns parameter jumps and ensures that fixed models are in
%   the neighborhood all possible plant states)
%   simClassicalArrangHypTest.m
%   (this is not included in the appendix because it is identical to
%   simClassicalMMAC.m using selectModel_Htest.m for model selection
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% List terms not defined in previous lists:
% myOutMSE - This is the MSE of control error resulting from simulating
%            'simClassicalArrangHypTest.m' for 'simL' time-steps.
% modOutlier - this is the outlier threshold discussed in Appendix A.1.
% maxSTD - this is the maximum value of the running standard
%          deviation, as discussed in the main body of the thesis
%          after equation (4.2).
% Tr - rejection threshold for hypothesis testing.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% Setup initial conditions
close all; clear all; clc;
maxSTD = .2;
modOutlier = .015;
simL = 400;
lambdaReset = .94;
lambdaFree = .97;
initModelSelection = 1;
noiseSeed = sum(clock)*1000;
randn('seed',noiseSeed);
myNoise = 0.05*randn(1,simL);
for k = 1:simL, myRef(k) = sin(2*pi*k/55) + sin(2*pi*k/165); end
poleDist = 1; zeroDist = 1; magnitudeDist = 1; modelDist = 0.3;
numbFixedModels = 3;
```

```

plantModelSeed = sum(clock)*1000
getRandPlantsAndModels; % Gets 3 random plant states and 3 models in the
                        % neighborhood of those plant states
Tr = 3.5; % Rejection threshold
%%% Now perform Simulation
simClassicalArrangHypTest;
myOutMSE = mse(saveAbContE)
save saveSimClassicResult1;
% end of file: demoDetailedHTest.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: getRandPlantsAndModels.m (by: Alex Campbell)
% This file randomly assigns parameter jumps and ensures that fixed
% models are in the neighborhood of all possible plant states.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This file is used by a parent files which have defined the following:
%
% double: plantModelSeed - the seed for generating random parameters
% double: zeroDist - the max parameterization error for fixed model zeros
% double: poleDist - the max parameterization error for fixed model poles
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rand('seed',plantModelSeed);
magnitude = 1;
zero1 = zeroDist*(0.5-rand);
pole1 = poleDist*(0.5-rand);
pole2 = poleDist*(0.5-rand);
pState1 = [ (pole1+pole2), (-pole1*pole2), magnitude, (-magnitude*zero1),
            -0.5, 0.4];
modelMagnitude = magnitude + modelDist*(0.5-rand);
zero1 = zero1 + modelDist*(0.5-rand);
pole1 = pole1 + modelDist*(0.5-rand);
pole2 = pole2 + modelDist*(0.5-rand);
model1 = [ (pole1+pole2), (-pole1*pole2), modelMagnitude,
            (-modelMagnitude*zero1), -0.5, 0.4];
magnitude = magnitude + magnitudeDist;
zero1 = zeroDist*(0.5-rand);
pole1 = poleDist*(0.5-rand);
pole2 = poleDist*(0.5-rand);
pState2 = [ (pole1+pole2), (-pole1*pole2), magnitude, (-magnitude*zero1),
            -0.5, 0.4];
modelMagnitude = magnitude + modelDist*(0.5-rand);
zero1 = zero1 + modelDist*(0.5-rand);
pole1 = pole1 + modelDist*(0.5-rand);
pole2 = pole2 + modelDist*(0.5-rand);
model2 = [ (pole1+pole2), (-pole1*pole2), modelMagnitude,
            (-modelMagnitude*zero1), -0.5, 0.4];
magnitude = magnitude + magnitudeDist;
zero1 = zeroDist*(0.5-rand);
pole1 = poleDist*(0.5-rand);
pole2 = poleDist*(0.5-rand);
pState3 = [ (pole1+pole2), (-pole1*pole2), magnitude, (-magnitude*zero1),
            -0.5, 0.4];
modelMagnitude = magnitude + modelDist*(0.5-rand);

```

```

zero1 = zero1 + modelDist*(0.5-rand);
pole1 = pole1 + modelDist*(0.5-rand);
pole2 = pole2 + modelDist*(0.5-rand);
model3 = [ (pole1+pole2), (-pole1*pole2), modelMagnitude,
           (-modelMagnitude*zero1), -0.5, 0.4];
% cause parameter jumps:
for k = 1:simL
    if k < 200,      plantParameters(:,k) = pState1; plantStateNumber(k) = 1;
    elseif k < 400, plantParameters(:,k) = pState3; plantStateNumber(k) = 3;
    elseif k < 600, plantParameters(:,k) = pState2; plantStateNumber(k) = 2;
    elseif k < 800, plantParameters(:,k) = pState1; plantStateNumber(k) = 1;
    elseif k < 1000, plantParameters(:,k) = pState2; plantStateNumber(k) = 2;
    elseif k < 1200, plantParameters(:,k) = pState3; plantStateNumber(k) = 3;
    else,           plantParameters(:,k) = pState1; plantStateNumber(k) = 1;
    end
end
% end of file: getRandPlantsAndModels.m

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: selectModel_Htest.m (by: Alex Campbell)
% This file is used by parent classes throughout Chapter 4.
% It perform hypothesis test switching.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This file is used by a parent files which have defined the following:
%
% double: AbsEc - the initial value of AbsEc is defined.
% double: r - the initial running standard deviation sample size.
%           the sum of all past modeling errors for the 5
%           models, initialized at 0 by parent class.
% double: maxSTD - the maximum magnitude of the running standard deviation
%                 defined after equation (4.2) in Section 4.
% double: Tr - the rejection threshold
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
J1 = abs(Ym1-Y);
J2 = abs(Ym2-Y);
J3 = abs(Ym3-Y);
J4 = abs(Ymr-Y);
J5 = abs(Ymf-Y);
AbsEcd = AbsEc; % 'AbsEcd' is the past value of 'AbsEc' (delay 1)
AbsEc = abs(Ec); % 'AbsEc' is absolute vlaue of the current control
               % error sample
r = r + 1; % r denotes the running standard deviation sample size.
           % r initializes at 2 (because of equation (4.2) in Section 4)
saveEc(r) = AbsEcd; % save the past r samples of AbsEcd
% calculate the running standard deviation:
stdE = ( (sum( (saveEc - mean(saveEc)).^2 ))/(r - 1) )^0.5;
% see if running STD has saturated to maxSTD:
if stdE > maxSTD
    stdE = maxSTD;
end
% calculate t-test result:
tr = (sum([abs(EcNorm)]))/(stdE/(1^0.5));
% decide to accept/reject null hypothesis
H = 0; % null hypothesis accepted automatically,
       % now check if it should be rejected:

```

```

if tr >= Tr %because parameter jump was detected, reject null hypothesis
    H = 1;
    r = 1; % sample size of standard deviation reset to 1.
    EcNorm = 0.001;
    clear saveEc;
    saveEc(1) = EcNorm;
    [scrap ModSel] = min([J1 J2 J3 J4 J5]);
end
% end of file: selectModel_Htest.m

```

B.9 Hypothesis Test Switching with Poor Choices for Rejection Threshold, T_r

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoBadHTEST.m (by: Alex Campbell)
% Simulates how hypothesis testing works when the threshold Tr is set to a
% too large value and too small value.
% This file is used to create the simulations for Section 4.4.3, titled:
% "Hypothesis Test Switching with Poor Choices for Rejection Threshold,Tr"
% This file uses:
%   getRandPlantsAndModels.m
%   (randomly assigns parameter jumps and ensures that fixed models are in
%   the neighborhood all possible plant states)
%   simClassicalArrangHypTest.m
%   simModifiedArrangHypTest.m
%   (these two files are not included in the appendix because they are
%   identical to simClassicalMMAC.m and simModifiedMMAC.m using
%   selectModel_Htest.m for model selection)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Setup initial conditions
close all; clear all; clc;
maxSTD = .2;
modOutlier = .015;
simL = 1400;
lambdaReset = .94;
lambdaFree = .97;
initModelSelection = 1;
for k = 1:simL, myRef(k) = sin(2*pi*k/55) + sin(2*pi*k/165); end
poleDist = 1; zeroDist = 1; magnitudeDist = 1; modelDist = 0.3;
numbFixedModels = 3;
noiseSeed = sum(clock)*1000;
randn('seed',noiseSeed);
myNoise = 0.05*randn(1,simL);
plantModelSeed = sum(clock)*1000
getRandPlantsAndModels;
save initialCond
% Now perform Simulations
load initialCond;
Tr = inf;
simClassicalArrangHypTest;
myOutMSE = mse(saveAbContE)

```

```

save saveSimClassicResult1;
load initialCond;
Tr = 0;
simClassicalArrangHypTest;
myOutMSE = mse(saveAbContE)
save saveSimClassicResult2;
load initialCond;
Tr = inf;
simModifiedArrangHypTest;
myOutMSE = mse(saveAbContE)
save saveSimModifResult1
load initialCond;
Tr = 0;
simModifiedArrangHypTest;
myOutMSE = mse(saveAbContE)
save saveSimModifResult2
% end of file: demoBadHTEST.m

```

B.10 Comparing Performance Index B Switching and Hypothesis Test Switching in the Absence of Disturbances

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoNoDisturbance.m (by: Alex Campbell)
% This file is used to create the simulations for Section 4.4.4, titled:
% "Comparing Performance Index B Switching and Hypothesis Test Switching
% in the Absence of Disturbances"
% This file uses:
%   getRandPlantsAndModels.m
%   (randomly assigns parameter jumps and ensures that fixed models are in
%   the neighborhood all possible plant states)
%   simClassicalArrangPerfIndex.m
%   simClassicalArrangHypTest.m
%   (these two files are not included in the appendix because they are
%   identical to simClassicalMMAC.m using first selectModel_index.m
%   switching then selectModel_Htest.m switching.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Setup initial conditions
close all; clear all; clc;
maxSTD = .2;
modOutlier = .1;
simL = 1400;
lambdaReset = .94;
lambdaFree = .97;
for k = 1:simL, myRef(k) = sin(2*pi*k/55) + sin(2*pi*k/165); end
poleDist = 1; zeroDist = 1; magnitudeDist = 1; modelDist = 0.5;
numbFixedModels = 3;
initModelSelection = 4;
noiseSeed = sum(clock)*1000

```

```

randn('seed',noiseSeed);
myNoise = 0.0*randn(1,simL);
plantModelSeed = sum(clock)*1001
getRandPlantsAndModels;
maxTuneAttempts = 100;
save initialCond
%%% Now Perform Simulations
clear all;
load initialCond;
d3 = 0.0000
simClassicalArrangPerfIndex;
save saveSimPIndex
clear all;
load initialCond;
Tr = 0.0000
simClassicalArrangHypTest;
save saveSimHTest
% end of file: demoNoDisturbance.m

```

B.11 Comparing Performance Index B and Hypothesis Test Switching with Noise Variance

$$\sigma^2 = 0.025$$

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FILE: demoTuning4Methods.m (by: Alex Campbell)
% This file is used to create the simulations for Section 4.4.5, titled:
% "Comparing Performance Index B and Hypothesis Test Switching with Noise
% Variance ?2 = 0.025"
% This file uses:
% getRandPlantsAndModels.m
% (randomly assigns parameter jumps and ensures that fixed models are in
% the neighborhood all possible plant states)
% simClassicalArrangPerfIndex.m
% simClassicalArrangHypTest.m
% simModifiedArrangHypTest.m
% simModifiedArrangPerfIndex.m
% (these files are not included in the appendix because they are
% identical to simClassicalMMAC.m and simModifiedMMAC.m, with different
% switching methods already shown in code.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% Setup initial conditions
close all; clear all; clc;
maxSTD = .2;
modOutlier = 1;
simL = 1400;
lambdaReset = .97;
lambdaFree = .99;
for k = 1:simL, myRef(k) = sin(2*pi*k/55) + sin(2*pi*k/165); end
poleDist = 1; zeroDist = 1; magnitudeDist = 1; modelDist = .3;
numbFixedModels = 3;

```

```

initModelSelection = 4;
save initialCond
noiseSeed = sum(clock)*1000
randn('seed',noiseSeed);
myNoise = 0.025*randn(1,simL);
plantModelSeed = sum(clock)*1001
getRandPlantsAndModels;
maxTuneAttempts = 100;
%%% Perform tuning, 100 attempts, for each model arrangement and
%%% switching method, 100
for tuneAttempt = 1:maxTuneAttempts
    d3 = tuneAttempt/(maxTuneAttempts); saveD3(tuneAttempt) = d3;
    simClassicalArrangPerfIndex;
    sim1MSE(tuneAttempt) = mse(saveAbContE);
    Tr = tuneAttempt/10; saveTr(tuneAttempt) = Tr;
    simClassicalArrangHypTest;
    sim2MSE(tuneAttempt) = mse(saveAbContE);
    d3 = tuneAttempt/(maxTuneAttempts); saveD3(tuneAttempt) = d3;
    simModifiedArrangPerfIndex;
    sim3MSE(tuneAttempt) = mse(saveAbContE);
    Tr = tuneAttempt/10; saveTr(tuneAttempt) = Tr;
    simModifiedArrangHypTest;
    sim4MSE(tuneAttempt) = mse(saveAbContE);
end
save saveTuningProcess
%%% Re-Simulate each method one final time, using the tuned values,
%%% this is for plotting the best result.
clear all;
load saveTuningProcess
[value index] = min(sim1MSE)
d3 = saveD3(index)
load initialCond;
simModifiedArrangPerfIndex;
save saveSim1
clear all;
load saveTuningProcess
[value index] = min(sim2MSE)
Tr = saveTr(index)
load initialCond;
simModifiedArrangHypTest;
save saveSim2
clear all;
load saveTuningProcess
[value index] = min(sim3MSE)
d3 = saveD3(index)
load initialCond;
simModifiedArrangPerfIndex;
save saveSim3
clear all;
load saveTuningProcess
[value index] = min(sim4MSE)
Tr = saveTr(index)
load initialCond;
simModifiedArrangHypTest;
save saveSim4
% end of file: demoTuning4Methods.m

```