SYSTEM HEALTH MONITORING USING MULTIPLE-MODEL ADAPTIVE ESTIMATION TECHNIQUES

by

STANLEY RYAN SIFFORD

Presented to the Faculty of the Graduate School of

The University of Texas at Arlington in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT ARLINGTON

December 2016

Acknowledgements

First, I would like to thank my supervising professor, Dr. Kamesh Subbarao. Thank you for the many years you worked with me, even through all the distractions. You were patient when I had to focus on tasks for my employer. You did not give up on me when I was called to go to Africa to help repair a bush hospital in Zimbabwe and the delay after my return, as my heart remained in Africa. You continued to work with me. I cannot number the countless times you met with me to discuss and guide my research. Without your care and support, I would not have completed this effort.

In addition, I would like to acknowledge the Department of Mechanical and Aerospace Engineering. Particularly Dr. Ratan Kumar, Dr. Kent Lawrence, Dr. Seiichi Nomura and Dr. Panos Shiakolas for the effort on my committee. Furthermore, I would like to thank Debi Barton who has spent countless time guiding me through the university process to complete my degree. Both the university and I are blessed to have you.

Furthermore, I would like to thank my wife, Rashel, for supporting me through it all. It has been a long journey that we both are ready to complete. Paige, Katy and Chloe, I know each of you are grateful that this project is concluding. There will be no more days for me to say "I cannot play right now because I have to work on school." I have longed for the day to come home and just be with my family.

Finally, I want to thank my Lord and Savior who put me on this journey. A journey that began many years ago. Therefore, I take this journey under His direction knowing the sacrifice that He made.

> Hebrews 12:1-3 (NASB[1])
> [1]Therefore, since we have so great a cloud of witnesses surrounding us, let us also lay aside every encumbrance and the sin which so easily entangles us, and let us run with endurance the race that is set before us, [2]fixing our eyes on Jesus, the author and perfecter of faith, who for the joy set before Him endured the cross, despising the shame, and has sat down at the right hand of the throne of God. [3]For consider Him who has endured such hostility by sinners against Himself, so that you will not grow weary and lose heart.

December 9, 2016

[1] Scripture quotation taken from the New American Standard Bible® (NASB), Copyright © 1960, 1962, 1963, 1968, 1971, 1972, 1973, 1975, 1977, 1995 by The Lockman Foundation. Used by permission. www.Lockman.org

Abstract

SYSTEM HEALTH MONITORING USING MULTIPLE-MODEL ADAPTIVE ESTIMATION TECHNIQUES

Stanley Ryan Sifford, Ph.D.

The University of Texas at Arlington, 2016

Supervising Professor: Kamesh Subbarao

Monitoring system health for fault detection and diagnosis by tracking system parameters concurrently with state estimates is approached using a new multiple-model adaptive estimation (MMAE) method. This novel method is called GRid-based Adaptive Parameter Estimation (GRAPE). GRAPE expands existing MMAE methods by using new techniques to sample the parameter space. GRAPE expands on MMAE with the hypothesis that sample models can be applied and resampled without relying on a predefined set of models. GRAPE is initially implemented in a linear framework using Kalman filter models. A more generalized GRAPE formulation is presented using extended Kalman filter (EKF) models to represent nonlinear systems. GRAPE can handle both time invariant and time varying systems as it is designed to track parameter changes.

Two techniques are presented to generate parameter samples for the parallel filter models. The first approach is called selected grid-based stratification (SGBS).

SGBS divides the parameter space into equally spaced strata. The second approach uses Latin Hypercube Sampling (LHS) to determine the parameter locations and minimize the total number of required models. LHS is particularly useful when the parameter dimensions grow. Adding more parameters does not require the model count to increase for LHS. Each resample is independent of the prior sample set other than the location of the parameter estimate. SGBS and LHS can be used for both the initial sample and subsequent resamples. Furthermore, resamples are not required to use the same technique. Both techniques are demonstrated for both linear and nonlinear frameworks.

The GRAPE framework further formalizes the parameter tracking process through a general approach for nonlinear systems. These additional methods allow GRAPE to either narrow the focus to converged values within a parameter range or expand the range in the appropriate direction to track the parameters outside the current parameter range boundary. Customizable rules define the specific resample behavior when the GRAPE parameter estimates converge. Convergence itself is determined from the derivatives of the parameter estimates using a simple moving average window to filter out noise. The system can be tuned to match the desired performance goals by making adjustments to parameters such as the sample size, convergence criteria, resample criteria, initial sampling method, resampling method, confidence in prior sample covariances, sample delay, and others.

Table of Contents

Acknowledgements iii

Abstract v

List of Illustrations x

List of Tables xiv

CHAPTER 1 Introduction 1

1.1 Background and Motivation 1

1.2 Research Problem Statement 3

1.3 Fault Detection and Diagnosis Literature Review 5

1.4 Overview of Fault Detection and Diagnosis 9

1.4.1 Faults, Failure and Malfunction 10

1.4.2 System Behavior, Parameter Space and Fault Diagnosis 14

1.4.3 Fault Diagnostic Problem 16

1.5 Fault Tolerant Control 18

1.6 Overview of State and Parameter Estimation 22

1.6.1 State Estimation 22

1.6.2 Parameter Estimation with the Extended Kalman Filter 22

1.6.3 MMAE for Parameter Estimations 24

1.6.4 Relevant MMAE Literature 26

1.6.5 Defining MMAE Models Using Latin Hypercube Sampling 30

1.7 Research Objectives and Document Organization 30

CHAPTER 2 Problem Formulation 33

2.1 Common System Model 33

2.2 Linear Systems Overview 34

2.3 The Kalman Filter 41

2.4 Extended Kalman Filter for Parameter Estimation 45
2.5 Multiple-Model Adaptive Estimation Framework 47
2.5.1 MMAE Derivation 49
2.5.2 Summary of the MMAE Algorithm 54
CHAPTER 3 Choosing the Hypothesis Models 58
3.1 FDD Parameter Space 58
3.2 Parameter Space for MMAE 59
3.3 Model Location and Sample Size 61
3.4 Selected Grid-Based Stratification (SGBS) 62
3.5 Sampling Techniques 66
3.5.1 Evaluating MMAE as a type of Monte Carlo Method 66
3.5.2 IID Sampling 69
3.5.3 Latin Hypercube Sampling 74
3.5.4 Evaluating Confidence 81
3.6 Summary 89
CHAPTER 4 GRid Adaptive Parameter Estimation (GRAPE) 91
4.1 GRAPE Algorithm 91
4.1.1 Requirements and Assumptions 96
4.1.2 Dynamic Sampling 96
4.1.2.1 Selected Grid-Based Stratification 97
4.1.2.2 Latin Hypercube Sampling 98
4.1.2.3 Convergence Criteria 99
4.1.2.4 Resampling 100
4.2 Application to Fault Detection and Diagnosis 100
4.3 Application for System Identification 101

4.4 Linear GRAPE Framework Results 102
4.4.1 Selected Grid-Based Stratification Results 107
4.4.2 EKF Parameter Estimation using Augmented Parameter Models 110
4.4.3 Latin Hypercube Sampling Results 112
4.4.4 Discussion of Results 115
CHAPTER 5 Generalized Framework for Nonlinear GRAPE using the Extended Kalman Filter 116
5.1 EKF GRAPE 116
5.2 FDD Approach for EKF GRAPE 122
5.2.1 Resample Behavior within Center Range 125
5.2.2 Resample Behavior near Parameter Limits 127
5.3 Updated Convergence Determination Approach 130
5.3.1 Simple Moving Average of Parameter Estimate Derivative 130
5.3.2 Criteria for Convergence 132
5.3.3 Resample Rules 133
5.4 Closing Remarks on EKF GRAPE 135
CHAPTER 6 Exploring the Duffing Oscillator using GRAPE 136
6.1 Overview of the Duffing Oscillator 136
6.2 Chosen form of Duffing Oscillator 138
6.3 EKF Model of Duffing Oscillator 138
6.4 Range of Values Considered 139
6.4.1 Harmonic Excitation 142
6.4.2 Nonpersistent Excitation 143
6.5 Determining a General Configuration for EKF GRAPE 146
6.5.1 Tuning the EKF Models 148

6.5.2 Option to Use Prior State Estimate Covariance 150
6.5.3 Combined Effect of Convergence Threshold and Delay 152
6.5.4 Resampling Rules 156
6.6 Simulations 158
6.6.1 Persistent Sinusoidal Input Simulations 159
6.6.2 Pulse Input Simulations 167
6.6.3 Doublet Input Simulations 174
6.6.4 Simulation Summary 181
6.7 Summary of GRAPE Simulations using the Duffing Oscillator 183
CHAPTER 7 Summary and Closing Remarks 184
7.1 Summary of Research Achievements 184
7.2 Future Work 186
7.3 Closing Remarks 192
Appendix A Acronyms 193
References 196
Biographical Statement 215

List of Illustrations

Figure 1-1 Typical system model 3
Figure 1-2 Classification of diagnostic algorithms (adapted from [7]) 7
Figure 1-3 Classification of dynamic systems (adapted from [46]) 11
Figure 1-4 Additive and multiplicative faults (adapted from [3]) 12
Figure 1-5 Fault classification (adapted from [1]) 12
Figure 1-6 Temporal fault classification (adapted from [3]) 12
Figure 1-7 Progression of unattended fault (adapted from [3]) 14

Figure 1-8 Overlapping system behavior regions (adapted from [1]) 15

Figure 1-9 3-D Parameter space with behavior regions 16

Figure 1-10 Parameter space 17

Figure 1-11 Fault tolerant control systems (adapted from [1]) 21

Figure 1-12 Overview of multiple-model Adaptive Estimation (adapted from [53] and [57]) 25

Figure 1-13 Moving-bank from predefined models (adapted from [76]) 29

Figure 2-1 Idealized linear spring and viscous dashpot (adapted from [89]) 35

Figure 2-2 Mechanical elements represented by a spring (from [89]) 36

Figure 2-3 Simple MDOF spring-mass-damper system (adapted from [89]) 38

Figure 2-4 Typical Kalman filter application (adapted from [95]) 42

Figure 2-5 Multiple-model adaptive estimation (adapted from [53] and [57]) 48

Figure 2-6 Flow of MMAE steps in [53] 57

Figure 3-1 2-dimensional parameter space behavior definition example 60

Figure 3-2 2-Dimensional parameter space example 64

Figure 3-3 3-dimensional parameter space example 64

Figure 3-4 Uniform distribution, U(a,b) 72

Figure 3-5 Uniform IID samples, U(0,1) 72

Figure 3-6 Gaussian IID samples, N(0,1) 73

Figure 3-7 Gaussian distribution, N(μ,σ^2) (adapted from [106]) 73

Figure 3-8 2-d LHS example 76

Figure 3-9 3-d LHS example 77

Figure 3-10 1-d LHS based on CDF 81

Figure 3-11 Univariate LHS and IID mean vs. sample size 83

Figure 4-1 Linear GRAPE high-level summary 92

Figure 4-2 GRAPE vs. moving-bank approach to parameter Space 92

Figure 4-3 GRAPE MMAE high-level summary 93
Figure 4-4 Dynamic resizing 96
Figure 4-5 Selected grid-based stratification example 98
Figure 4-6 Hydroelectric servo model 104
Figure 4-7 Component models 104
Figure 4-8 Model development 105
Figure 4-9 Behavior regions 106
Figure 4-10 Synthetic data 107
Figure 4-11 SGBS parameter estimates for c=0.20, k=2.5, 100 models 108
Figure 4-12 SGBS parameter estimates for c=0.70, k=2.5, 100 models 108
Figure 4-13 SGBS parameter estimates for c=1.00, k=2.5, 100 models 109
Figure 4-14 SGBS parameter estimates for c=1.30, k=2.5, 100 models 109
Figure 4-15 EKF parameter estimates for c=0.20, k=2.5 110
Figure 4-16 EKF parameter estimates for c=0.70, k=2.5 111
Figure 4-17 EKF parameter estimates c=1.00, k=2.5 111
Figure 4-18 EKF parameter estimates c=1.30, k=2.5 112
Figure 4-19 LHS state estimate results for c=1.30, k=2.5, 20 models 113
Figure 4-20 LHS parameter estimate results for c=1.30, k=2.5, 20 models 113
Figure 4-21 LHS state estimate results for c=1.30, k=2.5, 50 models 114
Figure 4-22 LHS parameter estimate results for c=1.30, k=2.5, 50 models 114
Figure 5-1 Nonlinear GRAPE high-level summary 118
Figure 5-2 GRAPE EKF MMAE high-level summary 119
Figure 5-3 GRAPE parameter range regions 123
Figure 5-4 GRAPE range regions for 3-dimensional parameter space 124
Figure 5-5 GRAPE range regions for 3-dimensional parameter space 124

Figure 5-6 Resampling in region of normal operation 128

Figure 5-7 Lower boundary region resample (upper is mirrored) 129

Figure 5-8 Noisy derivative to resample flag 131

Figure 5-9 Example using resample rule 1 133

Figure 6-1 Contribution of kx and βx^3 terms 140

Figure 6-2 Evaluating the effect of changing c 141

Figure 6-3 Evaluating the effect of changing c 141

Figure 6-4 Chosen synthetic system 143

Figure 6-5 Sine, pulse and doublet input with constant frequency(ω) and amplitude (γ) 145

Figure 6-6 Pulse input synthetic data 145

Figure 6-7 Doublet input synthetic data 146

Figure 6-8 EKF tuning process 149

Figure 6-9 Selecting P_0 150

Figure 6-10 Effect of P_0 on SGBS state estimate covariances 151

Figure 6-11 Effect of P_0 on SGBS parameter estimates 152

Figure 6-12 System with frequent resampling (threshold to high) 153

Figure 6-13 System with infrequent resampling (threshold to low) 153

Figure 6-14 System with nearly fixed interval resampling 154

Figure 6-15 System with reasonable interval resampling 155

Figure 6-16 Rule 2 resampling 156

Figure 6-17 Resampling with rule 1 157

Figure 6-18 SGBS state estimates and residuals example 161

Figure 6-19 SGBS parameter estimates example 162

Figure 6-20 SGBS parameter derivatives and convergence flags 163

Figure 6-21 LHS state estimates and residuals 164

Figure 6-22 LHS parameter estimates 165
Figure 6-23 LHS parameter derivatives and convergence flags 166
Figure 6-24 SGBS state estimates and residuals 168
Figure 6-25 SGBS parameter estimates 169
Figure 6-26 SGBS parameter derivatives and convergence flags 170
Figure 6-27 LHS state estimates and residuals 171
Figure 6-28 LHS parameter estimates 172
Figure 6-29 LHS parameter derivatives and convergence flags 173
Figure 6-30 SGBS state estimates and residuals 175
Figure 6-31 SGBS parameter estimates 176
Figure 6-32 SGBS parameter derivatives and convergence flags 177
Figure 6-33 LHS state estimates and residuals 178
Figure 6-34 LHS parameter estimates 179
Figure 6-35 Parameter derivatives and convergence flags 180
Figure 6-36 Parameter derivatives and convergence flags 182
Figure 7-1 System with reasonable interval resampling 188
Figure 7-2 Resampling continues after final convergence 189
Figure 7-3 Alternate input signals 190

List of Tables

Table 1-1 Miljković's overview of fault detection methods (list extracted from [3]) 7
Table 1-2 Areas using FDD within sample literature 9
Table 1-3 Fault detection and diagnosis steps (adapted from [1]) 17
Table 1-4 Fault tolerant controller tasks (adapted from [1]) 18
Table 1-5 Steps to achieve fault tolerance (adapted from [1]) 19

Table 1-6 MMAE applications in FDD 27

Table 1-7 Considerations when using Maybeck's moving-bank MMAE (MBMMAE) (adapted from [75]) 28

Table 1-8 Areas of research using Maybeck's moving-bank MMAE (MBMMAE) 29

Table 2-1 SDOF systems (simplified single mass models) (extracted from [89]) 37

Table 2-2 Kalman filter algorithm 42

Table 2-3 Continuous-time Kalman filter (adapted from [53]) 43

Table 2-4 Continuous-discrete time Kalman filter (adapted from [53]) 43

Table 2-5 Discrete-time Kalman filter (adapted from [53]) 44

Table 2-6 Continuous-discrete extended Kalman filter (adapted from [53]) 45

Table 2-7 Miller's summary of MMAE basic assumptions (adapted from [52]) 54

Table 2-8 Summary of detailed steps of discrete-time MMAE of [53] 55

Table 4-1 GRAPE algorithm detailed steps 94

Table 4-2 Fault behavior definitions required for FDD 101

Table 4-3 Variable descriptions for hydroelectric servo 103

Table 4-4 Behavior identification of electrohydraulic servo 105

Table 5-1 GRAPE EKF algorithm detailed steps 120

Table 5-2 GRAPE range parameter options 126

Table 5-3 GRAPE range parameter calculated values 127

Table 5-4 GRAPE convergence options and calculated values 132

Table 6-1 Coefficient values considered for duffing equation 141

Table 6-2 Characteristics of final duffing oscillator system 142

Table 6-3 General configuration of EKF GRAPE 147

Table 6-4 Approach for tuning EKF filters 148

Table 6-5 Simulation summary 159

Table 7-1 Summary of potential future work/modifications to GRAPE 187

Table A-1 List of acronyms used 194

CHAPTER 1

Introduction

This chapter presents the relative background for the topics of fault diagnosis and parameter estimation. The contents form the foundation on which GRid-based Adaptive Parameter Estimation (GRAPE) is built. GRAPE explores the topic of monitoring system health for fault detection and diagnosis through new multiple-model adaptive estimation framework to concurrently estimate system parameters and system states. The capability of estimating system parameters, rather than assuming their values are constant, provides a measure of the system health, because the system parameters define the system behavior.

The chapter begins by identifying relevant background and motivation for study. The research problem statement is then provided. Following a high-level literature review of Fault Diagnosis and Detection (FDD), an overview of FDD is provided. Then a brief introduction to Fault Tolerant Control is covered. Relevant topics of state and parameter estimation are introduced. The chapter concludes with the objectives of the dissertation research and a summary of the organization of the remaining chapters.

1.1 Background and Motivation

The study of system health is motivated by the desire to maintain adequate control of the system to prevent a failure. All types of designed systems are susceptible to faults. The system could be a simple aerospace, chemical, electrical,

industrial, financial and mechanical or any other system designed to perform some type of action dependent on the system input. A fault may degrade the performance of a system or cause a complete failure. The fault could potentially put lives in danger. Therefore, methods for detecting and handling a potential fault are critical.

The complexity of the methods employed to detect and handle faults are directly proportional to the associated risk of a fault. A low-risk application may just turn on a warning light such as a low battery charge warning on the dashboard of a car. A high-risk application could be a complicated system to shut down a process at a petroleum refinery to prevent an massive financial loss or even an explosion. Somewhere in the middle could be a fault-tolerant control process that accommodates minor control actuator failures in aircraft allowing the pilot to safely land a plane.

A systematic approach must be used to evaluate nature of a system and whether or not it is functioning properly. FDD is presented through the approach taken by Blanke et al. in [1] and Isermann in [2]. Adaptive parameter estimation, specifically the multiple-model adaptive estimator, will be presented as a way to identify the current system parameters concurrently with the state estimation. The techniques developed can be applied to various components of the system shown in Figure 1-1. For the purpose of this research, the focus is on the system parameters of the plant (amplifier, actuator and load in Figure 1-1). The process and measurement noise characteristics are assumed to be zero mean with a specific

variance. Initially, simple systems such as the spring-mass-damper will be used to illustrate concepts that will be applied to a wider scope of linear problems. These concepts are later expanded into a more general nonlinear approach.

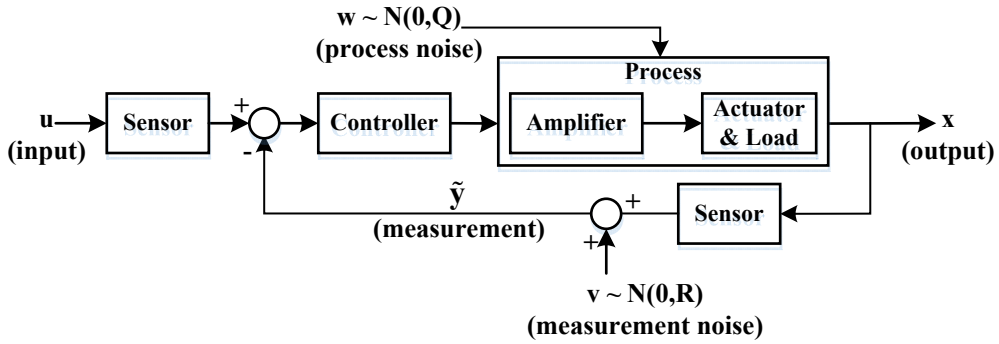


Figure 1-1 Typical system model

1.2 Research Problem Statement

The motivation of this research is to develop methods for concurrent parameter and state estimation of dynamic systems in the framework of FDD. The specific purpose is to identify system behavior from the input and output measurements using the new MMAE approach called GRAPE. The majority of the research effort was focused on developing and testing the GRAPE algorithm. GRAPE is a derivative of existing MMAE approaches. GRAPE provides a novel approach to use MMAE techniques for observing system parameters and their relationship to predefined regions of behavior. The GRAPE framework covers both linear and nonlinear systems. GRAPE provides a tool for fault detection. It can be extended to aid the diagnosis of a faults.

This research and development of GRAPE makes three primary contributions to FDD and MMAE techniques in general.

- The first contribution is the adaptation of MMAE to a formalized framework that supports system health monitoring. A parameter range is defined with limits for each parameter. This forms the upper and lower bounds of where the parameter is explored. The current estimate range reflects the location of latest parameter estimate with a minimum and maximum bound within the parameter estimate range. These definitions coupled with a modified MMAE approach support concurrent parameter and state estimates within the FDD framework.
- The second contribution is the way GRAPE handles MMAE model resampling. Resampling deviates from other research efforts by defining behaviors of resampling and then performing a resample independent of any predefined model set. Each resampled model set represents an entirely new estimate range for the parameters based on the location of the previous parameter estimate.
- The third contribution is the method of sampling the parameter space. Two sampling methods are provided within the GRAPE framework. The first method, called Selected Grid-Based Sampling (SGBS), breaks the parameter space into uniformly distributed strata. A region within the overall parameter range is initially selected and processed through the algorithm to focus on the

region with the current parameter estimate. The second method, called Latin Hypercube Sampling, uses a Monte Carlo/design of experiment method to sample the same parameter range randomly with the purpose of minimizing the number of samples models required.

GRAPE was developed for the purpose of studying system health, but the framework is applicable to other uses where MMAE can be applied. The second and third contributions add to the current state of art for MMAE in general.

1.3 Fault Detection and Diagnosis Literature Review

Fault detection and diagnosis (FDD) is a broad field that has been recently expanded to cover many areas of industry. Early FDD is critical to minimize the effects of plant or process downtime, extend the life of equipment, provide a safe environment and reduce manufacturing costs.[3] FDD ranges from simple limit checking of measurements to complicated analysis of large-scale processes.[3]

There are many researchers and authors focused on the basic methodologies that have texts that are often referenced. Three have shown up often in the literature. Isermann's *Fault-Diagnosis Systems* [2] is widely referenced. The Blanke et al. text *Diagnosis and Fault-Tolerant Control* [1] is also cited often. J. Chen and R.J. Patton are other authors often cited. One such example is their text *Robust Model-Based Fault Diagnosis for Dynamic Systems* [4].

The FDD literature spans many professional organizations including the Association for Computing Machinery (ACM), the American Institute of Aeronautics and Astronautics (AIAA), the American Society of Mechanical Engineers (ASME), and the Institute of Electrical and Electronics Engineers (IEEE). It also covers many science, technology, engineering and mathematic publications. The literature is too vast to summarize in just one document. An early survey can be found from Stengel in [5]. In 1997, Patton provided a summary of the technology of the time in [6]. The following paragraphs provide summaries extracted from two more recent surveys by Venkatasubramanian et al. (three-part series [7-9]in 2003) and Miljković ([3] in 2011).

FDD is of particular interest to petrochemical industries and chemical engineering. Venkatasubramanian et al. [7] describe the twenty billion dollar annual loss within the petroleum industry due to abnormal events. FDD is a central component of abnormal event management (AEM) which is the "timely detection, diagnosis and correction of abnormal conditions of faults in a process." Early FDD is important to prevent productivity loss.[7] Venkatasubramanian et al. present a three-part discussion of FDD in [7-9], dividing the area of study into quantitative models, qualitative models and process history approaches respectively. Figure 1-2 represents the diagnostic method classifications summarized in [7]. Reference [9] concludes with a discussion of integrating FDD methods into hybrid approaches to overcome limitations of individual strategies.

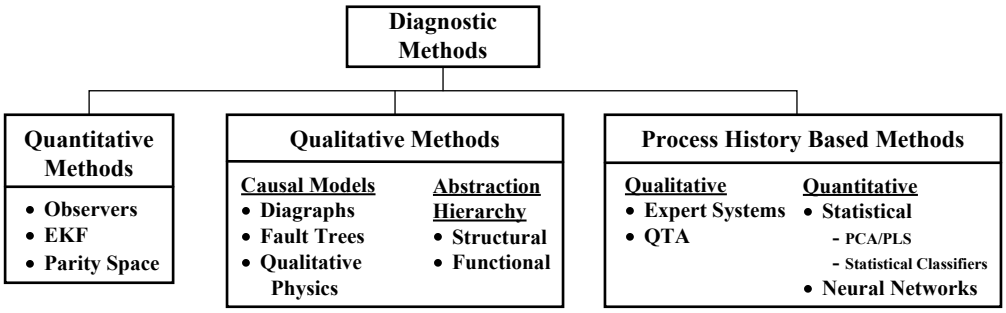


Figure 1-2 Classification of diagnostic algorithms (adapted from [7])

Miljković provides another survey in [3] reiterating the need for early FDD in "high-cost and safety-critical processes." Miljković's paper divides the fault detection methods into the methods and categories listed in Table 1-1.

Table 1-1 Miljković's overview of fault detection methods (list extracted from [3])

A. Data Methods and Signal Models • Limit checking and trend checking • Data analysis (PCA) • Spectrum analysis and parametric models • Pattern recognition (neural nets)
B. Process Model Based Methods • Parity equations • State observers • Parameter estimation • Nonlinear models (neural nets)
C. Knowledge Based Methods • Expert systems • Fuzzy logic

Miljković states that until the "early 1990s most research and development in fault detection was limited to nuclear power plants, aircraft, process plants, the automobile industry and national defense." FDD is now established in many industries.[3] Some of the major areas are summarized in Table 1-2. This table

originated from Miljković's ten application areas. The table has been augmented with literature references covering a wide range of fields representing a sample of available applications. The list is by no means exhaustive.

The focus of this research is within quantitative model-based methods shown in Figure 1-2. The MMAE method used to evaluate system health can be further classified as parameter estimation in Table 1-1. GRAPE performs concurrent estimation of the system states and parameters. The current value of each are used to evaluate the system health against a predefined set of behavior.

Table 1-2 Areas using FDD within sample literature

- Actuators [10-22]
- Aircraft [23]
- Automotive Systems
- Bearings and Machinery
- Chemical Process [7-10]
- Communications [24]
- Computer Networks [11]
- Crack Propagation [25]
- Electrical Motors [26]
- Electromechanical Systems [27]
- Gas Turbines
- Heating, Ventilation, Air Conditioning [28]
- Rotating Machine and Engines [29]
- Manufacturing
- Medical/Biological [30-33]
- Power Systems/Supplies [34]
- Pumps
- Robotics [35, 36]
- Sensors [20, 37, 38]
- Structural [20, 39, 40]
- Steam Turbines
- Vibration [41]
- Wind Turbines [42]
- Unmanned Vehicle Systems [10, 20, 43-45]

1.4 Overview of Fault Detection and Diagnosis

In this section, the focal concepts of FDD are expanded. The definitions of fault, failure and malfunction are covered first. Then the concepts of system behavior and parameter space are introduced. An introduction to fault tolerant control will be covered in section 1.5.

1.4.1 Faults, Failure and Malfunction

A fault is defined as something that changes the behavior of a system so that the system no longer satisfies its purpose. The change can be internal such as a component failure or an external change such as ambient conditions. Furthermore, the fault could be an actual error in the system design that may go undetected until a specific operating condition is reached.

For the purpose of this research, a fault is a system change that produces an undesired system response. This definition follows Blanke et al.'s description of a fault as a deviation for the dynamical system structure or system parameters from the nominal situation.[1] The system change may be a result of changes to the system parameters which are coefficients or related to the system model. The parameters may be constant for time-invariant systems or varying with time for time-variant systems. A parameter change to a value outside a predefined acceptable tolerance will indicate a fault. The system response is evaluated through the measurement process. Furthermore, an input may be outside of the acceptable input range and generate a fault. For the purpose of this research, the input is assumed to be bounded within acceptable levels for the system of interest.

Figure 1-1 illustrates a simple linear system model. The system parameters typically are a subset of the coefficients composing the system matrix A. System parameters, θ, may also include the variance of the process noise and measurement

noise. The noise is typically white, zero mean Gaussian. Its characteristics are summarized in the notation w~N[0,1], where N means normal or Gaussian, 0 is the mean and 1 is the variance (or square of the standard deviation). For the techniques of this research, the system does not need to have a linear constraint. The framework will work on all classifications of systems shown in Figure 1-3 including the general nonlinear time varying system. The necessary framework for the system model is provided in section 2.1.

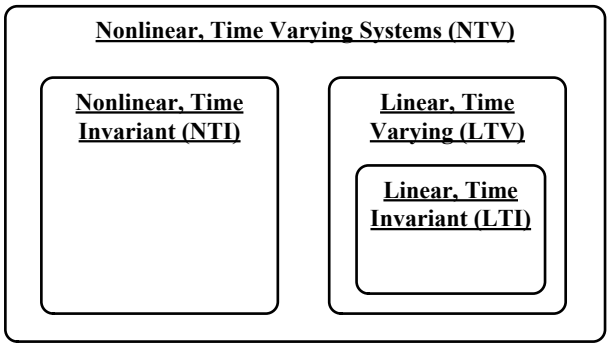


Figure 1-3 Classification of dynamic systems (adapted from [46])

Blanke et al. [1] further distinguish faults into types and classifications. The major types are additive or multiplicative shown in Figure 1-4. Additive faults enter the model equations through superposition or addition. Whereas multiplicative faults have a scaling effect.

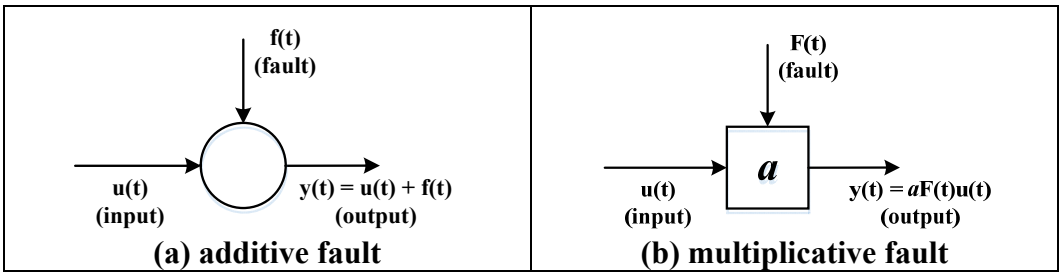


Figure 1-4 Additive and multiplicative faults (adapted from [3])

There are three main classifications to faults: plant faults, sensor faults and actuator faults as shown in Figure 1-5. Plant faults change the dynamical input/output properties of the system. Sensor faults can generate erroneous sensor readings. For actuator faults, the plant properties are not affected, but the influence of the controller on the plant is interrupted or modified. Faults may be further delineated according to their size and temporal behaviors such as abrupt, increasing (incipient) or intermittent as shown in Figure 1-6.[1-3]

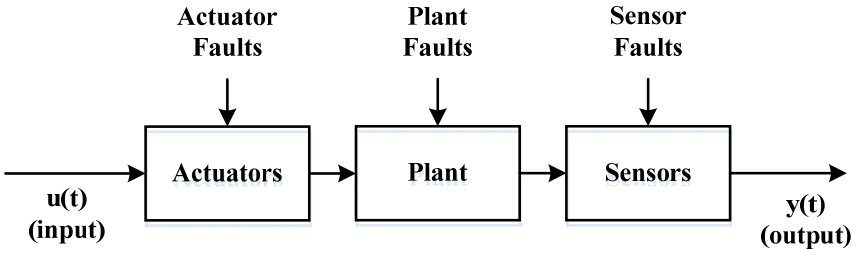


Figure 1-5 Fault classification (adapted from [1])

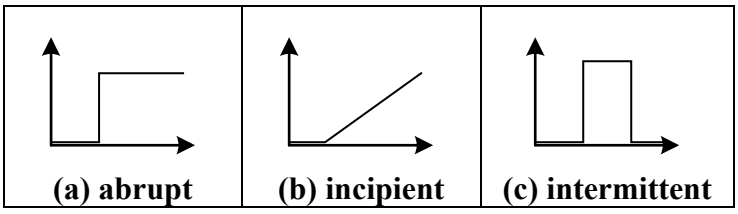


Figure 1-6 Temporal fault classification (adapted from [3])

It is important to distinguish between a fault and a failure. A fault causes a change than can be worked around. A failure describes the inability of a system or component to accomplish its function. A failure is an irrecoverable event. Fault-tolerant control has to prevent the fault from causing failure at the system level. This research has narrowed the scope to observing plant behavior that can help identify faults. The focus is on the method to determine the system characteristics, primarily the parameters of interest and characteristics of the process and measurement noise. While this effort does stop short of true fault-tolerant control, the methodologies are provided for a solid framework to influence design and decisions towards fault-tolerant control. Blanke et al. may be reviewed for further reading on the subject of fault-tolerant control.

An additional term is defined according to Isermann in [2]. A malfunction is defined as an intermittent irregularity.[3] Both failures and malfunctions develop from faults as illustrated in Figure 1-7. In the GRAPE framework, a malfunction's observability will depend on its duration. If the duration of a malfunction is shorter than GRAPE's ability to respond to the change, then the malfunction may not be detectable.

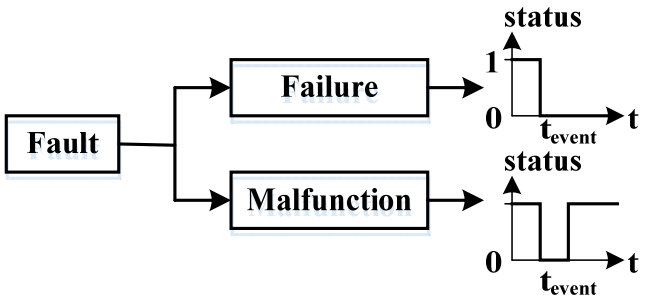


Figure 1-7 Progression of unattended fault (adapted from [3])
(1 represents a properly operating system)

1.4.2 System Behavior, Parameter Space and Fault Diagnosis

Blanke et al. [1] describe the system behavior, B, as the set of all possible input/output (I/O) pairs (u,y). The system behavior can be divided into subsets representing a faultless system, B_0, and a faulty system, B_{fault}, and failed system, B_{failed}, as shown in Figure 1-8. The union of each ($B_0 \cup B_{fault} \cup B_{failed}$) does not have to encompass the entire set of possible behaviors. For the purpose of this research, the regions are defined uniquely per equations (1-1) and (1-2) to facilitate detailed examination. Analysis becomes more complicated if the intersections of each behavior set are non-empty as shown in Figure 1-8.

$$B = B_0 \cup B_{fault} \cup B_{failed} \tag{1-1}$$

$$B_0 \cap B_{fault} = \varnothing; \quad B_0 \cap B_{failed} = \varnothing; \quad B_{fault} \cap B_{failed} = \varnothing \tag{1-2}$$

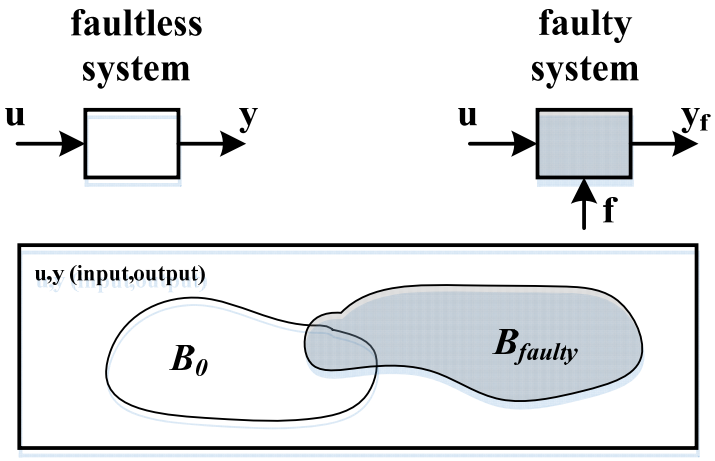


Figure 1-8 Overlapping system behavior regions (adapted from [1])

Blanke et al. [1] further describe fault diagnosis as a method of determining which behavior the I/O pairs belongs. This definition will be used for the case of nonlinear additive faults. For the remainder of the fault cases, a more appropriate description of system behavior is based on the parameter space. This is illustrated for a three-dimensional parameter space in Figure 1-9. The nominal performance region, $B_{nominal}$, is the designed operating point and including some acceptable tolerance for implementation. A region of degraded performance is represented by the faulty region, $B_{faultly}$. In this region, either the performance is acceptable, or a fault-tolerant controller can recover by adjusting the system plant model to match actual parameters. Failure is the region outside of the faulty region. Drastic measures would be taken, such as a system shutdown, for the failure case. These concepts will be further explored within the problem formulation of Chapter 2. The parameter space is defined in detail in section 2.4.3.1.

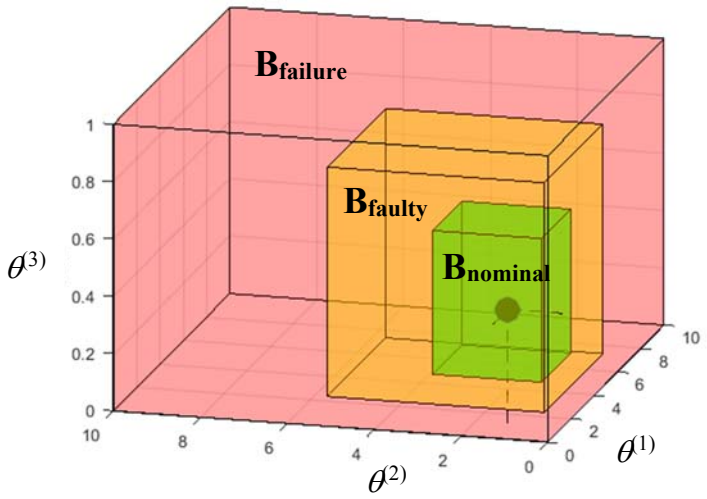


Figure 1-9 3-D Parameter space with behavior regions

1.4.3 Fault Diagnostic Problem

The diagnostic problem is to determine the correct fault *f* case or cases from the current subset of input values and associated measurements $\left(\mathbf{U}, \tilde{\mathbf{Y}}_{\mathbf{m}}\right)$. If there is no fault, then the fault is f_0 which is the faultless case. The FDD steps are summarized in Table 1-3. The method used for diagnosis follows the consistency-based diagnosis of Blanke et al. [1] Consistency-based diagnosis determines if the measurement subset $\left(\mathbf{U}, \tilde{\mathbf{Y}}_{\mathbf{m}}\right)$, called input/output (I/O) pair is consistent with the nominal system behavior, B_{nominal}. If the system is not part of the nominal behavior *B*, then a fault f_i has likely occurred. The diagnosis step compares fault candidate, f_i, to a predefined set of faults. The final diagnosis may be a single fault or a set of faults if the fault behavior definitions are not unique. There is also the possibility that the behavior is outside the defined regions. In that case, the fault candidate f_i

is not detectable. This research modifies the consistency based approach to determine system behavior from the current estimated parameter values.

Table 1-3 Fault detection and diagnosis steps (adapted from [1])

1. **Fault Detection** – determines if a fault has occurred
2. **Fault Isolation** – determines the location where the fault occurred
3. **Fault Identification and Estimation** – determines identity and magnitude of the fault

For the purpose of this research, the behavior is defined by the current parameter estimate. As mentioned in section 1.4.2, the fault regions within the parameter space are assumed to be unique for simplicity of analysis. The faultless behavior described by B_0 is also unique in its definition. This assumption is not required. However, it would require additional methods beyond the parameter space evaluation to isolate the faults.

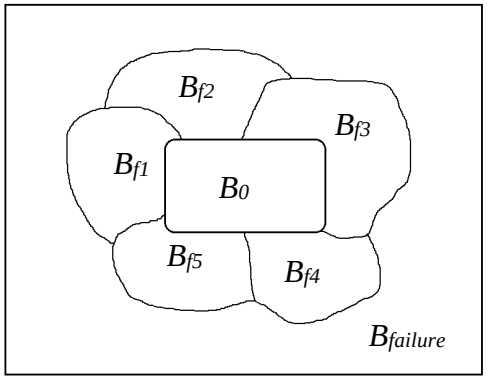


Figure 1-10 Parameter space

1.5 Fault Tolerant Control

The concept of fault tolerant control (FTC) is introduced from the point of view of Blanke et al. in [1]. This section provides the necessary motivation for the research problem statement. It is desired to find and stop the effects of a fault before the fault degrades the system performance or becomes a failure. The process of detecting and stopping the propagation of the effects of a fault is handled by a fault tolerant controller. The fault tolerant controller interacts between a given system (plant) and the controller. The fault tolerant controller performs or controls the following tasks listed in Table 1-4.

Table 1-4 Fault tolerant controller tasks (adapted from [1])

- handles the feedback or feedforward control law
- handles decision-making layer that determines the control configuration
- reacts to the existence of a fault by adjusting the control to handle the faulty behavior of the plan

From an outside observer of the overall system, the system is considered fault tolerant if the fault is not visible. This does not mean that the system will not give status to what state it is in. Fault tolerant simply means the system satisfies its intended design even in the presence of a fault.

There are two distinct steps to achieve fault tolerance: fault detection and diagnosis, followed by control redesign. These steps are described in Table 1-5. Both steps are performed by a supervision system that oversees the FTC process in the fault tolerant controller.

Table 1-5 Steps to achieve fault tolerance (adapted from [1])

1. **Fault Detection and Diagnosis** – detect and identify the fault 2. **Control Redesign** – adaptation of controller to the appropriate algorithm and parameters for the feedback controller

The basic fault tolerant control systems are illustrated in Figure 1-11. FDD is handled by a diagnosis algorithm using a system model with the input and measured output. The algorithm estimates the states and evaluates the residuals. The residuals are the errors between the system state estimates and measurements. The model includes assumed properties for both the process noise acting on the plant and the measurement noise on the output sensor. Fault tolerance is either made by an adaptation/accommodation process in the controller or reconfiguration to an alternate control scheme.

Historically, the simplest method of FTC is limit-checking of system parameters. Other methods of control have limited fault tolerance built-in. For example, both robust and adaptive control are fault tolerant to an extent by design. Robust control tolerates change in the plant dynamics and performs according to its desired intent. For robust control, fault tolerance is achieved without changing the controller and thus is called passive fault tolerant. Robust controllers are suboptimal and exist only for a restricted set of changes of the plant. Therefore, there is a tradeoff between robustness and performance. For adaptive control, the controller adapts to the plant changes. This adaptation is a type of active fault

tolerance. Adaptive control has historically been limited to linear models and slowly varying parameters.[1] However, recent research has shown that MMAE adaptive methods can be constructed for nonlinear systems [47] and handle abrupt changes[48]. The derivation of MMAE provided in section 2.5.1 sets up the foundation upon which the nonlinear GRAPE framework is built.

The focus of this research is limited to the FDD part of FTC. A hypothetical fault tolerant controller can control operation within a nominal range and acceptable faulty adaptive range. This fault tolerant controller could use this information to provide a system reconfiguration or shutdown in the failure case. Designing the fault tolerant controller to react and maintain fault tolerance is outside the scope of this research.

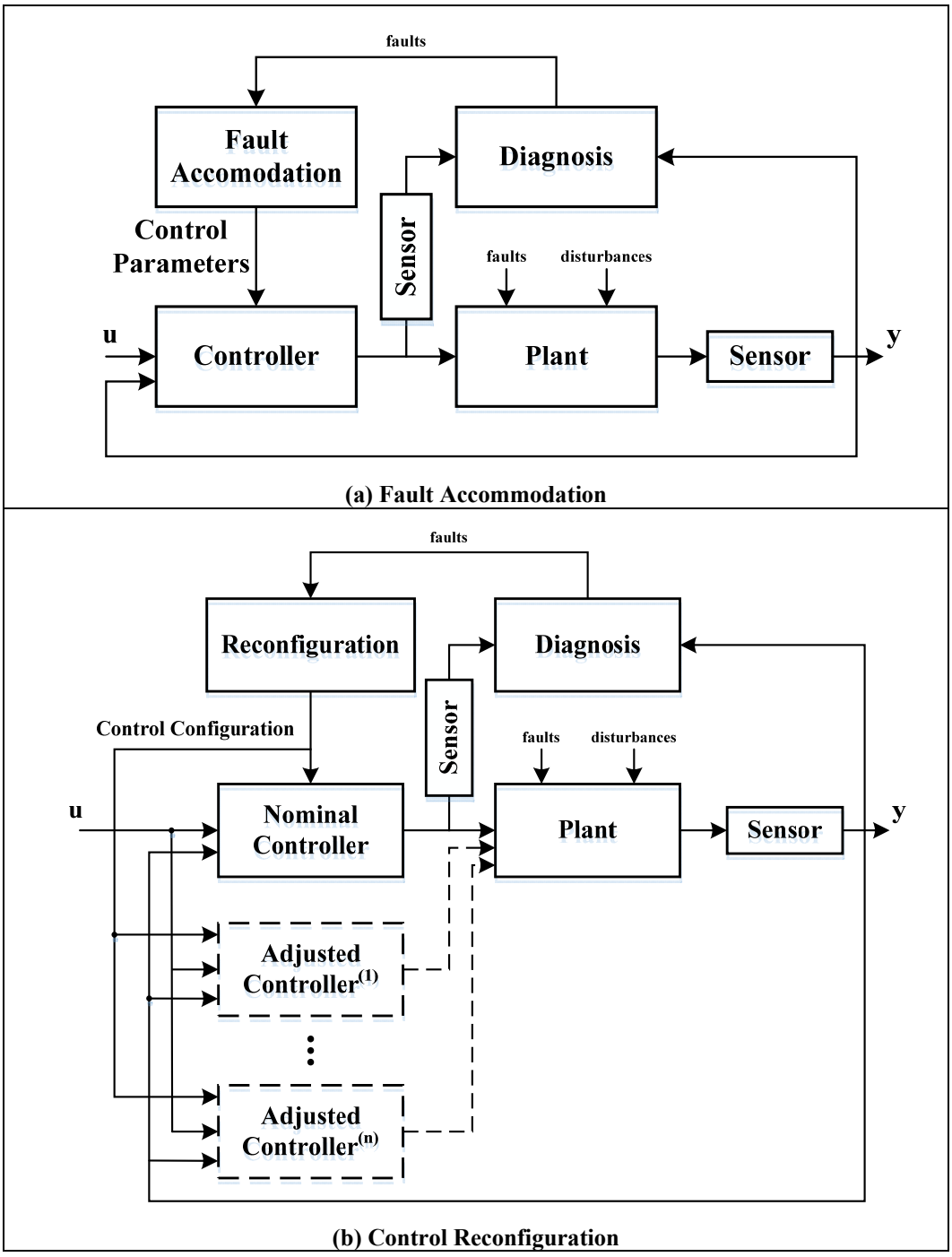


Figure 1-11 Fault tolerant control systems (adapted from [1])

1.6 Overview of State and Parameter Estimation

1.6.1 State Estimation

This research addresses the issue of concurrent state and parameter estimation. There are many methods to estimate the states from assumed models using the input and measured output. The most common and useful method for linear systems is the Kalman filter (KF).[49] Sorenson states that "…the Kalman filter represents the most widely applied and demonstrably useful result to emerge from the state variable approach of modern control theory."[50] The Kalman filter is capable of producing accurate estimates of the true states of a physical system, but it is limited by the system model.[47, 51, 52]

1.6.2 Parameter Estimation with the Extended Kalman Filter

When the parameters of the system model are uncertain, alternative techniques must be performed. One such method is to augment the parameters to the state model. There will be one extra state for each parameter of interest. The state space system becomes nonlinear. Therefore, a nonlinear filter such as the Extended Kalman Filter (EKF) is required. Crassidis and Junkins provides an example using the EKF to estimate model parameters in Example 3.6.[53](pgs. 190-192) The EKF takes advantage of the characteristic that the error dynamics can be accurately represented by first-order Taylor series expansion if the initial condition is close. This approximation is updated for each new data point. The

EKF is not precisely optimum; however, it has successfully been applied to a large range of nonlinear systems. [53]

Augmenting parameters to the EFK system model without additional measurements creates an observability issue. In practice, the EKF does well when initial conditions are known even when the observability rank is less than the number of states. However, extending the EKF to with more parameters for the purpose of FDD produces the possibility that the EKF will no longer converge. There are other nonlinear filtering method options such as nonlinear least squares and the unscented Kalman filter.[47] However, the observability issue will still be present. Increasing the number of measurements is one possible method to address the observability issue. However, adding measurements may not be feasible. Therefore, an alternate method is desired.

Alternative methods using system identification techniques can help provide an understanding of the system at the designed nominal state. Ljung provides recognized methods in [54]. Electronic and structural systems typically are typically designed for specific environmental requirements.[55] They will undergo finite element analysis followed by extensive verification. Vibration testing is used to validate or develop a transfer function representation. Testing includes frequency response analysis, modal parameter estimation and mode shape estimation and model verification.[56] The system is typically evaluated near its nominal design state. Changes can happen in real-time or even during repair. A

simple weld repair may change heat treated aluminum and alter its properties. The desire is to understand the system during real-time operation using online methods.

1.6.3 MMAE for Parameter Estimations

There are multiple alternative estimation methods from which to choose. Probabilistic methods such as the particle filters and bootstrap filters are options, but they are complicated to implement. Multi-hypothesis modeling techniques offer the use of simple representative models with methods to determine which model or combination of models represent the system. Two such methods often found in literature are the ensemble Kalman Filter (ENKF) and multiple-model adaptive estimation (MMAE). The method chosen for this research is MMAE.

MMAE creates a system of independent Kalman filters representing hypothesis models. The measurement residuals from the bank of filters are used with a hypothesis testing technique to weight the models. The hypothesis technique is typically maximum likelihood estimation; however, it can be other weighted hypothesis testing techniques. The resultant weights determine the most likely model or weighted average of "near" models that represents a system. The advantage of the MMAE technique is the models can be Kalman filters of simple state models as shown in Figure 1-12. These models can be extended to other more complicated models such as the EKF as the formulation of [53] does not assume linear model measurements. The parameter values are devised from the hypothesis testing technique based off a posterior probability distribution function (pdf) built

from the covariances of the measurement residuals. Much analysis is utilized to determine the models to use in the system. They are based on interest in accuracy and the regions of the FDD criteria. For the purpose of this research, the regions are defined by the area of interest within the possible parameter space.

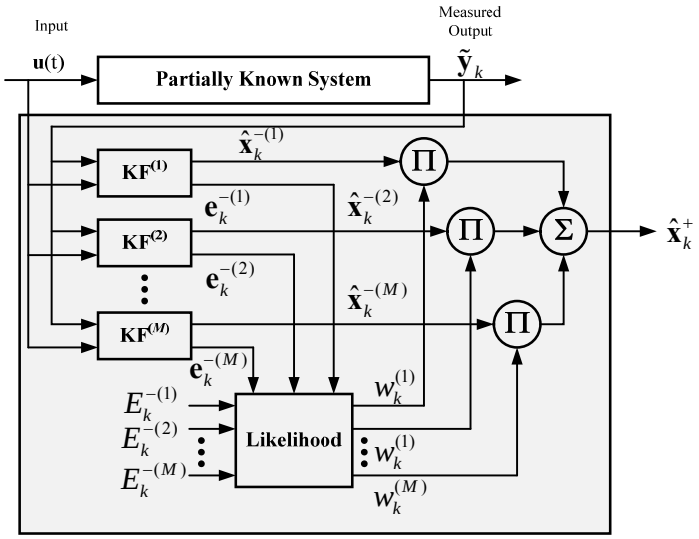


Figure 1-12 Overview of multiple-model Adaptive Estimation (adapted from [53] and [57])

For this research, MMAE is used to identify the system parameters from limited input/output (I/O) data. MMAE is also used to track the system parameters in real-time. Ultimately, FDD is capable using the parameter tracking of MMAE with simple monitoring algorithm defining the behavior according to the current parameter estimates. The MMAE approach has been used to successfully estimate both linear (Chapters 2 through 4) and a more general nonlinear approach (Chapters 5 and 6).

1.6.4 Relevant MMAE Literature

MMAE research has evolved significantly from its introduction in Magill's 1965 paper ([51]) when it "was beyond the capability of computer processing technology."[53](pg 249) MMAE can be easily implemented with a higher number of models today than when it was introduced because of increases in computing capability. The parallel nature of MMAE also lends itself to distributed computing approaches. There are several sources of literature where MMAE is applied or improved. One such area of interest is FDD applications. Table 1-6 provides a summary FDD applications of MMAE in literature. For MMAE research in general, there are two groups that stand out. The main researchers reviewed are Crassidis with the generalized multiple-model adaptive estimation (GMMAE) and Maybeck with his moving-block MMAE (MBMMAE, acronym adopted from [58]). Each researcher and their collaborators have contributed significantly to the state of the art of MMAE methods. Miller's dissertation on Modified-MMAE (M^3AE) is also worth mentioning.

Crassidis and Junkins present the base MMAE approach used for GRAPE in section 4.6.2. of [53]. Their recursive version is based on [59] and [60]. This MMAE version uses maximum likelihood estimation to determine the parameter and state estimates from the parallel models. In literature, Crassidis and Cheng introduce the generalized multiple-model adaptive estimation (GMMAE) in [61] with further research in [62], [63] and [64]. GMMAE's goal is to improve the

likelihood function used to update the weights of MMAE. GMMAE exploits the correlation between measurements by using history of the residuals. If only one correlation step (history) is used, then the GMMAE is equivalent to MMAE.[63] Further research is presented in [62], [63] and [64] using GMMAE to estimate noise parameters of a tracking system.

Table 1-6 MMAE applications in FDD

- general FDD [48, 65]
- aircraft [16-18, 66]
- aircraft interference/jamming, GPS/INS [67-69]
- unmanned aerial vehicle (UAV) [19]
- suction climbing robot [14, 15]
- determination of fault type after detection [70]
- model-matching [71]
- fault detection for Li-ion battery [72]
- sensor failures on mobile robot [73]

Maybeck and Hentz introduced moving-bank multiple-model adaptive estimation (MBMMAE) to literature in 1985.[57, 74] MBMMAE "is an attempt to reduce the computational loading associated with the implementation of a fullscale" MMAE.[57] MBMMAE provides increased state estimation and control performance. However, the performance increase and computational costs must be compared with using a lower granularity fixed bank MMAE. [57] A good summary of MBMMAE considerations is found in [75]. These comments are paraphrased in Table 1-7 below. Maybeck has collaborated with several other authors to apply

and improve the MBMMAE to areas of research, many of which are listed in Table 1-8.

Table 1-7 Considerations when using Maybeck's moving-bank MMAE (MBMMAE) (adapted from [75])

- the true parameter should lie within the filter's parameter space
- the level of discretization of the continuous parameter space directly impacts the accuracy of the estimate
- one of the filters should be close to the true parameter

The GRAPE approach used in the research effort has similarities to Maybeck's MBMMAE shown in Figure 1-13. Both approaches use filter banks that move. However, GRAPE uses Crassidis's MLE approach to estimate the filter parameters rather than MAP to select a discrete filter with the highest probability. Furthermore, the GRAPE filter banks are not limited to the discrete predefined banks of MBMMAE. Rather, the GRAPE filter bank is limited to a continuous parameter space defined by upper and lower bounds of the region of interest. Furthermore, GRAPE employs multiple methods to sample and resample the region.

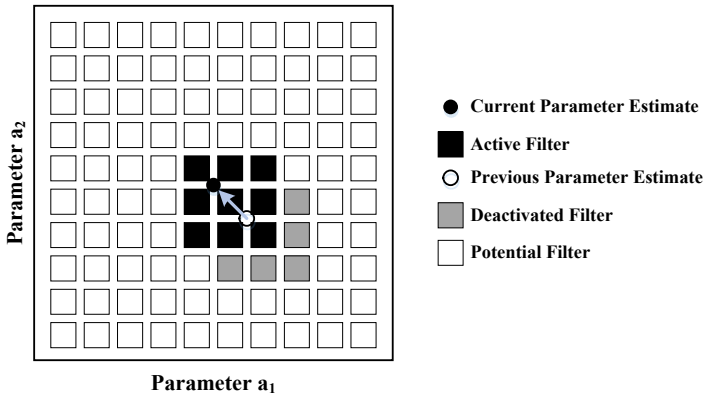


Figure 1-13 Moving-bank from predefined models (adapted from [76])

Table 1-8 Areas of research using Maybeck's moving-bank MMAE (MBMMAE)

- FDD on aircraft [16-18]
- FDD on an unmanned aerial vehicle (UAV) [19]
- FDD using enhanced motion and sizing improvements [48, 65]
- comparison with single Kalman filter and linear quadratic (LQG) controller [57]
- comparison with Standard Kalman Filter Bank (SKFB) and Generalized Likelihood Ratio (GLR) [77]
- evaluation of performance using space structure models [57, 78-81]
- FDD in aircraft interference/jamming, GPS/INS [67, 68]
- attenuate vibrations in a simulated large flexible space structure [76, 82-86]
- moving-bank decisions and parameter discretization [68]
- robustness evaluation [79, 80]
- aircraft tracking [58] (comparison with supervised learning of adaptive interacting multiple model (SLAIMM))
- optimizing design strategy [87]

One other MMAE approach is worth mentioning is Miller's Modified-MMAE (M^3AE). Miller introduces M^3AE to simultaneous estimate states from a single KF designed to accept parameter estimates from MMAE. His dissertation [52] tests the method using second-order mechanical systems with uncertain natural

frequency. M^3AE is then applied to a "13-state nonlinear integrated Global Positioning System/Inertial Navigation System (GPS/INS) system" where the variance of the GPS measurement noise affecting the GPS outputs" is the unknown parameter. Miller provides a design tool for analyzing and predicting M^3AE performance before simulation.

1.6.5 Defining MMAE Models Using Latin Hypercube Sampling

Latin Hypercube Sampling is a popular method to distribute samples over the region of interest. LHS guarantees that a sample is distributed in every stratum for each parameter. It is easy to implement, and the number of required samples does not grow in the manner other random sampling techniques does. So far, it has not been found in literature related to sampling MMAE models. Chapter 3 covers LHS and other sampling techniques.

1.7 Research Objectives and Document Organization

The primary objectives of this research were:

1. Develop an FDD framework to accurately estimate system states and parameters concurrently based on multiple-model adaptive estimation (MMAE) techniques
    a. Framework shall dynamically reallocate the models to improve the resultant estimates
    b. Framework should track both constant and time varying parameters
    c. Framework should handle both slow and abrupt changes in parameters

2. Implement both uniformly spaced sampling and sampling using the Latin Hypercube Sampling technique
3. Evaluate the relationship between Latin Hypercube Sampling (LHS) sample size (number of required models) and the performance of LHS sampling MMAE models
4. Expand the framework to handle nonlinear parameters
5. Demonstrate the framework on a nonlinear system

The research has met all objectives with the implementation of the GRAPE framework. The evidence is provided in this document. The document is organized in a systematic manner to present the necessary theoretical background to support an understanding of each topic followed by the research contributions and simulated results. Chapter 2 develops the problem formulation and provides the specifics of MMAE setting up the necessary theoretical background. Chapter 3 covers methods for choosing the hypothesis models. This research focuses on two primary methods, selected grid-based stratification and Latin Hypercube sampling. Chapter 4 develops the GRAPE framework from the basic algorithm through the sampling techniques and concludes with a linear system implementation. Chapter 5 develops a more generalize approach with the nonlinear GRAPE framework. A rigorous definition of the GRAPE framework is presented in respect to the FDD problem, each parameter, the methods used to measure parameter estimate convergence, and the process to trigger a resample. Chapter 6 takes a common nonlinear system, the Duffing oscillator, and applies nonlinear GRAPE framework

for study. A method for tuning the EKF models is also presented. Results are provided using both persistent sinusoidal input and nonpersistent signals. This document concludes in Chapter 7 by providing a summary of the research achievements including the primary contributions. Finally, an extensive discussion of potential future research is provided.

CHAPTER 2

Problem Formulation

The focus of this chapter is to present the mathematical framework for the concepts used for fault diagnosis through adaptive estimation. The parameters of interest are presented in the framework of the common system model. A brief linear system overview is provided. The parameter estimation techniques using the EKF and MMAE are then presented.

2.1 Common System Model

A complex system can be represented as a function of the system states ($\mathbf{x}$), initial values ($\mathbf{x}_0$), system parameters ($\boldsymbol{\theta}$), input (u), process noise (w) and measurement noise (v). This relationship is generalized in the measurement function $\tilde{\mathbf{y}}$ provided as

$$\tilde{\mathbf{y}} = \mathbf{f}\left(\mathbf{x}, \mathbf{x_0}, \boldsymbol{\theta}, u, w, v\right) \tag{2-1}$$

The system states are defined by the state-space representation of the system model. The state-space representation breaks the n^{th}-order system model into n first order differential equations. Each desired state is represented by its own differential equation. The states and their corresponding initial conditions are as follows

$$\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_i & \dots & x_{n-1} & x_n \end{bmatrix}^T \tag{2-2}$$

$$\mathbf{x}_0 = \begin{bmatrix} x_{01} & x_{02} & \dots & x_{0i} & \dots & x_{0n-1} & x_{0n} \end{bmatrix}^T \tag{2-3}$$

The parameter set θ contains the parameters necessary to represent the system. Typically, these parameters are the coefficients of the differential equations or some relationship to them. The parameter set could also include the noise characteristics if desired.

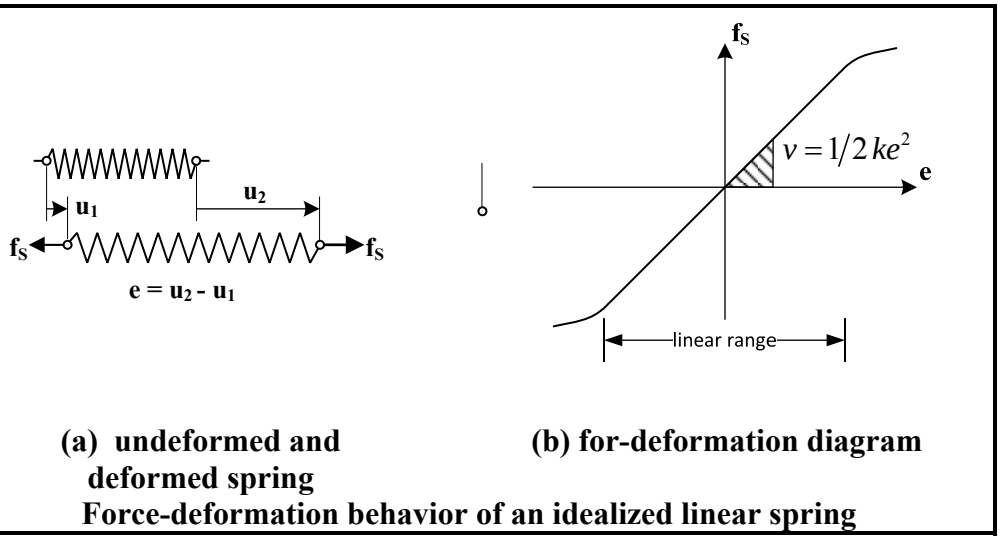
2.2 Linear Systems Overview

The linear system of interest is presented as Equation (2-4) which is an idealized version of Craig and Kurdila's fundamental single degree of freedom (SDOF) equation [56](Eqn 2.19, pg 29). This equation is also called the "fundamental equation in structural dynamics and linear vibration theory."[56] It is the basic equation of linear system analysis for second order differential equations.[88] In its nondescript form, the equation represents many linear systems including mechanical, electrical and a variety of others. This equation is the basis for second order linear system analysis of [46, 49, 53, 56, 88-94].

$$m\ddot{x} + c\dot{x} + kx = p(t) \tag{2-4}$$

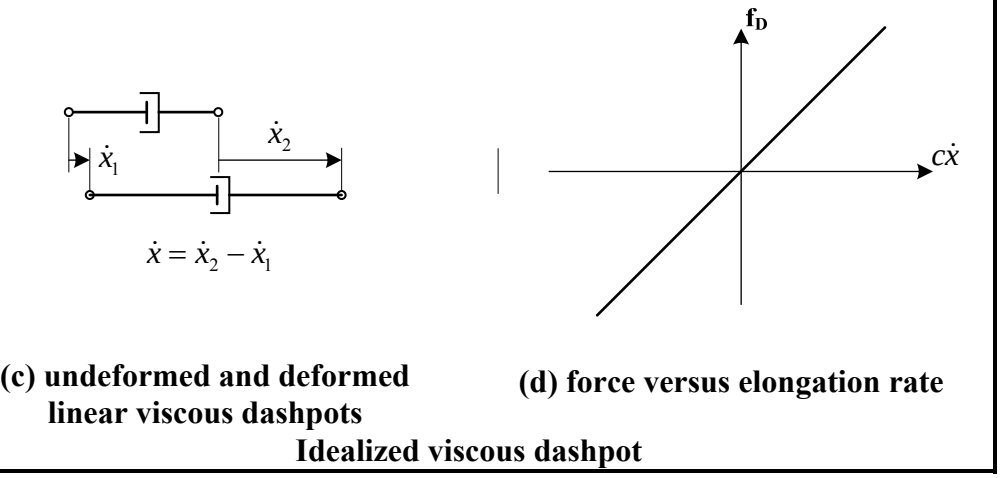
For second order mechanical systems, the equation represents the idealized spring-mass-damper where *m* is the mass, *c* is the viscous damping coefficient, *k* is the spring constant and *p(t)* (also called u) is the forcing or driving function (also called the load). When the driving function is sinusoidal, the resulting system is called a harmonic oscillator. The spring is typically assumed to be linear by limiting its operation to the linear region as shown in Figure 2-1. There are several

mechanical components that are represented with a linear spring or stiffness. Figure 2-2 provides a few.



(a) undeformed and deformed spring **(b) for-deformation diagram**

Force-deformation behavior of an idealized linear spring



(c) undeformed and deformed linear viscous dashpots **(d) force versus elongation rate**

Idealized viscous dashpot

Figure 2-1 Idealized linear spring and viscous dashpot (adapted from [89])

Damping is typically represented by the linear viscous dashpot model.[56]

$$f_d = c\dot{e} = c\left(\dot{x}_2 - \dot{x}_1\right) \tag{2-5}$$

The idealized version of damping used in (2-4) assumes that one end is fixed and there is only one velocity term.

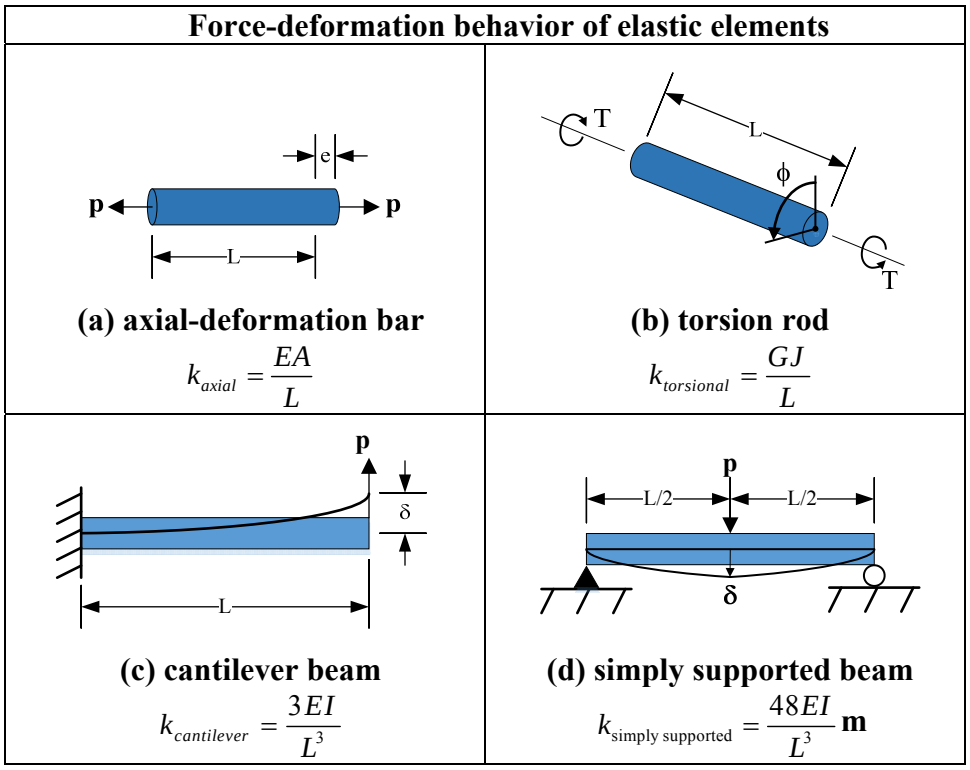


Figure 2-2 Mechanical elements represented by a spring (from [89])

SDOF systems are used to model a wide variety of systems. Several examples of SDOF systems are provided in Table 2-1. These systems can easily be expanded to the multiple degrees of freedom cases. For mechanical systems, the degree of freedom is defined by the number of mass or inertial elements**.**

Table 2-1 SDOF systems (simplified single mass models) (extracted from [89])

- Automotive suspension
- Engine Propeller System
- Hoisting system
- Open loop DC Motor speed control
- Motor speed control with speed feedback
- Position control system using DC motor
- Hydro-mechanical position servo system
- Pneumatic position servo system
- Electro-hydraulic position servo control system

The typical state model for equation (2-4) is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{c}{m} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \frac{1}{m} \begin{bmatrix} 0 \\ 1 \end{bmatrix} p(t) + \frac{1}{m} \begin{bmatrix} 0 \\ 1 \end{bmatrix} w \tag{2-6}$$

were the states x_1 and x_2 are the position and velocity respectively. The measurement model is

$$\tilde{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v \tag{2-7}$$

For simplicity, mass is assumed to be one ($m = 1$). With this assumption, the second order equation becomes

$$\ddot{x} + c\dot{x} + kx = p(t) \tag{2-8}$$

Furthermore, the state-space model becomes

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w \tag{2-9}$$

with the measurement model

$$\tilde{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v \tag{2-10}$$

Equations (2-9) and (2-10) are the basis of the linear methods presented in this research. Many variations are possible. The methods discussed are extendable to those variations.

The SDOF equation (2-4) can be easily extended to the generalized multiple degrees of freedom (MDOF) case shown below

$$\mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} = \mathbf{p}(t) \tag{2-11}$$

where **M** is the mass matrix $\left(\mathbf{M} \in \Re^{nxn}\right)$, **C** is viscous damping matrix $\left(\mathbf{C} \in \Re^{nxn}\right)$, **K** is the stiffness matrix $\left(\mathbf{K} \in \Re^{nxn}\right)$, and **p** is the load vector $\left(\mathbf{p} \in \Re^{nx1}\right)$. A simple MDOF system is shown in Figure 2-3.

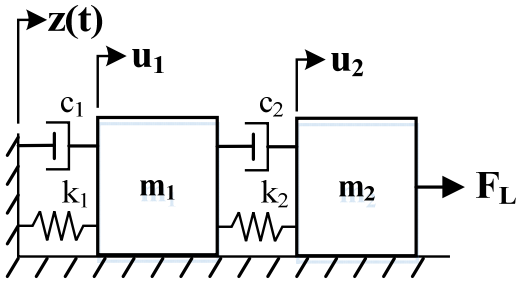


Figure 2-3 Simple MDOF spring-mass-damper system (adapted from [89])

For a simple second-order mechanical system, the parameters are the mass (m), damping (c), stiffness (k) and input coefficient (b) as shown in Equation (2-12). The multiplier, g, for the process noise is typically not called a system parameter.

$$m\ddot{x} + c\dot{x} + kx = bu + gw \tag{2-12}$$

The system parameters would be

$$\begin{gathered} \boldsymbol{\theta} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \theta_4 \end{bmatrix}^T \\ \text{or} \\ \boldsymbol{\theta} = \begin{bmatrix} m & b & c & k \end{bmatrix}^T \end{gathered} \tag{2-13}$$

Each parameter could be a known real value denoted as $\theta^{(i)}$ or defined over a region as

$$\boldsymbol{\theta}^{(i)} \equiv \left\{ \theta^i \in \Re^{nx1} \mid \theta_l^{(i)} \le \theta^i \le \theta_u^{(i)} \right\} \tag{2-14}$$

or equivalently as

$$\boldsymbol{\theta}^{(i)} \equiv \left[\theta_l^{(i)}, \theta_u^{(i)} \right] \tag{2-15}$$

in other literature. (Note: The notation [and] refer to inclusive end points where (and) refer to exclusive end points. Each may be combined as necessary such as a ≡ (b,c] to represent .b < a ≤ c.)

Understanding that c and k are not truly independent, one could also choose the damping ratio (ζ) and natural frequency (ω_n) as parameters. Equations (2-16) through (2-18) provide a summary of this alternative formulation.

$$\ddot{x} + 2\zeta\omega_n\dot{x} + \omega_n^2 x = \left(b/m\right)u + \left(g/m\right)w \tag{2-16}$$

$$c/m = 2\zeta\omega_n \tag{2-17}$$

$$k/m = \omega_n^2 \tag{2-18}$$

where (b/m) is also known as the static gain (G_S).

The general state-space model is provided as

$$\dot{\mathbf{x}} = \mathbf{A}(t)\mathbf{x} + \mathbf{B}(t)\mathbf{u} + \mathbf{G}(t)w \tag{2-19}$$

where **A** is the system matrix, **B** is the input matrix, and **G** is the process noise matrix. The process noise w is defined by its type, Gaussian (N), mean, μ_p (typically equal to zero) and variance Q.

$$w \sim N(\mu_p, Q) \tag{2-20}$$

Recall that equation (2-1) represents the general system measurement model. The exact values of the system parameters are not available. A measurement process is performed such as that shown in Figure 2-4 on a system such as the one illustrated in Figure 1-1. The plant model (equation (2-19)) is assumed to be of a known form but may have unknown parameters $\boldsymbol{\theta}$. The measurement model is provided as

$$\tilde{\mathbf{y}} = \mathbf{H}(t)\mathbf{x} + v \tag{2-21}$$

The measurement noise w is defined by its type (typically N for normal (Gaussian), mean μ_m (typically equal to zero) and variance R.

$$v \sim N(\mu_m, R) \tag{2-22}$$

where $\tilde{\mathbf{y}}$ is the measurement, **H** is the output matrix, and *v* is the measurement noise. Each is a vector. Equations (2-19) and (2-21) can be time-variant or time-invariant. Simulating the system simply involves using an ordinary differential equation (ODE) solver to integrate the set of first order differential equations. The ODE solver is typically accomplished by using a variable step 4th order Runge-Kutta method (RK4) such as ode23 or ode45 in MATLAB.

Through an appropriate transformation using matrix exponentials, the system may equivalently be represented in discrete-time equations

$$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{\Gamma}_k \mathbf{u}_k + \Upsilon_k \mathbf{w}_k, \quad \mathbf{w}_k \sim N\left(0, \mathbf{Q}(t)\right) \tag{2-23}$$

$$\tilde{\mathbf{y}}_{k+1} = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k, \quad \mathbf{v}_k \sim N\left(0, \mathbf{R}_k\right) \tag{2-24}$$

The discrete-time model is executed by simply looping through the equations. This process executes at a faster rate because it is a simple matrix multiplication rather than an ordinary differential equation solver.

2.3 The Kalman Filter

The Kalman filter developed in the 1950's provides the optimal linear filter for linear systems with zero-mean, uncorrelated and white Gaussian process and parameter noise.[49, 53, 95] Even if the noise is not Gaussian, the Kalman filter is still the optimal linear filter [49](pg 130). Figure 2-4 provides an overview of the typical application of the Kalman filter for measurements.

The Kalman filter provides an estimate to the system states by propagating the mean and covariance of the state through time [49](pg 123). The basic algorithm to the Kalman filter is presented in Table 2-2.

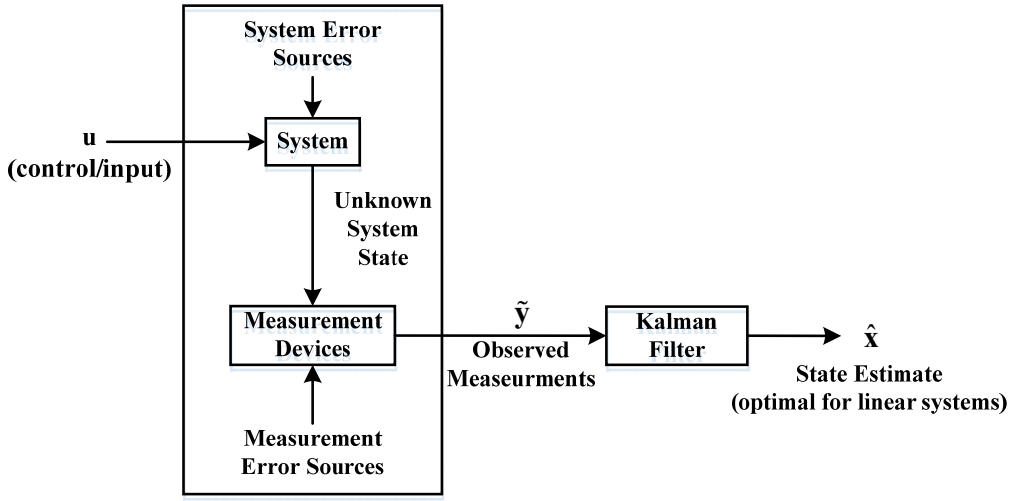


Figure 2-4 Typical Kalman filter application (adapted from [95])

Table 2-2 Kalman filter algorithm

1. Define the System Model
2. Initialize the state estimates and covariance.
3. Propagate the next state (predict)
4. Calculate the Kalman Gain
5. Update
    a. Perform the measurement update
    b. Update the error covariance
6. Repeat 2 through 4 until the measurements are depleted.

Many references provide a rigorous derivation of various types of Kalman filters that match the particular application. Some common tables defining the steps are extracted from [53] in Table 2-3 through Table 2-5.

Table 2-3 Continuous-time Kalman filter (adapted from [53])

Model	$\dot{\mathbf{x}}(t)=\mathbf{A}(t)\mathbf{x}(t)+\mathbf{B}(t)\mathbf{u}(t)+\mathbf{G}(t)w(t), \quad w(t)\sim N(0,Q(t))$ $\tilde{\mathbf{y}}(t)=\mathbf{H}(t)\mathbf{x}(t)+v(t), \quad v(t)\sim N(0,R(t))$
Initialize	$\hat{\mathbf{x}}_{\mathbf{0}}(t)_0=\hat{\mathbf{x}}_{\mathbf{0}}$ $\mathbf{P}_{\mathbf{0}}=E\left\{\hat{\mathbf{x}}_{\mathbf{0}}\hat{\mathbf{x}}_{\mathbf{0}}{}^T\right\}$
Kalman Gain	$\mathbf{K}(t)=\mathbf{P}(t)\mathbf{H}(t)\mathbf{R}^{-1}(t)$
Update:	
Estimate	$\dot{\hat{\mathbf{x}}}(t)=\mathbf{A}(t)\mathbf{x}(t)+\mathbf{B}(t)u(t)+\mathbf{K}(t)\left[\tilde{\mathbf{y}}(t)-\mathbf{H}(t)\hat{\mathbf{x}}(t)\right]$
Covariance	$\dot{\mathbf{P}}(t)=\mathbf{A}(t)\mathbf{P}(t)+\mathbf{P}(t)\mathbf{A}^T(t)-\mathbf{P}(t)\mathbf{H}^T(t)\mathbf{R}^{-1}(t)\mathbf{H}\mathbf{P}(t)$ $+\mathbf{G}(t)\mathbf{Q}(t)\mathbf{G}^T(t)$

Table 2-4 Continuous-discrete time Kalman filter (adapted from [53])

Model	$\dot{\mathbf{x}}(t)=\mathbf{F}(t)\mathbf{x}(t)+\mathbf{B}(t)\mathbf{u}(t)+\mathbf{G}(t)w(t), \quad w(t)\sim N(0,Q(t))$ $\tilde{\mathbf{y}}_k=\mathbf{H}(x_k)+\mathbf{v}_k, \quad \mathbf{v}_k\sim N(0,\mathbf{R}_k)$
Initialize	$\hat{\mathbf{x}}(t_0)=\hat{\mathbf{x}}_0$ $P_0=E\left\{\tilde{\mathbf{x}}(t_0)\tilde{\mathbf{x}}^T(t_0)\right\}$
Gain	$\mathbf{K}_k=\mathbf{P}_k^-\mathbf{H}_k^T\left[\mathbf{H}_k\mathbf{P}_k^-\mathbf{H}_k^T+\mathbf{R}_k\right]^{-1}$
Update	$\hat{\mathbf{x}}_k^+=\hat{\mathbf{x}}_k^-+\mathbf{K}_k\left[\tilde{\mathbf{y}}_k-\mathbf{H}_k\hat{\mathbf{x}}_k^-\right]$ $\mathbf{P}_k^+=\left[\mathbf{I}-\mathbf{K}_k\mathbf{H}_k\right]\mathbf{P}_k^-$
Propagation	$\dot{\mathbf{x}}(t)=\mathbf{F}(t)\mathbf{x}(t)+\mathbf{B}(t)\mathbf{u}(t)$ $\dot{\mathbf{P}}(t)=\mathbf{F}(t)\mathbf{P}(t)+\mathbf{P}(t)\mathbf{F}^T(t)+\mathbf{G}(t)\mathbf{Q}(t)\mathbf{G}^T(t)$

Table 2-5 Discrete-time Kalman filter (adapted from [53])

Model	$\mathbf{x}_{k+1} = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{\Gamma}_k \mathbf{u}_k + \mathbf{\Upsilon}_k \mathbf{w}_k, \quad \mathbf{w}_k \sim N\left(0, \mathbf{Q}(t)\right)$ $\tilde{\mathbf{y}}_{k+1} = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k, \quad \mathbf{v}_k \sim N\left(0, \mathbf{R}_k\right)$
Initialize	$\hat{\mathbf{x}}\left(t_0\right) = \hat{\mathbf{x}}_0$ $P_0 = E\left\{\tilde{\mathbf{x}}\left(t_0\right)\tilde{\mathbf{x}}^T\left(t_0\right)\right\}$
Gain	$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T \left[\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{R}_k\right]^{-1}$
Update	$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left[\tilde{\mathbf{y}}_k - \mathbf{H}_k \hat{\mathbf{x}}_k^-\right]$ $\mathbf{P}_k^+ = \left[\mathbf{I} - \mathbf{K}_k \mathbf{H}_k\right]\mathbf{P}_k^-$
Propagation	$\mathbf{x}_{k+1}^- = \mathbf{\Phi}_k \mathbf{x}_k + \mathbf{\Gamma}_k \mathbf{u}_k$ $\mathbf{P}_{k+1}^- = \mathbf{\Phi}_k \mathbf{P}_k^+ \mathbf{\Phi}_k^T \mathbf{x}_k + \Upsilon_k \mathbf{Q}_k \Upsilon_k^T$

The Kalman filter will be the basis of the filter models for linear systems when considering Multiple-Model Adaptive Estimation. The Kalman filter is capable of producing accurate estimates of the true states of a physical system but it limited by the system model.[47, 51, 52] When the parameters of the system model are uncertain, alternative techniques must be performed. One such method is to add the parameters to the state model using an Extended Kalman Filter on the resultant nonlinear system. Crassidis and Junkins provide an example using the EKF to estimate model parameters in Example 3.6.[53](pgs. 190-192)

The extended Kalman filter (EKF) linearizes the system about the current state. Jacobians are used to update the measurement matrix, **H**, and the system matrix, **F**. (**F** is a linearized equivalent to **A** in equation (2-19).) The EKF version used in this research is the continuous-discrete EKF provide in Table 2-6. The continuous-discrete version takes advantage or the simplicity of discrete gain and

updates with a continuous model running with an ODE solver. The algorithm follows the basic Kalman filter steps of Table 2-2.

Table 2-6 Continuous-discrete extended Kalman filter (adapted from [53])

Model	$\dot{\mathbf{x}}(t) = \mathbf{f}\left(x(t), u(t), t\right) + \mathbf{G}(t)\mathbf{w}(t), \quad \mathbf{w}(t) \sim N\left(0, \mathbf{Q}(t)\right)$ $\tilde{\mathbf{y}}_k = \mathbf{h}\left(x_k\right) + \mathbf{v}_k, \quad \mathbf{v}_k \sim N\left(0, \mathbf{R}_k\right)$
Initialize	$\hat{\mathbf{x}}(t_0) = \hat{\mathbf{x}}_0$ $P_0 = E\left\{\tilde{\mathbf{x}}(t_0)\tilde{\mathbf{x}}^T(t_0)\right\}$
Gain	$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T\left(\hat{\mathbf{x}}_k^-\right)\left[\mathbf{H}_k\left(\hat{\mathbf{x}}_k^-\right)\mathbf{P}_k^- \mathbf{H}_k^T\left(\hat{\mathbf{x}}_k^-\right) + \mathbf{R}_k\right]^{-1}$ $\mathbf{H}_k\left(\hat{\mathbf{x}}_k^-\right) \equiv \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right\|_{\hat{\mathbf{x}}_k^-}$
Update	$\hat{\mathbf{x}}_k^+ = \hat{\mathbf{x}}_k^- + \mathbf{K}_k\left[\tilde{\mathbf{y}}_k - \mathbf{h}_k\left(\hat{\mathbf{x}}_k^-\right)\right]$ $\mathbf{P}_k^+ = \left[\mathbf{I} - \mathbf{K}_k \mathbf{H}_k\left(\hat{\mathbf{x}}_k^-\right)\right]\mathbf{P}_k^-$
Propagation	$\dot{\mathbf{x}}(t) = \mathbf{f}\left(x(t), u(t), t\right) + \mathbf{G}(t)\mathbf{w}(t), \quad \mathbf{w}(t) \sim N\left(0, \mathbf{Q}(t)\right)$ $\dot{\mathbf{P}}(t) = \mathbf{F}(t)\mathbf{P}(t) + \mathbf{P}(t)\mathbf{F}^T(t) + \mathbf{G}(t)\mathbf{Q}(t)\mathbf{G}^T(t)$ $\mathbf{H}_k\left(\hat{\mathbf{x}}_k^-\right) \equiv \left.\frac{\partial \mathbf{h}}{\partial \mathbf{x}}\right\|_{\hat{\mathbf{x}}_k^-}$

2.4 Extended Kalman Filter for Parameter Estimation

As previously mentioned in section 1.6.3, parameter estimation requires additional states to be augmented to the system model in equation (2-19). The result is a nonlinear system.

For example, using the system equation (2-12) with a measurement matrix of **H** = [1 0], the state-space representation is

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u$$
$$y = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + v \tag{2-25}$$

Augmenting the parameters to the state-space representation expands the model to

$$\dot{\mathbf{x}} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{z}_1 \\ \dot{z}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -z_1 x_2 - z_2 x_1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} w \tag{2-26}$$
$$\tilde{y} = x_1 + v$$

This system requires a nonlinear filter. The EKF is chosen with **F(**x**)** and **H**(x) for the EKF defined as

$$\mathbf{F}(x) = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} x_2 \\ -z_1 x_2 - z_2 x_1 \\ 0 \\ 0 \end{bmatrix} \tag{2-27}$$
$$\mathbf{H}(x) = \begin{bmatrix} x_1 & 0 \end{bmatrix}$$

Ultimately, the EKF can be used in some conditions with well-known initial conditions. **F** and **H** must be linearized at each time step. There is typically a single measurement such as the position. Adding parameters to the EKF increase the order of the system without increasing the measurements. The EKF model becomes less and less observable and may not converge. Therefore, an alternate approach is desired. The approach chosen is MMAE.

2.5 Multiple-Model Adaptive Estimation Framework

Multiple-Model Adaptive Estimation, illustrated in Figure 2-5, is a method that uses a parallel bank of Kalman filters to generate estimates of state for each filter model. Each filter is composed of a hypothesis model representing a guess of the system model within the defined parameter space of possible values for the unknown parameters. The parameters can be coefficients of differential equations as well as process or measurement noise covariance values. The resultant combined state and parameter estimate is provided through the weighted sum of the filter bank elements. There are several methods to perform the hypothesis testing. Magill used minimum mean square error approach.[51] Maybeck used maximum a posteriori estimation (MAP) to select the highest probability filter out of his moving-bank [74]. For this research, the estimated weight is determined from the maximum likelihood estimation (MLE) following Crassidis and Junkins section 4.6.2. of [53]. This likelihood function provides the hypothesis of correctness for each filter. MMAE is recursive and may be executed in real time.[53](pg 249) Furthermore, MMAE itself is ideally suited for distributed computing due to its parallel filter banks.[74]

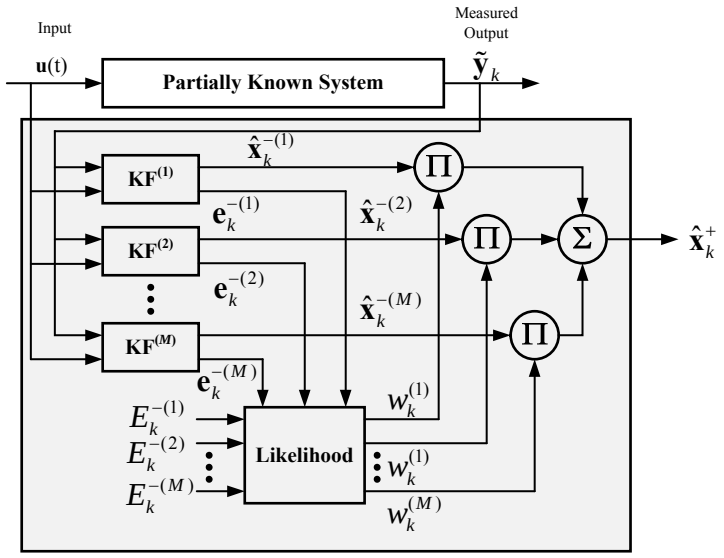


Figure 2-5 Multiple-model adaptive estimation (adapted from [53] and [57])

Multiple-Model Adaptive Estimation (MMAE) was introduced by D. T. Magill in 1965. The original paper, [51], describes the method as "an adaptive approach to the problem of estimating a sampled, stochastic process described by an initially unknown parameter vector." Magill's approach was different from prior approaches because the resulting system is optimal by minimizing the quadratic expectation of the generalized mean-square-error performance criterion.[51] Magill's approach provided a weighting algorithm that was expected to converge weighting coefficients to point to the true model out of a bank of estimators.[51] Aguiar provides an overview of MMAE circa 2007 in [47]. Aguiar describes MMAE as a type of adaptive observer. An adaptive observer is a process that provides real-time state estimate of the plant from partial and possibly noisy measurements of the inputs and outputs.[47] All or part of the initial conditions,

parameters of the plant and the noise may not be fully known.[47, 51] Without uncertainty of the plant parameters, the Kalman filter itself is the optimum linear filter, and there is no need for an adaptive estimator.[47, 59]

The motivation for MMAE methods was the desire for accurate stochastic state estimation with systems containing significant parameter uncertainty.[47] In other words, simultaneous estimation of both the states and parameters. MMAE has found applications in a variety of fields including the following domains: surveillance and fusion algorithms involving multiple sensors and multiple targets, adaptive control, fault detection and isolation, biomedical engineering and other systems requiring multiple hypothesis testing.[47]

2.5.1 MMAE Derivation

The development of the MMAE filter in this document follows Crassidis and Junkins section 4.6.2. of [53] which gives credit to [59] and [60]. The bank of *M* filters depends on the unknown parameters represented by the vector $\boldsymbol{\theta}$. The set of parameters, $\boldsymbol{\theta}$, can be constant or time varying during the time of adaptation (or convergence) of the MMAE. Elements representing the *M* filters are defined from a known probability density function (pdf) of $\boldsymbol{\theta}$,. These probabilities are represented by $\mathbf{p}(\boldsymbol{\theta})$ to provide $\{\theta^{(i)}; \mathrm{i} = 1, \ldots, \mathrm{M}\}$.

The ultimate goal of MMAE is to determine the conditional pdf of the i^{th} element of $\mathbf{p}^{(i)}$ from the measurements. Recall, to calculate the probability that event *a* occurs given *b* has already occurred is *p(a|b)*. Bayes' rule is derived from

the notion that this probability is proportion to $p(a \cap b)$. A proportionality constant of $1/p(b)$ is required to ensure the total probabilities of all possible events, which a and b are two such events, sum up to 1. Therefore, probability of a given b is expressed as

$$p(a \mid b) = \frac{p(a \cap b)}{p(b)} \tag{2-28}$$

Similarly, the probability of b given a is

$$p(b \mid a) = \frac{p(b,a)}{p(a)} = \frac{p(b \cap a)}{p(a)} \tag{2-29}$$

Combining both equations yields Bayes' rule

$$p(a \mid b) = \frac{p(b \mid a)\, p(a)}{p(b)} \tag{2-30}$$

The conditional pdf of the i^{th} element of $\theta(i)$ is derived by Bayes' rule to give a recursive form

$$p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_k\right) = \frac{p\left(\tilde{\mathbf{Y}}_k \mid \mathbf{p}^{(i)}\right) p\left(\mathbf{p}^{(i)}\right)}{p\left(\tilde{\mathbf{Y}}_k\right)} = \frac{p\left(\tilde{\mathbf{Y}}_k \mid \mathbf{p}^{(i)}\right) p\left(\mathbf{p}^{(i)}\right)}{\sum_{i=1}^{M} p\left(\tilde{\mathbf{Y}}_k \mid \mathbf{p}^{(i)}\right) p\left(\mathbf{p}^{(i)}\right)} \tag{2-31}$$

where $\tilde{\mathbf{Y}}$ is the sequence $\left\{\tilde{\mathbf{y}}_0, \tilde{\mathbf{y}}_1, ..., \tilde{\mathbf{y}}_k\right\}$. The desire is to develop an update law that is only dependent on the current measurement $\tilde{\mathbf{y}}_k$. The conditional probability of equation (2-29) and Bayes' rule (equation (2-30)) are used to determine

$$p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_k\right) = \frac{p\left(\tilde{\mathbf{y}}_k, \tilde{\mathbf{Y}}_{k-1}, \mathbf{p}^{(i)}\right)}{p\left(\tilde{\mathbf{y}}_k, \tilde{\mathbf{Y}}_{k-1}\right)} \tag{2-32}$$

$$p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_k\right) = \frac{p\left(\tilde{\mathbf{y}}_k, \mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_{k-1}\right) p\left(\tilde{\mathbf{Y}}_{k-1}\right)}{p\left(\tilde{\mathbf{y}}_k \mid \tilde{\mathbf{Y}}_{k-1}\right) p\left(\tilde{\mathbf{Y}}_{k-1}\right)} \tag{2-33}$$

$$p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_k\right) = \frac{p\left(\tilde{\mathbf{y}}_k, \mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_{k-1}\right)}{p\left(\tilde{\mathbf{y}}_k \mid \tilde{\mathbf{Y}}_{k-1}\right)} \tag{2-34}$$

$$p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_k\right) = \frac{p\left(\tilde{\mathbf{y}}_k \mid \tilde{\mathbf{Y}}_{k-1}, \mathbf{p}^{(i)}\right) p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_{k-1}\right)}{\sum_{i=1}^{M} p\left(\tilde{\mathbf{y}}_k \mid \tilde{\mathbf{Y}}_{k-1}, \mathbf{p}^{(i)}\right) p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_{k-1}\right)} \tag{2-35}$$

For each $\boldsymbol{p}^{(i)}$, the set of state estimates $\hat{\mathbf{x}}_k^-\left(\mathbf{p}^{(i)}\right) \equiv \hat{\mathbf{x}}_k^{-(i)}$ is provided from the bank of filters. Then, $p\left(\tilde{\mathbf{y}}_k \mid \tilde{\mathbf{Y}}_{k-1}, \mathbf{p}^{(i)}\right)$ is provided by $p\left(\tilde{\mathbf{y}}_k \mid \hat{\mathbf{x}}_k^{-(i)}\right)$ because $\hat{\mathbf{x}}_k^{-(i)}$ is a function of $\boldsymbol{p}^{(i)}$ and it uses all of the measurements up through k-1. Equation (2-35) becomes

$$p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_k\right) = \frac{p\left(\tilde{\mathbf{y}}_k \mid \hat{\mathbf{x}}_k^{-(i)}\right) p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_{k-1}\right)}{\sum_{i=1}^{M} p\left(\tilde{\mathbf{y}}_k \mid \hat{\mathbf{x}}_k^{-(i)}\right) p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_{k-1}\right)} \tag{2-36}$$

where the denominator is the normalizing factor to ensure $p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_k\right)$ is a pdf. Defining the weights as $w_k^{(i)} = p\left(\mathbf{p}^{(i)} \mid \tilde{\mathbf{Y}}_k\right)$ means equation (2-35) can be used for the weight equation below.

$$w_k^{(i)} = w_{k-1}^{(i)} p\left(\tilde{\mathbf{y}}_k \mid \hat{\mathbf{x}}_k^{-(i)}\right) \tag{2-37}$$

The weights are normalized, to sum up to 1 by dividing each by the total weight.

$$w_k^{(i)} \leftarrow \frac{w_k^{(i)}}{\sum_{i=1}^{M} w_k^{(i)}} \tag{2-38}$$

Notice that only the current time measurement $\tilde{\mathbf{y}}_k$ is required.

The pdf $p\left(\tilde{\mathbf{y}}_k \mid \hat{\mathbf{x}}_k^{-(i)}\right)$ is computed from the measurement residual

$$\mathbf{e}_k^{-(i)} \equiv \tilde{\mathbf{y}}_k - \tilde{\mathbf{y}}_k^{-(i)} \tag{2-39}$$

The covariance of $\mathbf{e}_k^{-(i)}$ is

$$\mathbf{E}_k^{-(i)} \equiv E\left\{\mathbf{e}_k^{-(i)}\mathbf{e}_k^{-(i)T}\right\} = H_k^{(i)} P_k^{-(i)} H_k^{(i)T} + R_k^{(i)} \tag{2-40}$$

where $P_k^{-(i)}$ is the covariance from the i^{th} Kalman filter.

Notice that there is no assumption that the output is a linear function. Therefore, the nonlinear measurement relation

$$\tilde{\mathbf{y}}_k^{-(i)} = \mathbf{h}\left(\hat{\mathbf{x}}_k^{-(i)}, k\right) \tag{2-41}$$

is applicable in for residual calculation of equation (2-39), becoming

$$\mathbf{e}_k^{-(i)} \equiv \tilde{\mathbf{y}}_k - \mathbf{h}\left(\hat{\mathbf{x}}_k^{-(i)}, k\right) \tag{2-42}$$

$H_k^{(i)}$ can be taken directly from the EKF, becoming

$$H_k^{(i)} = H_k^{(i)}\left(\hat{\mathbf{x}}_k^{-(i)}\right) \tag{2-43}$$

$H_k^{(i)}\left(\hat{\mathbf{x}}_k^{-(i)}\right)$ is applicable for the residual covariance of equation (2-40), becoming

$$\mathbf{E}_k^{-(i)} \equiv E\left\{\mathbf{e}_k^{-(i)}\mathbf{e}_k^{-(i)T}\right\} = H_k^{(i)}\left(\hat{\mathbf{x}}_k^{-(i)}\right) P_k^{-(i)} \left(H_k^{(i)}\left(\hat{\mathbf{x}}_k^{-(i)}\right)\right)^T + R_k^{(i)} \tag{2-44}$$

Equations (2-41) through (2-44) show that the MMAE algorithm can be easily extended to use EKF models. The pdf $p\left(\tilde{\mathbf{y}}_k \mid \hat{\mathbf{x}}_k^{-(i)}\right)$ is given by

$$p\left(\tilde{\mathbf{y}}_k \mid \mathbf{x}_k^{-(i)}\right)^{(i)} = \frac{1}{\left[\det\left(2\pi\mathbf{E}_k^{-(i)}\right)\right]^{1/2}} \exp\left\{-\frac{1}{2}\mathbf{e}_k^{-(i)T}\left(\mathbf{E}_k^{-(i)}\right)^{-1}\mathbf{e}_k^{-(i)}\right\} \tag{2-45}$$

The weighted sum of the parallel filter estimates is provided by the conditional mean estimate

$$\hat{\mathbf{x}}_k^{+} = \sum_{i=1}^{M} w_k^{(i)} \hat{\mathbf{x}}_k^{+(i)} \tag{2-46}$$

The covariance of the state estimate is computed using

$$\mathbf{P}_k^{+} = \sum_{i=1}^{M} w_k^{(i)} \left[\left(\hat{\mathbf{x}}_k^{+(i)} - \hat{\mathbf{x}}_k^{+}\right)\left(\hat{\mathbf{x}}_k^{+(i)} - \hat{\mathbf{x}}_k^{+}\right)^T + \mathbf{P}_k^{+(i)}\right] \tag{2-47}$$

The parameter estimate $\hat{\boldsymbol{\theta}}_k$ at time t_k is computed using

$$\hat{\boldsymbol{\theta}}_k = \sum_{i=1}^{M} w_k^{(i)} \hat{\boldsymbol{\theta}}^{(i)} \tag{2-48}$$

The error covariance is computed using

$$\mathbf{P}_{\boldsymbol{\theta}k} = \sum_{i=1}^{M} w_k^{(i)} \left(\hat{\boldsymbol{\theta}}^{(i)} - \hat{\boldsymbol{\theta}}_k\right)\left(\hat{\boldsymbol{\theta}}^{(i)} - \hat{\boldsymbol{\theta}}_k\right)^T \tag{2-49}$$

Equation (2-49) can be used to define and evaluate 3σ boundaries of the estimate $\hat{\boldsymbol{\theta}}_k$. When the number of models M is large and the models cover significant regions of the parameter space $\boldsymbol{\theta}$ using $\boldsymbol{\theta}^{(i)}$, the resultant parameter estimate of

equations (2-48) and (2-49) provide a good approximation of the conditional mean of θ.

With [52], Miller provides a detailed summary of the assumptions behind the basic concepts of MMAE from [78, 96, 97]. Table 2-7 below provides those assumptions paraphrased with the references from this document.

Table 2-7 Miller's summary of MMAE basic assumptions (adapted from [52])

- Sampled- data system can be represented by linear Kalman filters resulting in Gaussian probability density functions [78, 96]
- The extended Kalman filters are typically used for nonlinear systems and the probability density functions are approximated as Gaussian
- The values to be estimated can be the uncertain parameters and the statistical properties of the noise
- Parameter estimates vary over a continuous range of the parameter space. The range must be stratified in some manner. The choice of discrete values affect the resulting MMAE filters and resulting estimate. Reference [87] provides a method for choosing the discrete values.

2.5.2 Summary of the MMAE Algorithm

The basic MMAE algorithm as presented by Crassidis and Junkins in section 4.6.2. of [53] is provided in Figure 2-5. The steps are detailed in order in Table 2-8. Steps 1 and 2 are only performed during initialization. Steps 3 through 9 are performed for each data point available.

Table 2-8 Summary of detailed steps of discrete-time MMAE of [53]

1. Generate models $\left(\boldsymbol{\Phi}^{(i)}\right)$ to represent parameter sets of interest $\left(\boldsymbol{\theta}^{(i)} \in \boldsymbol{\theta}\right)$

$$\mathbf{x}_{k+1}^{(i)} = \boldsymbol{\Phi}^{(i)}\mathbf{x}_k^{(i)} + \boldsymbol{\Gamma}_k u_k + \boldsymbol{\Upsilon}_k w_k; \quad w_k \sim N\left(0, Q_k^{(i)}\right)$$

$$\tilde{\mathbf{y}}_k = \mathbf{H}_k\mathbf{x}_k^{(i)} + v_k; \quad v_k \sim N\left(0, R_k^{(i)}\right)$$

2. Initialize, initial values for:
   a. MMAE state estimate $\left(\hat{\mathbf{x}}_\mathbf{0}\right)$
   b. MMAE parameter estimates $\left(\hat{\boldsymbol{\theta}}_\mathbf{0}\right)$
   c. Parameter estimate covariances $\left(\mathbf{P_0} = E\left\{\hat{\mathbf{x}}_\mathbf{0}\hat{\mathbf{x}}_\mathbf{0}^{\,T}\right\}\right)$
   d. KF model state estimates $\left(\hat{\mathbf{x}}_0^{(i)}\right)$
   e. KF model covariances $\left(\mathbf{P}_0^{(i)} = E\left\{\hat{\mathbf{x}}_0^{(i)}\hat{\mathbf{x}}_0^{(i)T}\right\}\right)$
   f. Initialize filter weights $w_0^{(i)} = 1/M$
3. For each model:
   a. Propagate the next state (predict)

$$\hat{\mathbf{x}}_{k+1}^{-(i)} = \boldsymbol{\Phi}^{(i)}\mathbf{x}_k^{+(i)} + \boldsymbol{\Gamma}_k^{(i)}u_k$$

$$\mathbf{P}_{k+1}^{-(i)} = \boldsymbol{\Phi}^{(i)}\mathbf{P}_k^{+(i)}\boldsymbol{\Phi}^{(i)T} + \boldsymbol{\Upsilon}_k^{(i)}Q_k\boldsymbol{\Upsilon}_k^{(i)T}$$

   b. Calculate the Kalman Gain

$$\mathbf{K}_k^{(i)} = \mathbf{P}_k^{-(i)}\mathbf{H}_k^T\left[\mathbf{H}_k\mathbf{P}_k^{-(i)}\mathbf{H}_k^T + \mathbf{R}_k\right]^{-1}$$

   c. Update
      i. Perform the measurement update

$$\mathbf{x}_k^{+(i)} = \mathbf{x}_k^{-(i)} + \mathbf{K}_k^{-(i)}\left[\tilde{\mathbf{y}}_k^{(i)} - \mathbf{H}_k\hat{\mathbf{x}}_k^{(i)}\right]$$

      ii. Update the error covariance

   d. $\mathbf{P}_k^{+(i)} = \left[\mathbf{I} - \mathbf{K}_k^{(i)}\mathbf{H}_k\right]\mathbf{P}_k^{-(i)}$
   e. Calculate the posterior pdf (likelihood)

$$p\left(\tilde{\mathbf{y}}_k \mid \mathbf{x}_k^{-(i)}\right)^{(i)} = \frac{1}{\left[\det\left(2\pi\mathbf{E}_k^{-(i)}\right)\right]^{1/2}}\exp\left\{-\frac{1}{2}\mathbf{e}_k^{-(i)T}\left(\mathbf{E}_k^{-(i)}\right)^{-1}\mathbf{e}_k^{-(i)}\right\}$$

   where

   $e_k^{-(i)}$ is the measurement residual and measurement covariance is

$$\mathbf{E}_k^{-(i)} \equiv E\left\{\mathbf{e}_k^{-(i)}\mathbf{e}_k^{-(i)T}\right\} = H_k^{(i)}P_k^{-(i)}H_k^{(i)T} + R_k^{(i)}$$

Table 2-8 Summary of detailed steps of discrete-time MMAE (continued)

4. Update the weights

$$w_k^{(i)} = w_{k-1}^{(i)} p\left(\tilde{\mathbf{y}}_k \mid \mathbf{x}_k^{-(i)}\right)^{(i)}$$

5. Normalize the weights

$$w_k^{(i)} \leftarrow \frac{w_k^{(i)}}{\sum_{i=1}^{M} w_k^{(i)}}$$

6. Calculate MMAE State Estimate

$$\hat{\mathbf{x}}_k^{+} = \sum_{i=1}^{M} w_k^{(i)} \hat{\mathbf{x}}_k^{+(i)}$$

7. Calculate the MMAE State Error Covariances

$$\mathbf{P}_k^{+} = \sum_{i=1}^{M} w_k^{(i)} \left[\left(\hat{\mathbf{x}}_k^{+(i)} - \hat{\mathbf{x}}_k^{+} \right) \left(\hat{\mathbf{x}}_k^{+(i)} - \hat{\mathbf{x}}_k^{+} \right)^T + \mathbf{P}_k^{+(i)} \right]$$

8. Repeat 3 through 9 until the measurements are depleted

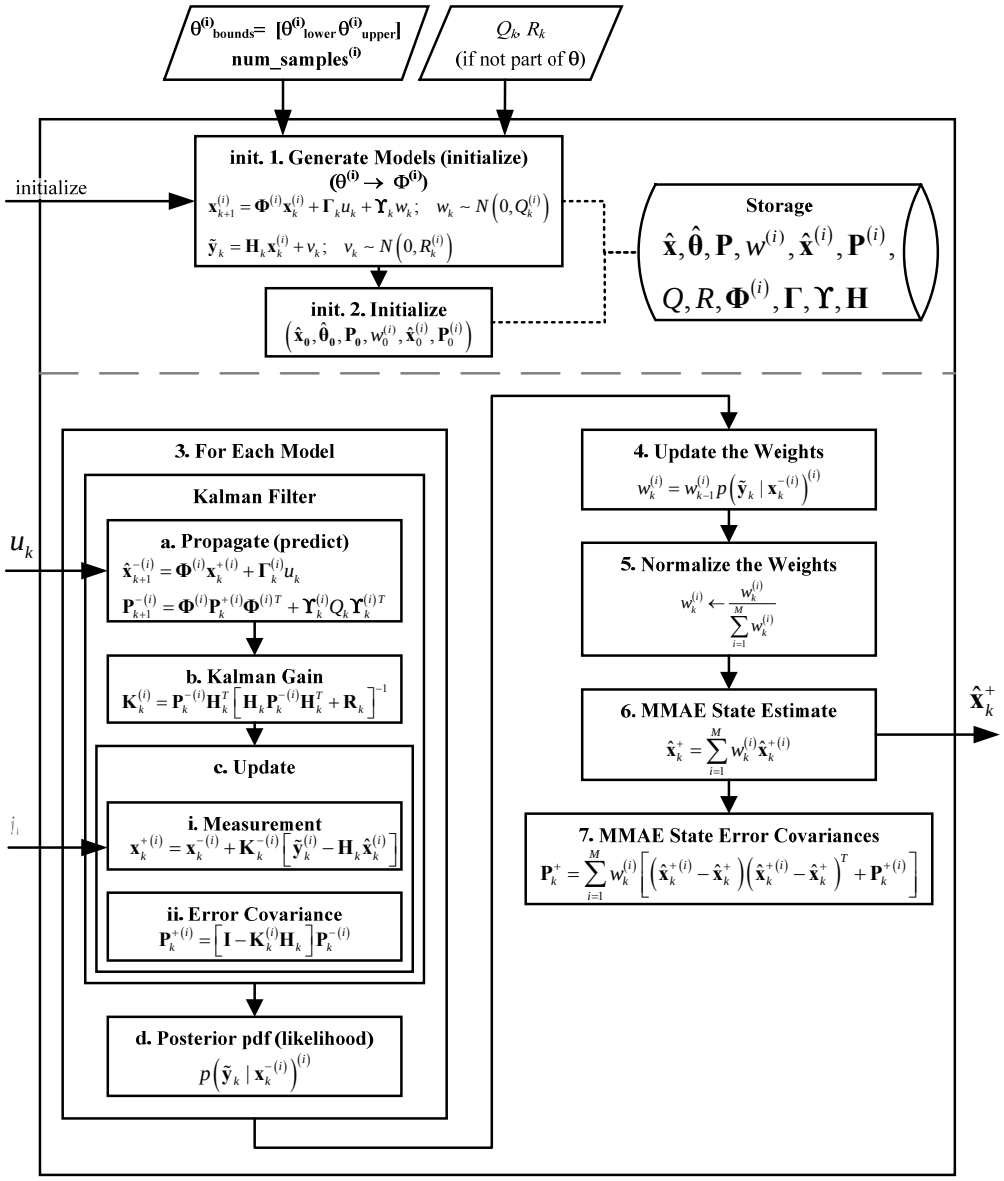


Figure 2-6 Flow of MMAE steps in [53]

CHAPTER 3 Choosing the Hypothesis Models

The focus of this chapter is to cover the methods used to choose the hypothesis models of GRAPE. Two methods to define the hypothesis models are present. The first is the selected grid-based stratification (SGBS) and the second is Latin Hypercube Sampling (LHS) used. The GRAPE framework is formalized using each of these methods later in Chapter 4. This chapter covers a more rigorous definition of the parameter space in respect to MMAE and FDD. The topics of model location and sample size are also discussed. SGBS is then defined as a method to stratify the sample space. The chapter then covers statistical sampling techniques used to minimize the model count. LHS, a widely used sampling technique, is presented as the method to appropriately space the model samples while also minimizing the model count.

3.1 FDD Parameter Space

There are three main concerns when choosing the hypothesis models in the FDD framework. The first concern is defining the parameter space from the application of interest. For the idealized equation (2-8), the parameter space is the upper and lower bounds of the coefficients c and k. The parameter space can be further evaluated to include possible the noise properties of Q and R. The topic of parameter space was introduced in section 1.4.2 and is formally defined in section 3.2. The second concern is the location of models (parameters sets) within the parameter space. The models can be spread evenly over the bounds of the particular

parameter, or they could be selected according to a probability distribution. The final concern is the sample size. The number of filter models affects the accuracy of both the parameter and state estimates.[65](pg 770) The concepts of model location and sample size are discussed in section 3.3.

In the original approach, Magil constructs an optimal estimator by assuming the "parameter vector is unknown but is selected from a finite set of known vectors."[51](pg. 435,439) When the true model is part of the hypothesis model set, the weight for the true model can be expected to converge to one with all the remaining zero.[51](pg. 435,437) For the approach of this research, the true model is known and used to generate synthetic measurements. However, the true model is not necessarily a specific value within the discretized parameter space. Therefore, the model that generates the highest normalized weight is considered closest to the true model. Typically, one model will generate the highest normalized weight with the vast majority of normalized weights near zero. Grid-based sampling technique may produce several models that are close to the true model. In this case, there is one model with normalized weight near one and the other close models with much smaller normalized weights followed by models with effective weights near zero.

3.2 Parameter Space for MMAE

This section builds on the concepts of parameter space introduced in section 1.4.2 to fully define the parameter space of the system of interest. The parameter

space is defined as the set of possible values for each parameter. Each parameter defines another dimension, *d*, in the space. Therefore, each parameter is selected from a parameter space defined below.

$$\boldsymbol{\theta} \in \Re^{d\times 1} \text{ where } \boldsymbol{\theta}^{(i)}_{bounds} \in \left[\theta_l^{(i)} \quad \theta_u^{(i)}\right] \quad \text{for } i = 1,\ldots,d \tag{3-1}$$

The parameter set, or sample set, defines the current set of parameters that define equation. This is defined below.

$$\boldsymbol{\theta}^{(i)} = \left[\theta_1^{(i)} \quad \theta_2^{(i)} \quad \ldots \quad \theta_{ni}^{(i)}\right] \quad \text{for } i = 1,\ldots,d \tag{3-2}$$

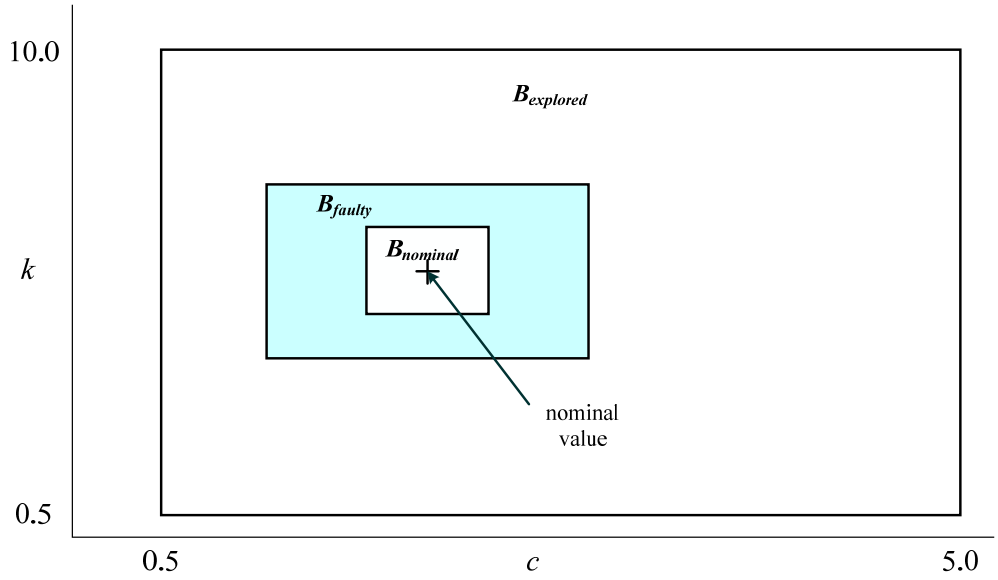


Figure 3-1 2-dimensional parameter space behavior definition example

From equation (2-14) the each parameter dimension is represented by the parameter, θ_i, a lower bound θ_{il}, and an upper bound θ_{iu}. When an expected value of the parameter is known, that parameter dimension may also be defined by the mean and variance.

$$\boldsymbol{\theta}_i \equiv \left\{ \boldsymbol{\theta}_i \in \Re^{nx1} \mid \boldsymbol{\theta}_i \sim N\left(\mu_i, \sigma^2{}_i\right) \right\} \tag{3-3}$$

For purposes of analysis, the MMAE method allows the mean and variance characteristics of the process noise (μ_p,Q) and variance noise (μ_m,R) also to be included in the parameter space. Using these noise characteristics can add up to four parameters (μ_p,Q,μ_m,R). Typically, the mean is assumed to be zero.

3.3 Model Location and Sample Size

There are a variety of techniques to define the Kalman filter models used in MMAE. These models are hereby referred to as model samples, or simply samples. They are chosen from the parameter space. The sample, $\theta^{(i)}$, is the parameter set of interest. It will include the values of all parameters of interest, $\theta^{(i)} = [\theta^{(i)}_1, \theta^{(i)}_2, \ldots, \theta^{(i)}_{n-1} \ldots, \theta^{(i)}_n]$, where n is the number of parameters, i is the current sample and l is the current parameter.

For this research effort, the grid-based method using equally spaced parameter values to span each respective parameter dimension is called selected grid-based stratification (SGBS). Selected in the name SGBS indicates that it is not a random value. Selected means the model is generated a at grid depending on

the number of models selected and the current range of the parameter space. This method is similar to the original MMAE of Magil [51], Crassidis and associated researchers Alsuwaidan, Cheng, García and Martí ([61-63]), along with Maybeck's moving-bank MMAE (MBMMAE) in [78]. Vasquez has modified the moving-bank approach to generate samples models using the chi-squared distribution.[48, 65] However, there is one key difference. The models are not predefined. An initial set is predefined on the original parameter ranges and the number of model divisions per parameter. Each subsequent resample reinitializes based off of the number of model divisions per parameter, the latest parameter estimate and the updated parameter ranges. The parameter ranges follow a process to either contract to a parameter estimate or expand to track parameter estimates on the periphery or outside the current bounds. (Specifics on the final resampling behavior and framework can be found in section 5.2.)

An alternate sampling approach introduced by this research is to sample the parameter space using Latin Hypercube Sampling (LHS) techniques. LHS is a variance reduction technique used to minimize the number of models (samples) for Monte Carlo applications. The following two sections present SGBS and LHS techniques in detail.

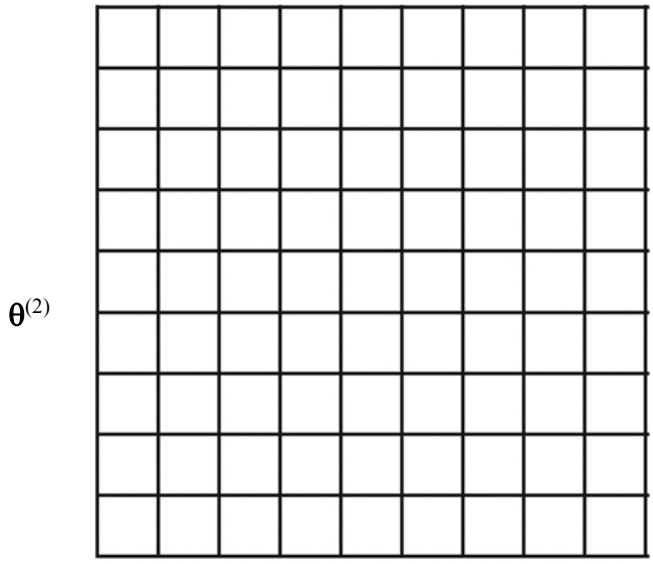
3.4 Selected Grid-Based Stratification (SGBS)

The preliminary focus of this research is SGBS. This approach gives accurate values with the cost of computing a higher number of filters. SGBS simply

divides each parameter θ into equally spaced strata based on the accuracy and computational complexity criteria. The start and end points of each parameter dimension are defined by the region of interest following equation (3-1).

For this research, the primary region of interest is the parameter space of acceptable faulty operation, see Figure 3-1. The system is assumed to be designed with fault tolerance within this region. The faultless operation is a subset of the faulty operation parameter space. Outside of the faulty operation region is considered a system failure.

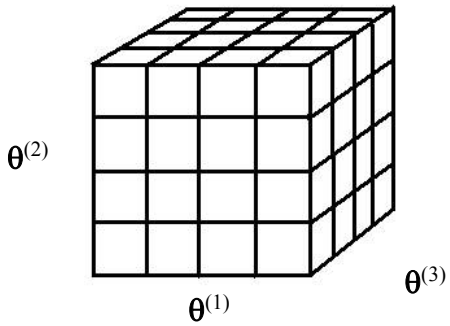
Parameter divisions or strata are determined by the desired precision. The number of models, *m*, affects the accuracy of the MMAE estimation. [48](pg 770) MMAE weights will converge primarily to the model with the minimum residual error. Therefore, the closer the models are spaced, the closer the converged MMAE model will be to the true system parameter. [52, 53] Figure 3-2 illustrates a grid-based stratification effort generating 100 models.



10x10 = 100 models, 1 at each intersection

Figure 3-2 2-Dimensional parameter space example

Expanding the notion to a three-dimensional grid is illustrated in Figure 3-3. Here, the process noise has added another parameter to the parameter space.



5x5x5 = 125 models,
1 at each intersection

Figure 3-3 3-dimensional parameter space example

The number of samples, or filter models, is determined by the product of the number of parameter divisions or strata. In general, the number of samples, *Sn* is

$$S_n = (strata\ \theta_1)(strata\ \theta_2)\cdots(strata\ \theta_d) \tag{3-4}$$

where *n* is the number of parameters and i is the individual parameter of interest. If the number of strata per parameter are equal, then equation (3-4) becomes a power relationship shown below.

$$S_n = (strata\ )^d \tag{3-5}$$

Equations (3-4) and (3-5), show that adding another parameter multiplies the number of models by the new strata of that parameter. This adversely affects the computational complexity by requiring more resources to implement additional models. Unless the strata and number of parameters are relatively small, an alternate sampling technique is desired.

The sample size of SGBS is chosen from two competing criteria, computational complexity and desired accuracy of the results. A Higher number of models requires more computational effort to perform each update and propagation of the Kalman filter. In addition, they also require more memory to store the data. Therefore, the desire is to have the least number of models as possible. However, minimizing the number of models affects the accuracy of the MMAE estimation. [48](pg 770) MMAE weights will converge primarily to the

model with the minimum residual error. Therefore, the closer spaced the models are, the closer the converged MMAE model converge to the true system model**.** In practice, the starting number of models per parameter is ten. The simulation is executed. Then a decision to increase or decrease the number of models is made based on both the time to execute the GRAPE framework and the final desired accuracy of the resultant estimates. Since the system is a simulation, the true values are known and available for comparison. For implementation on hardware, a more detailed evaluation of the cost of computational complexity and real-time execution capabilities must be compared with the desired accuracy.

3.5 Sampling Techniques

Random sampling techniques are presented in this section. The sampling method is compared with the Monte Carlo method. The independent and independently distributed (IID) sampling technique is then presented to be compared with the chosen Latin Hypercube Sampling method. The number of samples is addressed by evaluating the confidence of LHS based off IID which typically bounds LHS. Finally, the reasons for choosing LHS as a sampling method are summarized.

3.5.1 Evaluating MMAE as a type of Monte Carlo Method

The variance reduction technique known as Latin Hypercube Sampling is used to minimize the number of models (samples) for MMAE. This section presents the models (samples) analogous to random samples for the Monte Carlo

method. A brief history of the Monte Carlo method is presented followed by comparing the set of MMAE models, when the MMAE weighting method has converged, to a Monte Carlo simulation.

Monte Carlo (MC) methods are a broadly defined set of computational algorithms that repeatedly sample parameters to obtain some numerical result. They are used to evaluate physical or mathematical problems that are deterministic in principle but difficult to determine by other approaches. Nick Metropolis termed the name Monte Carlo method while working with Edward Teller, John von Neumann, Stanislaw Ulam, and Robert Richtmyer as they developed techniques for using the new computers in the late 1940s and early 1950's at Los Alamos National Laboratories.[98](pg 96), [99] Prior to the formal coining of the name, Enrico Fermi used the method in hand calculations. His use can be traced back as early as 1934 while working on neutron diffusion. [98](pg 98), [99](pg 128). Statistical sampling had been developed for quite some time. However, without computers, the process was very laborious. Early MC was performed on mechanical calculators and then mechanical punch card computers.[98] [99] In [100], Ulam, Richtmyer, and von Neumann discussed the statistical approach for solving neutron diffusion using Monte Carlo techniques.[99] The first computerized Monte Carlo calculations were conducted on the recently improved ENIAC in 1948.[98](pg 100). In 1949, Metropolis and Ulam presented "The Monte Carlo Method" [101] as "the motivation and a general description of a method dealing with a class of

problems in mathematical physics. The method is, essentially, a statistical approach to the study of differential equations, or more generally, of integro-differential equations that occur in various branches of the natural sciences."

Modern computing capabilities have rapidly expanded the use of the Monte Carlo method and many related techniques. It is used for simulations in a wide variety of fields. Engineering, finance, health care, manufacturing, military, political science and weather are just a few.

The Monte Carlo (MC) methods or techniques related to this proposal are used to evaluate the integral of complex random variable functions. The relationship is expressed below.

$$y = f(\boldsymbol{\theta}) \tag{3-6}$$

where $\boldsymbol{\theta}$ is a random variable or set of random variables. Equation (3-6) is the same form as the measurement equation (2-1).

A MC method will evaluate equation (3-6) at pseudo-random values of $\boldsymbol{\theta}$ to generate a simulation. These pseudo-random values will be generated according to the desired distribution to represent the system of interest. Multiple simulations, or runs, are evaluated with enough samples to ensure accurate results. In the end, these simulations are evaluated to estimate the integral of equation (3-6) in the region of interest.

MMAE methods are similar to Monte Carlo techniques. Kalman filters are executed, and their resulting estimates are weighted with a multiple hypothesis testing method. For this research, the focus of the MMAE is using maximum likelihood to evaluate the weights and ultimate sum the weight multiplied times the individual filter parameter. These individual evaluations are summed for the overall estimate. In effect, each Kalman filter of the MMAE is an evaluation of equation (3-6). This is a simple way to stratify the parameters θ into grid based strata. Evaluating this grid stratification approach would be closer to a rectangular rule integration rather than a pseudo-randomly distributed evaluation for integration using the MC method. For pseudo-random distribution of θ, each parameter is an independent and independently distributed (IID) random variable. (IID simply means that each random variable has the same probability distribution as the other random variables and the random variables are mutually independent.[102]) IID can be a uniform distribution over the entire region of interest or a normal distribution focused on the known or initial guess or one of a multitude of techniques to fit the desired need of the problem of interest.

3.5.2 IID Sampling

The focus of this research is two stratification methods, selected grid-based sampling (SGBS) and Latin Hypercube sampling. The independent and

independently distributed (IID) approach is briefly covered to provide a comparison bound with LHS.

IID describes data where the samples are independent and independently distributed, hence the name. IID means that for a given set of data $\boldsymbol{\gamma}$, each set, $\boldsymbol{\gamma}^{(i)}$ of samples is taken from a "fixed ("stationary") probabilistic model."[102] Independence is described by equation (3-7) below for two events *A* and *B*.[103](pg 65). For the parameter sample section case, given one sample selection for a parameter, it does not affect the probability sample selection for another dimension of the parameter space.

$$P(B \mid A) = P(B) \text{ or } P(A \mid B) = P(A) \tag{3-7}$$

The concept of independence can be expanded by the product rule to show that the probability of both sample selection A and sample selection B occurring is simply the product of their independent probability of the events occurring. This is reflected for two sample events A and B below.

$$P(B \cap A) = P(A)\ P(B) \tag{3-8}$$

The above concepts described for the two event case can simply be expanded to cover the number of samples, *n*, to collect an entire sample set per equation (3-2).

The two most common probability distributions are the uniform and Gaussian (normal) probability distributions. For the purpose of sampling, the uniform distribution would be used to populate the samples with equal probability

between the upper and lower bounds of each parameter. The samples are typically implemented with a pseudo-random number generator that generates values between 0 and 1. The samples are adjusted by scaling to the appropriate range and shifted to start at the lower bound. Figure 3-4 illustrates the uniform distribution. Figure 3-5 shows simple uniformly distributed samples taken (a) in two dimensions and (b) in three dimensions. Figure 3-6 shows samples with a Gaussian (normal) distribution for both (a) two dimensions and (b) three dimensions. This method is similar to the uniform distribution with the exception that the goal is to center the distribution on a mean value, μ, and scale the normalized Gaussian distribution (standard normal distribution) by multiplying by the appropriate standard deviation, σ. For the normal distribution, care must be taken to limit the low probability. Typically, this is handled by truncating the sampled generated outside the acceptable range of the parameter. There are a number of methods to handle this truncation. The easiest is to set the value to the mean to avoid flattening the probability distribution. Other methods are available that implement the truncation to more precisely represent the original normal distribution. Botev provides one method called "minmax tilting" in [104]. Chopin discusses specialized algorithms guaranteed to generate a random variable from the desired distribution concerning parameters and truncated intervals in [105].

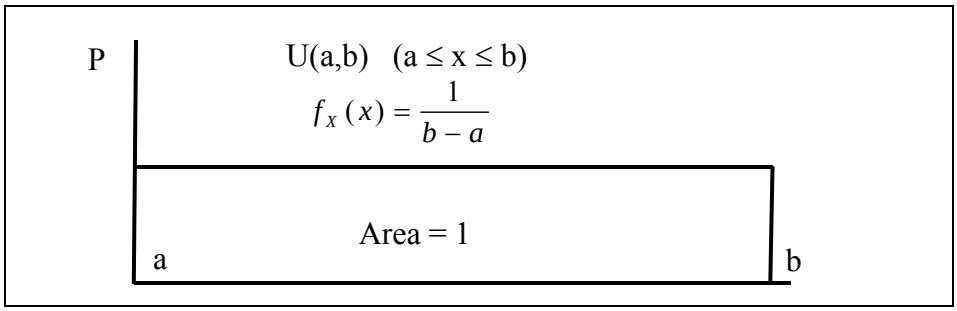
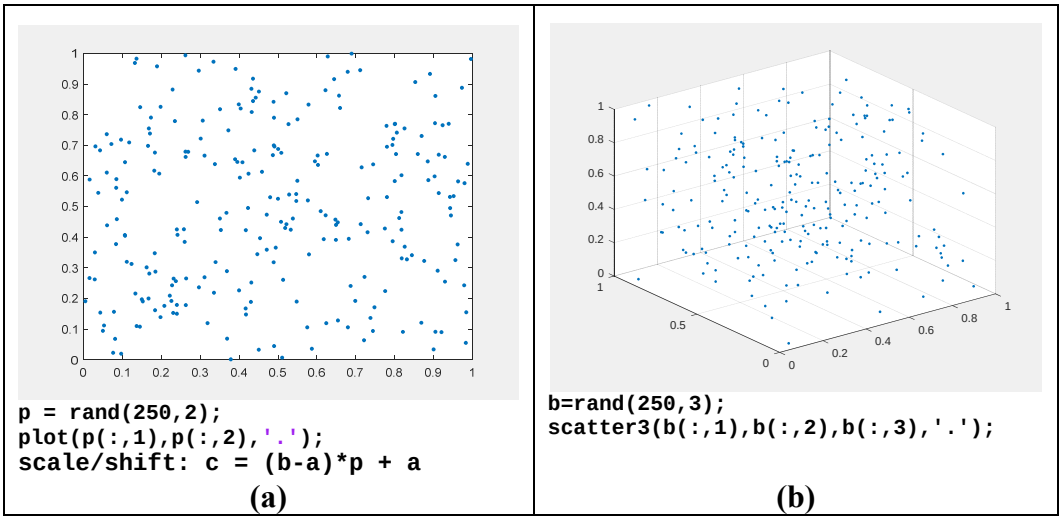


Figure 3-4 Uniform distribution, U(a,b)



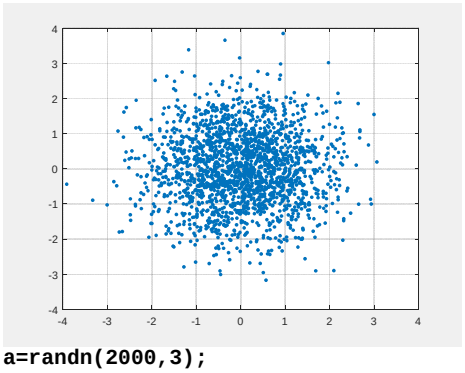
```
p = rand(250,2);
plot(p(:,1),p(:,2),'.');
scale/shift: c = (b-a)*p + a
```

(a)

```
b=rand(250,3);
scatter3(b(:,1),b(:,2),b(:,3),'.');
```

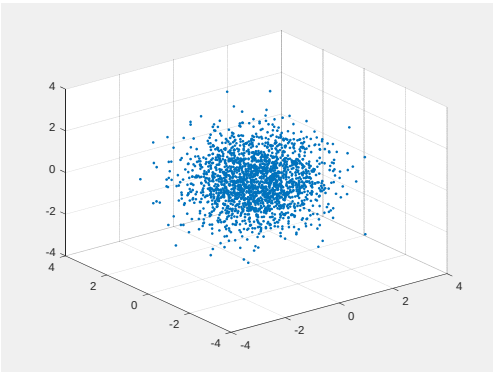
(b)

Figure 3-5 Uniform IID samples, U(0,1)



```
a=randn(2000,3);
plot(a(:,1),a(:,2),'.');
scale/shift: c = sigma*p + mean
```

(a)



```
a=rand(250,3);
scatter3(a(:,1),a(:,2),a(:,3),'.');
```

(b)

Figure 3-6 Gaussian IID samples, N(0,1)

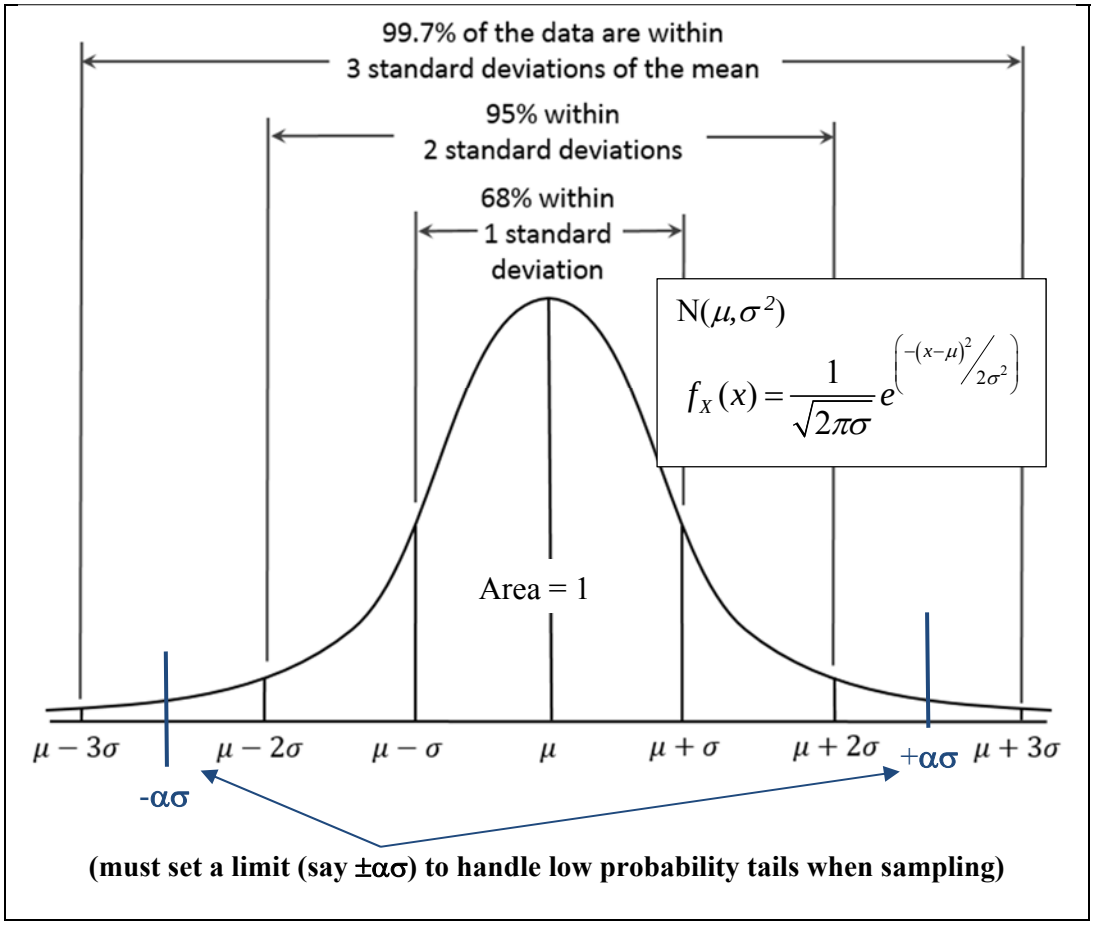


Figure 3-7 Gaussian distribution, N(μ,σ^2) (adapted from [106])

3.5.3 Latin Hypercube Sampling

Monte Carlo integration has a typical averaged squared error on the order of $O\left(\sigma^2/n\right)$. The denominator is the variance σ^2 and n is the number of simulations. To lower the error, the number of simulations, n, is usually increased resulting in an error reduction proportional to $1/\sqrt{n}$. Alternatively, a new MC formulation is introduced that produces the same answer for the problem. This new formulation reduces the variance σ^2 rather than increasing n. These techniques are called variance reduction techniques. [107](chapter 8, pg 3)

Many variance reduction techniques are available, such as antithetic sampling, stratification, common random numbers, conditioning and control variates, and importance sampling of [107](chapter 8) along with advanced methods described in [107](chapter 10) such as grid-based stratification, stratification and antithetics, Latin hypercube sampling, orthogonal array sampling, adaptive importance sampling, nonparametric AIS, generalized antithetic sampling, control varies with antithetics and stratification, bridge, umbrella and path sampling techniques. The goal is to reduce computational complexity and provide ease of implementation. The focus of this research has narrowed to LHS due to these goals together with the amount recent research in LHS.

For the purpose of this research, all parameters of interest are considered equally important. We have seen that grid based stratification multiplies the

number of required samples or filters per parameter to generate ever increasing models. Looking at this from another view, assume we have the computational capacity to compute 100 filters, *n*. It is desired to divide n into equal strata numbers for each parameter *d*. For d >1, the grid-based method can only generate $n^{1/d}$ strata per component. This is far less than the number of available samples. For example, if *n* is 100 and *d* is 3, the number of available strata is less than 5 for each. For the sake of accuracy, using 4 division per parameter is not acceptable. [107](chapter 10, pg 8) Computational capacity prevents more samples. Therefore, we visit the variance reduction technique of Latin Hypercube Sampling (LHS) to improve the result.

Latin Hypercube Sampling is a variance reduction technique that was discovered by different researchers for different applications. Reference [107] provides a good summary of the origin of LHS which was introduced to literature in 1979 by McKay et al. in [108]. The term Latin Hypercube is an extension of Latin squares used in experimental design.[107] Literature often refers to LHS as placing a point in the unit cube and uses the terminology $[0,1)^d$ where *d* is the dimension equal to the number of parameters.[107] The 1979 McKay et. al paper, [108], introduced LHS a way to explore computer experiments**.** LHS provides a way to explore the input space of a function while automatically stratifying the important variables without the user having to know the details of the variables. [107] Reference [108] proved the variance of the estimated mean of the LHS was

less than or equal to the variance of the estimated mean of IID when the function sampled is monotone in each of its d input variables.[107]

In general, the parameter space is stratified in LHS by generating *n* equal strata for each parameter. This is illustrated in Figure 3-8 where the number of parameters *d* is 2 and the LHS sample size of 10 points.

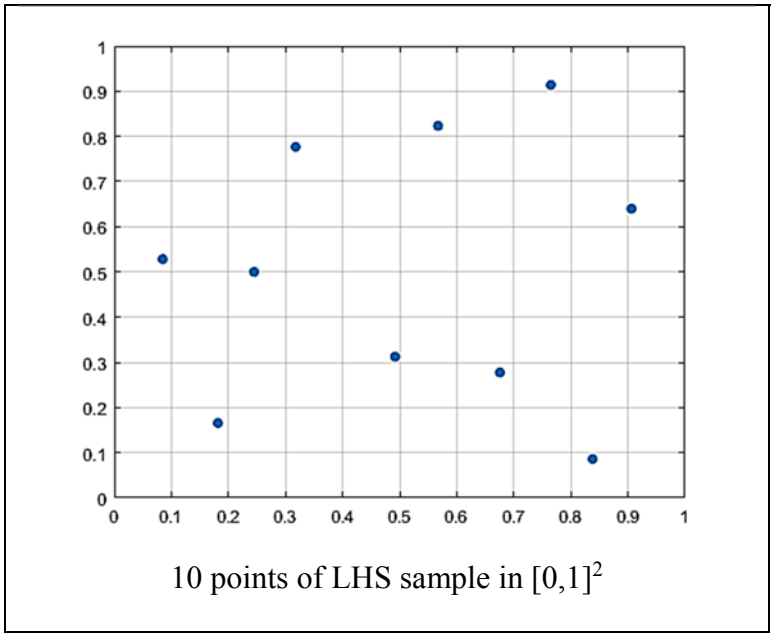


Figure 3-8 2-d LHS example

An alternate view is shown in Figure 3-9. Here the dimension *d* is 3 representing the number of parameters and sample size *n* remains 10. Increasing the number of parameters does not automatically increase the sample size.

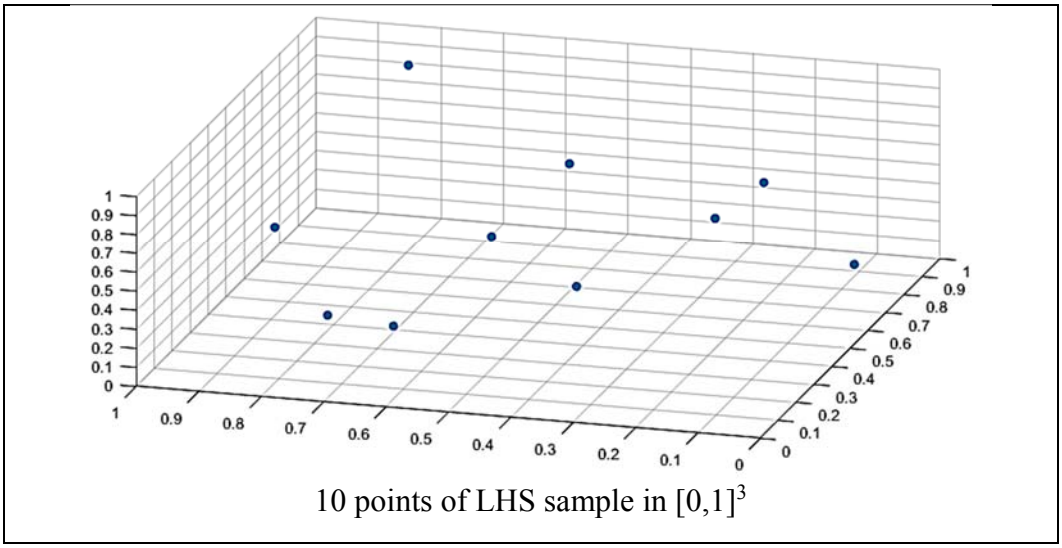


Figure 3-9 3-d LHS example

The parameter space can be described as a unit cube as dimension *d* of the number of parameters as $[0,1)^d$. The LHS process generates samples of each dimension between 0 and 1. Each dimension is then scaled to the actual parameter range of interest in a scaling and shifting technique similar to the IID uniform transformation techniques mentioned in 3.5.2.)

Owen provides a concise summary of the alternative implementations of uniform LHS in [107]. The first and most common method is described by the following equation

$$X_{ij} = \frac{\pi_j(i-1)+U_{ij}}{n}, 1 \le i \le n, 1 \le j \le d \tag{3-9}$$

where π_j is a uniform permutation of {0, …, *n*-1} and U_{ij} is uniform random variable, $U_{ij} \sim U[0,1)$. The number of samples is *n*, and the number of permutations

is *d*. The *d* permutations and *nd* random variables are mutually independent. With *n* samples, *nd* strata of volume 1/*n* each are balanced. [107] The LHS estimate of the mean $\mu = \int_{[0,1)^d} f(x)$ is

$$\hat{\mu}_{LHS} = = \frac{1}{n}\sum_{i=1}^{n} f(X_i) \tag{3-10}$$

with X_i generated by (3-9). For uniform LHS, the average squared error is on the order of $O(\sigma^2/n)$.[107] The comparative MC averaged squared error is on order of $O(\sigma^2/n)$.[107]

Analysis of variance (ANOVA) techniques [109] can be used to study the variance. For complex models, the variance σ^2 is not typically known but can be estimated from samples.

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(Y_i - \hat{\mu}_n\right)^2 \tag{3-11}$$

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{\mu}_n\right)^2 \tag{3-12}$$

The most common method used is (3-11) because it is unbiased.[107] The expected value is

$$E\left(s^2\right) = \sigma^2 \text{ for } n \geq 2 \tag{3-13}$$

With the variance estimate s^2, the error is on the order of $s/\sqrt{n}$. The estimated variance has an expectation, or mean, of μ. Applying CLT, $\hat{\mu}_n - \mu$ has approximately a normal distribution with a mean of zero and variance $\sigma^2/\sqrt{n}$.[107]

Reducing the variance σ^2 reduces the error, hence the name variance reduction. "LHS can be much more accurate than plain Monte Carlo." [107](Chapter 10, pg 10) Much better results can be achieved for small *d* and well-behaved *f()*.[109] If *f()* is additive, meaning it can be represented by a set of functions where each dependent fewer number of parameters, then the average error is small because of a smaller variance. "Asymptotically, the variance is less than that obtained using simple random sampling, with the degree of variance reduction depending on the degree of additivity in the function being integrated."[110] The improvement seen by LHS is maximized when all the additive functions are dependent only on one variable.[109] The stratification technique with uniform LHS will cover all the regions of every parameter no matter how the individual sets are paired. This can be seen in Figure 3-8. There are 25 samples spanning each parameter strata. Increasing the parameter space dimension will still leave 25 samples in each stratum of each parameter as illustrated in Figure 3-9 where the dimension is increased by one parameter.

An alternate implementation of LHS is a centered version, also called latticed LHS, where the value is not randomly distributed within each stratum but centered. The following equation describes the centered version

$$X_{ij} = \frac{\pi_j (i-1) + 1/2}{n}, 1 \le i \le n, 1 \le j \le d \tag{3-14}$$

Here, the uniform portion, U_{ij}, is replaced by the term 1/2. The centered version was actually discovered much earlier that LHS. In 1954, the latticed method was used in literature by Patterson in [111] in relation to agriculture experiments. This version has a small bias on the order of $O\left(1/n^2\right)$.[107]

All variants thus far are based on uniform distribution of independent parameters over the unit cube. Another alternative implementation of LHS is to evenly partition the cumulative distribution function (CDF) as illustrated in Figure 3-10. When implementing using the normal distribution, this approach focuses more samples near the mean value μ.

This method is used by Stein in [110] to produce LHS samples when the components of the input variables are statistically dependent. Stein's algorithm produces a sample vector that has the approximate joint distribution when large numbers of samples are taken. An implementation of this approach is provided in MATLAB using the lhsnorm function. The method can be used to stratify the normal CDF about the mean of parameters using a covariance matrix. An independent normal CDF implementation is implemented using a diagonal

covariance matrix. As in the uniform LHS method, a centering method is available to set the sample at the center of each stratum rather than randomly distributed through each.

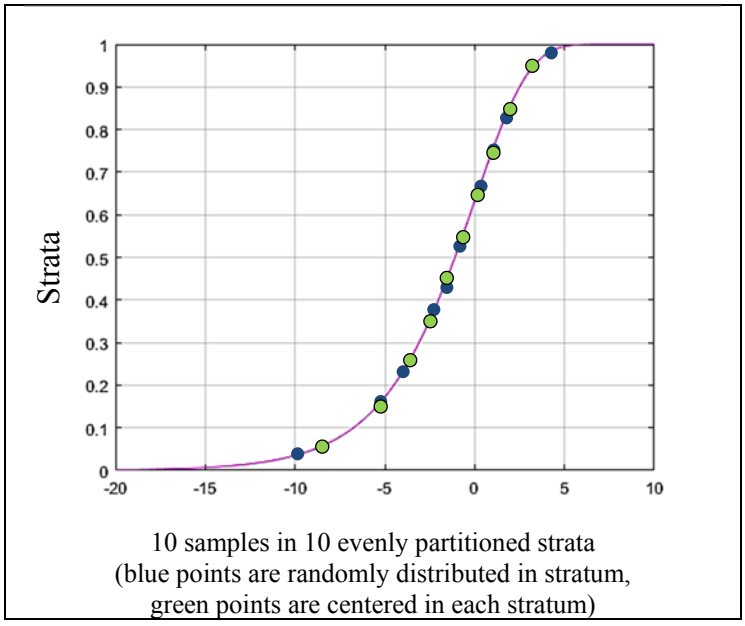


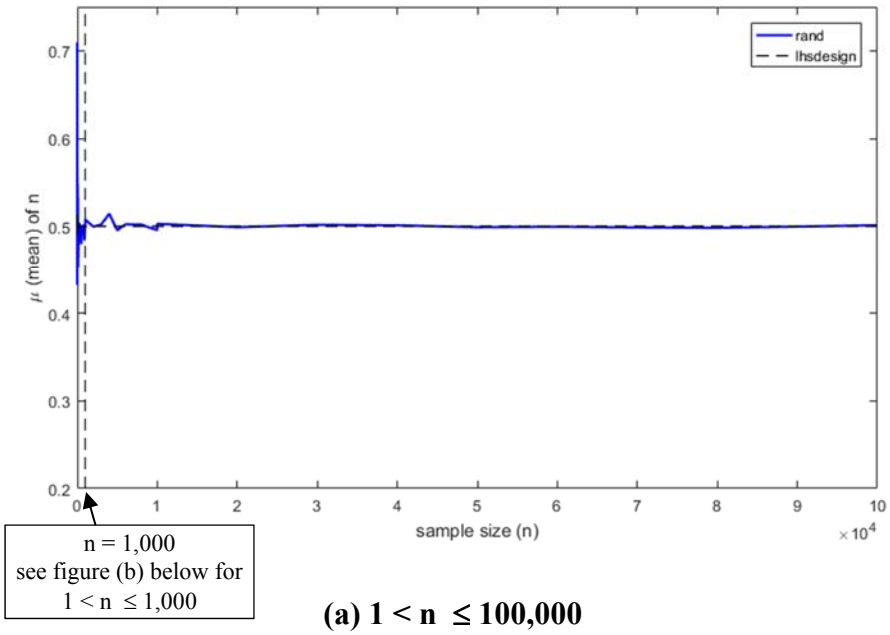
Figure 3-10 1-d LHS based on CDF

3.5.4 Evaluating Confidence

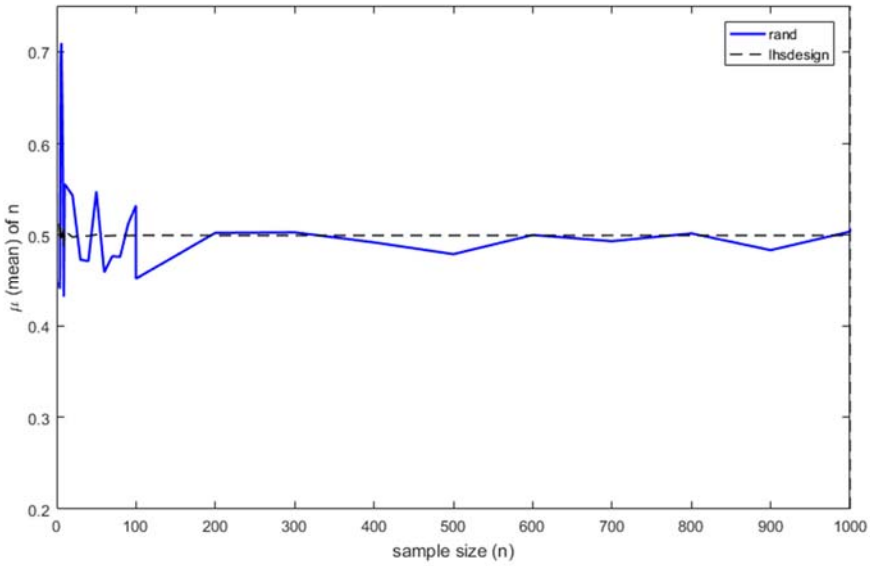
There is much research to validate MC methods including its many varieties. References [110, 112-123] provide a several approaches for understanding uncertainty and sensitivity. LHS is typically bounded by MC methods; therefore, standard MC confidence interval analysis are used with assumptions about the MMAE system of interest. This analysis provides conservative guidance on selecting the sample size. A guaranteed confidence

interval for MC is provided by in [124]. This method, like all the others, relies on sample runs to better understand the system. The guaranteed confidence interval has a large computational cost by a factor ranging from 2 to 100 over methods base off of the CLT with known standard deviation σ.[124](pg 126)

A simple example comparing LHS with IID techniques is shown in Figure 3-11. Here uniform LHS samples were generated following the MATLAB lhsdesign function. Their mean value was compared with the mean value of normally distributed random numbers from the MATLAB rand function. The figure shows that LHS samples produce a mean value closer to 0.5 for a smaller number of samples. As the sample size increases, the difference diminishes. The main takeaway from this example is that estimating the required number of samples for a Monte Carlo analysis with IID samples will typically bound the error of LHS methods. As previously stated, LHS will typically outperform standard IID analysis.



(a) 1 < n ≤ 100,000
(samples sizes 10,000 and larger produce similar results for both rand and lhsdesign)



(b) 0 < n ≤ 1000 (zoomed in view of figure (a))
(lhsdesign consistently produces a mean near 0.5 throughout the samples size range starting at 1)

Figure 3-11 Univariate LHS and IID mean vs. sample size

When knowledge about the variance σ^2 is available, one method to determine a starting point for sample size estimate follows confidence interval analysis using Chebychev's inequality. [107](section 2.9)

Following, Owen's analysis from [107](section 2.9): The variance is assumed or known to be bounded by σ_0^2 , then the variance of the expected value of the mean is σ_0^2/n. From Chebychev's inequality,

$$p\left(\left|\hat{\mu}-\mu\right| \geq k\sigma_0\sqrt{n}\right) \leq \frac{1}{k^2} \tag{3-15}$$

A conservative 99% confidence interval for μ is found by using

$$\hat{\mu} \pm \frac{10\sigma_0}{\sqrt{n}} \tag{3-16}$$

When the variance is less than or equal to the assumed variance, i.e., $\sigma^2 \leq \sigma_0^2$, equation (3-16) has at least 99% probability of containing the mean μ. This defines a confidence interval width of $10\sigma_0/\sqrt{n}$. Therefore, a 99% confidence interval of a given width ε will define a single dimension sample size as follows.

$$n_{Chebychev} = \frac{400\sigma_0^2}{\varepsilon^2} \tag{3-17}$$

Owen states this is more conservative and the interval only needs to multiply $\sigma_0/\sqrt{n}$ by 2.58 rather than 10. Therefore, an adjusted sample size is adjusted by a factor of $\left(2.58/10\right)^2 \approx 1/15$ as follows.

$$n_{adjusted} = \frac{26.67\sigma_0^2}{\varepsilon^2} \tag{3-18}$$

Owen provides an alternate sample size formulation based off of Hoeffding's inequality. Corollary 2.1 on pg 36 of Chapter 2 in [107] handles the case of common bounds on the random variables.

Let Y1, …., Yn be independent random variables with mean μ such that $a \le Yi \le b$ for finite a and b. Let $\hat{\mu} = (1/n)\sum_{i=1}^{n} Y_i$ and $\delta \in (0,1)$, then for $\varepsilon > 0$

$$p\left(\left|\hat{\mu} - \mu\right| \ge \varepsilon/2\right) \le \delta \tag{3-19}$$

when

$$n_{Hoeffding} \ge \frac{2(b-a)^2 \log(2/\delta)}{\varepsilon} \tag{3-20}$$

The sample size $n_{Hoeffding}$ provides the sample size for the interval $\hat{\mu} \pm \varepsilon/2$ to have a guaranteed confidence of 100(1-δ)%. One difficulty of applying this formulation is that the independent random variables span the same bounds *a* through b.

This research has derived a slightly different formulation based off of Chebychev's inequality. To evaluate confidence and uncertainty in MMAE, start with the assumption that each set of MMAE filters is a run of a Monte Carlo experiment for each available data point (measurement). Furthermore, assume the MMAE model has executed to the point where the weights have converged for each

model. With this assumptions, the *m* filters of MMAE is equivalent to *n* experiments of MC. Each filter is defined by equations (2-37) and (2-38). They are represented in further detail as the estimate using the index of the particular filter re-writing equation (2-1) below with the added term of the weight, *wt*.

$$\hat{\mathbf{y}}^{(i)} = f\left(\hat{\mathbf{x}}^{(i)}, \mathbf{x_0}^{(i)}, \boldsymbol{\theta}^{(i)}, u, w^{(i)}, v^{(i)}, wt^{(i)}\right) \tag{3-21}$$

Index *i* represents the i^{th} filter model, state estimate $\hat{\mathbf{x}}$, initial conditions $\mathbf{x_0}$, parameters $\boldsymbol{\theta}$, process noise *w* and measurement noise *v* and MMAE weight *wt* for each data point available. The input term is consistent across all models. With this understanding, the total MC equivalent evaluation for the current data point is as follows.

$$\begin{aligned} y_{total} &= \sum_{i=1}^{m} {}^{(i)}\hat{y} \\ &= {}^{(1)}\hat{y} + {}^{(2)}\hat{y} + \ldots + {}^{(i)}\hat{y} + \ldots {}^{(m-1)}\hat{y} + {}^{(m)}\hat{y} \end{aligned} \tag{3-22}$$

The individual estimates of equation (4.1) can be considered the instance of the random variable $\hat{y}$. The case where the distribution of the model is random for each parameter θ is called an independent and independently distributed (IID) random sequence. Therefore, each model estimate $\hat{\mathbf{y}}$ has the same probability distribution. Furthermore, each estimate has the same mean, μ, and variance, σ^2, as expressed below.

$$
\begin{aligned}
&E\left[{}^{(i)}\hat{y}\right] = \mu \\
&\mathrm{var}\left[{}^{(i)}\hat{y}\right] = \sigma^2 \\
&i = 1, 2, \ldots, m
\end{aligned}
\tag{3-23}
$$

The MMAE performs the random measurement *m* times for each data point. The empirical average is

$$
\begin{matrix} \textit{Emperical} \\ \textit{Average} \end{matrix} = \frac{{}^{(1)}\hat{y} + \cdots + {}^{(m)}\hat{y}}{m} \tag{3-24}
$$

The Law of Large Numbers states for large *m*, the average is very close to the expected value μ with high probability. Theorem 4 of [125](pg 6) is stated for reference.

Theorem 4. Let X1, …, X IID random variables with E[X_i] = μ and var(X_i) for all i. Then

$$
P\left(\left|\frac{X_1 + \cdots + X_n}{n} - \mu\right| \geq \varepsilon\right) \leq \frac{\sigma^2}{n\varepsilon^2} \tag{3-25}
$$

where the right hand side goes to 0 as $n \to \infty$.

Applying Theorem 4 of [125] to the MMAE evaluation of a single data point with the assumption of converged weights, the evaluation of probability can be expressed as

$$
P\left(\left|\frac{{}^{(1)}\hat{y} + \cdots + {}^{(m)}\hat{y}}{m} - \mu\right| \geq \varepsilon\right) \leq \frac{\sigma^2}{m\varepsilon^2} \tag{3-26}
$$

Assuming the variance is bounded by the measurement noise, *v~N(0,R)*, equation (3-26) can be transformed into a conservative probability relationship.

$$P\left(\left|\frac{^{(1)}\hat{y}+\cdots+{}^{(m)}\hat{y}}{m}-\mu\right|\geq\varepsilon\right)\leq\frac{R}{m\varepsilon^2} \tag{3-27}$$

The actual interest is the region within $\pm\,\varepsilon$. Therefore, the probability is $(1-P)$ of (3-27).

$$\begin{aligned}&P\left(\left|\frac{^{(1)}\hat{y}+\cdots+{}^{(m)}\hat{y}}{m}-\mu\right|\leq\varepsilon\right)\\&=1-P\left(\left|\frac{^{(1)}\hat{y}+\cdots+{}^{(m)}\hat{y}}{m}-\mu\right|\geq\varepsilon\right)\geq\frac{R}{m\varepsilon^2}\end{aligned} \tag{3-28}$$

In summary, the confidence interval can be generated from equation (3-28)

$$P\left(\text{value is in interval}\left(x-\varepsilon,x+\varepsilon\right)\right)\geq\frac{\beta}{100} \tag{3-29}$$

Therefore, in the context of the Law of Large Numbers (LLN), the relationship between variance σ^2, number of samples m, precision error ε and confidence percent β is

$$\frac{\sigma^2}{m\varepsilon^2}=1-\frac{\beta}{100} \tag{3-30}$$

Solving for m provides

$$m=\frac{\sigma^2}{\varepsilon^2\left(1-\frac{\beta}{100}\right)} \tag{3-31}$$

Adjusting for multiple parameters assuming that the samples are independent, the individual P_i probabilities of each parameter

$$\begin{aligned} {}^{(1)}\left(\frac{\sigma^2}{m\varepsilon^2}\right)\bullet\cdots\bullet{}^{(n)}\left(\frac{\sigma^2}{m\varepsilon^2}\right) &= \left(\frac{\sigma^2}{m\varepsilon^2}\right)^n \\ &= 1-\frac{\beta}{100} \end{aligned} \tag{3-32}$$

Therefore,

$$m = \frac{\sigma^2}{\varepsilon^2\left(1-\frac{\beta}{100}\right)^n} \tag{3-33}$$

3.6 Summary

Both SGBS and LHS are valuable tools for choosing the hypothesis models in the FDD framework. SGBS provides a method for stratification that can be applied to the initial sample parameter set and future resample sets. The accuracy of the MMAE process can fit the desired goal by selecting the right number of models to span each parameter space. By the nature of SGBS's design, it requires more models costing more time to perform computations. The number of models grows according to a product rule with each new dimension added to the parameter space. Therefore, an alternate approach is desired to keep the number of models in a range that is computable with the resources available.

Statistical sampling techniques were explored, and LHS was selected. LHS was chosen as the primary sampling technique for this research effort for four main reasons. First, the effect of increasing the number of parameters does not

automatically increase required number of samples by a power rule as SGBS does in equation (3-5). The total number of samples can be kept the same or increased depending on the desired accuracy of the results. Even if the number of samples are increased, LHS does not have to grow by the power rule and the stratification approach is maintained. Second, LHS is widely used in MC type research with positive results. It is a proven tool and is part of standard analysis software for over a decade. Third, LHS is very simple to implement. Additionally, there are methods to cover both uniform and normal distributions. Each method may also select the option of centering the sample. The algorithms exist in many software tools such as MATLAB. Finally, LHS provides adequate results. LHS will populate the models more sparsely; therefore, the accuracy of the MMAE process will be affected. Additionally, LHS is a statistical sampling technique. The results will will be different each time even if the same number of samples is chosen.

CHAPTER 4

GRid Adaptive Parameter Estimation (GRAPE)

The proposed approach is called the grid adaptive parameter estimation (GRAPE). This method is a modification to the MMAE applied to FDD following a consistency-based fault detection and diagnosis for linear systems. This chapter presents the detailed modifications to the standard MMAE algorithm. A summary of applied approach to FDD is then presented. A simple linear model of a hydroelectric actuator is used to illustrate the algorithm. The simulation results are discussed in the context of the algorithm components. The chapter concludes with a summary of future work to be performed on the linear version of GRAPE.

4.1 GRAPE Algorithm

Modifications were made to the base MMAE approach to track the system parameters. The same weights used for the state estimates $\hat{\mathbf{x}}$ are used for the parameter estimates $\hat{\boldsymbol{\theta}}$ as illustrated by Figure 4-1 Linear GRAPE high-level summary. Additional modifications were made to perform online dynamic resampling of the parameters from the parameter space. This modification is illustrated in Figure 4-3. The details of dynamic sampling including resampling are covered in section 4.1.2. For the purpose of determining convergence of the MMAE system, an additional step was added after step 9 to test the convergence of the MMAE estimates. If the convergence criteria is met, a series of additional steps are taken. First, the parameters are resampled from adjusted bounds. Second, the

models are regenerated for each parameter set. Third, a reinitialization process is executed. This reinitialization is different from the original initialization as it sets the center of each dimensional range to last MMAE parameter estimate for that range.

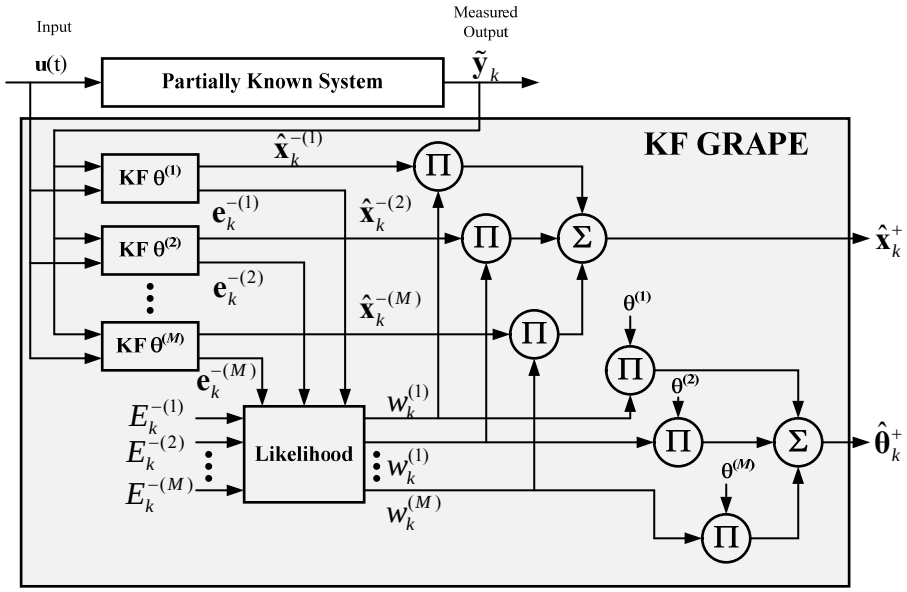
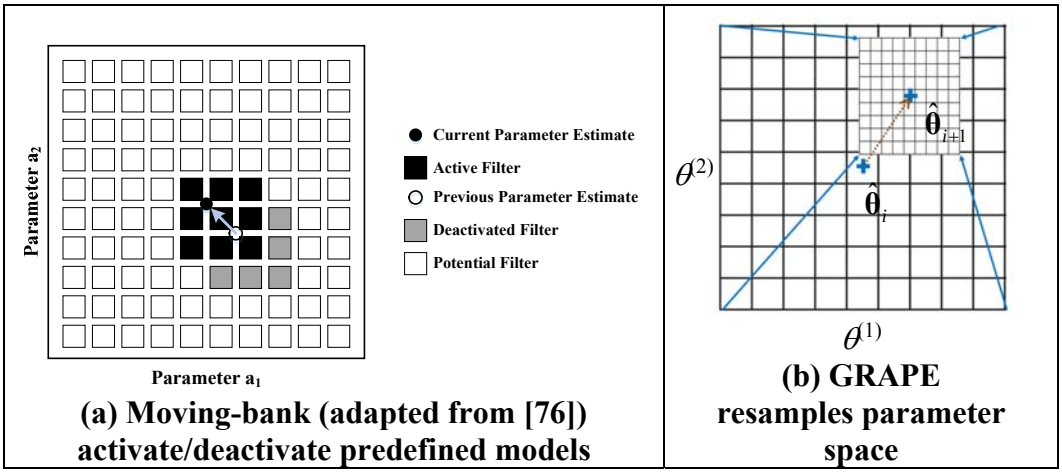


Figure 4-1 Linear GRAPE high-level summary



(a) Moving-bank (adapted from [76]) activate/deactivate predefined models

(b) GRAPE resamples parameter space

Figure 4-2 GRAPE vs. moving-bank approach to parameter Space

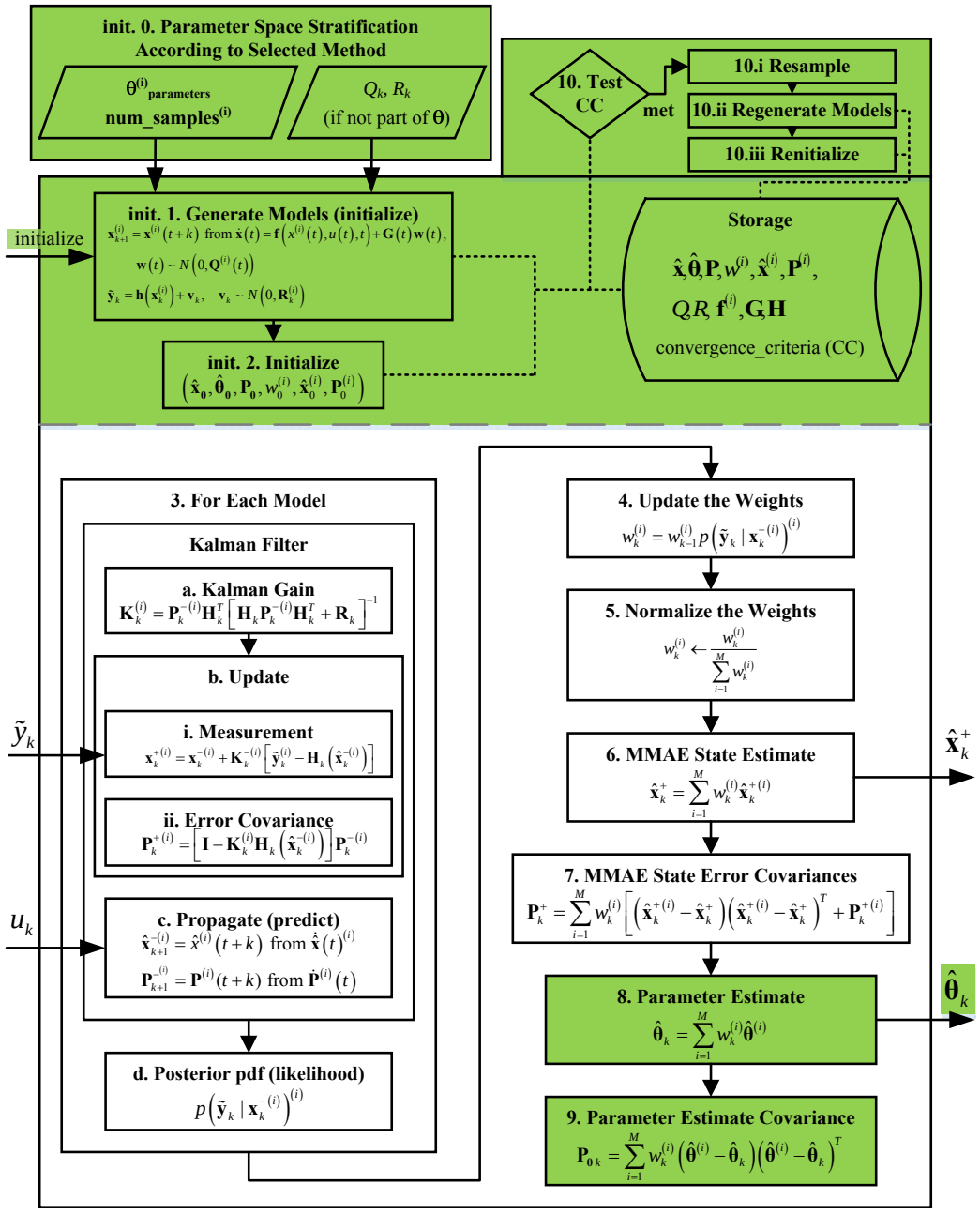


Figure 4-3 GRAPE MMAE high-level summary
(highlighted sections indicate deviation from base MMAE)

Table 4-1 GRAPE algorithm detailed steps

0. Parameter Space Stratification According to Selected Method
    a. SGBS – selected grid-based stratification
    b. LHS
        i. randomly distributed within uniform strata
        ii. centered within uniform strata
        iii. randomly distributed within normal (Gaussian) strata
        iv. centered within normal (Gaussian) strata
1. Generate models $\left(\boldsymbol{\Phi}^{(i)}\right)$ to represent parameter sets of interest $\left(\boldsymbol{\theta}^{(i)} \in \boldsymbol{\theta}\right)$

$$\mathbf{x}_{k+1}^{(i)} = \boldsymbol{\Phi}^{(i)}\mathbf{x}_k^{(i)} + \boldsymbol{\Gamma}_k u_k + \boldsymbol{\Upsilon}_k w_k; \quad w_k \sim N\left(0, Q_k^{(i)}\right)$$

$$\tilde{\mathbf{y}}_k = \mathbf{H}_k\mathbf{x}_k^{(i)} + v_k; \quad v_k \sim N\left(0, R_k^{(i)}\right)$$

2. Initialize, initial values for:
    a. MMAE state estimate $\left(\hat{\mathbf{x}}_\mathbf{0}\right)$
    b. MMAE parameter estimates $\left(\hat{\boldsymbol{\theta}}_\mathbf{0}\right)$
    c. Parameter estimate covariances $\left(\mathbf{P}_\mathbf{0} = E\left\{\hat{\mathbf{x}}_\mathbf{0}\hat{\mathbf{x}}_\mathbf{0}^{\,T}\right\}\right)$
    d. KF model state estimates $\left(\hat{\mathbf{x}}_0^{(i)}\right)$
    e. KF model covariances $\left(\mathbf{P}_0^{(i)} = E\left\{\hat{\mathbf{x}}_0^{(i)}\hat{\mathbf{x}}_0^{(i)T}\right\}\right)$
    f. Initialize filter weights $w_0^{(i)} = 1/M$
3. For each model:
    a. Propagate the next state (predict)

$$\hat{\mathbf{x}}_{k+1}^{-(i)} = \boldsymbol{\Phi}^{(i)}\mathbf{x}_k^{+(i)} + \boldsymbol{\Gamma}_k^{(i)}u_k$$

$$\mathbf{P}_{k+1}^{-(i)} = \boldsymbol{\Phi}^{(i)}\mathbf{P}_k^{+(i)}\boldsymbol{\Phi}^{(i)T} + \boldsymbol{\Upsilon}_k^{(i)}Q_k\boldsymbol{\Upsilon}_k^{(i)T}$$

    b. Calculate the Kalman Gain

$$\mathbf{K}_k^{(i)} = \mathbf{P}_k^{-(i)}\mathbf{H}_k^T\left[\mathbf{H}_k\mathbf{P}_k^{-(i)}\mathbf{H}_k^T + \mathbf{R}_k\right]^{-1}$$

    c. Update
        i. Perform the measurement update

$$\mathbf{x}_k^{+(i)} = \mathbf{x}_k^{-(i)} + \mathbf{K}_k^{-(i)}\left[\tilde{\mathbf{y}}_k^{(i)} - \mathbf{H}_k\hat{\mathbf{x}}_k^{(i)}\right]$$

        ii. Update the error covariance

$$\mathbf{P}_k^{+(i)} = \left[\mathbf{I} - \mathbf{K}_k^{(i)}\mathbf{H}_k\right]\mathbf{P}_k^{-(i)}$$

Table 4-1 GRAPE algorithm detailed steps (continued)

d. Calculate the posterior pdf (likelihood)

$$p\left(\tilde{\mathbf{y}}_k \mid \mathbf{x}_k^{-(i)}\right)^{(i)} = \frac{1}{\left[\det\left(2\pi\mathbf{E}_k^{-(i)}\right)\right]^{1/2}} \exp\left\{-\frac{1}{2}\mathbf{e}_k^{-(i)T}\left(\mathbf{E}_k^{-(i)}\right)^{-1}\mathbf{e}_k^{-(i)}\right\}$$

where

$e_k^{-(i)}$ is the measurement residual and measurement covariance is

$$\mathbf{E}_k^{-(i)} \equiv E\left\{\mathbf{e}_k^{-(i)}\mathbf{e}_k^{-(i)T}\right\} = H_k^{(i)}P_k^{-(i)}H_k^{(i)T} + R_k^{(i)}$$

4. Update the weights

$$w_k^{(i)} = w_{k-1}^{(i)} p\left(\tilde{\mathbf{y}}_k \mid \mathbf{x}_k^{-(i)}\right)^{(i)}$$

5. Normalize the weights

$$w_k^{(i)} \leftarrow \frac{w_k^{(i)}}{\sum_{i=1}^{M} w_k^{(i)}}$$

6. Calculate MMAE State Estimate

$$\hat{\mathbf{x}}_k^{+} = \sum_{i=1}^{M} w_k^{(i)} \hat{\mathbf{x}}_k^{+(i)}$$

7. Calculate the MMAE State Error Covariances

$$\mathbf{P}_k^{+} = \sum_{i=1}^{M} w_k^{(i)} \left[\left(\hat{\mathbf{x}}_k^{+(i)} - \hat{\mathbf{x}}_k^{+}\right)\left(\hat{\mathbf{x}}_k^{+(i)} - \hat{\mathbf{x}}_k^{+}\right)^T + \mathbf{P}_k^{+(i)}\right]$$

8. Calculate the Parameter Estimate

$$\hat{\boldsymbol{\theta}}_k = \sum_{i=1}^{M} w_k^{(i)} \hat{\boldsymbol{\theta}}^{(i)}$$

9. Calculate the Parameter Estimate Error Covariance,

$$\mathbf{P}_{\boldsymbol{\theta}k} = \sum_{i=1}^{M} w_k^{(i)} \left(\hat{\boldsymbol{\theta}}^{(i)} - \hat{\boldsymbol{\theta}}_k\right)\left(\hat{\boldsymbol{\theta}}^{(i)} - \hat{\boldsymbol{\theta}}_k\right)^T$$

10. Check the Convergence Criteria

11. Repeat 3 through 10 until the measurements are depleted

4.1.1 *Requirements and Assumptions*

Linear GRAPE requires an assumed model in discrete-time form with a fully defined parameter space according to equations (3-1) and (3-2). The method of sampling and resampling must be selected along with predefined convergence criteria.

GRAPE follows the same assumptions listed in Table 2-7. The GRAPE algorithm approach assumes that the actual parameter is within the outer limits of each parameter boundary definition. The method will start with a coarser sampling of a larger window and refine the window size to a smaller window as illustrated in Figure 4-4.

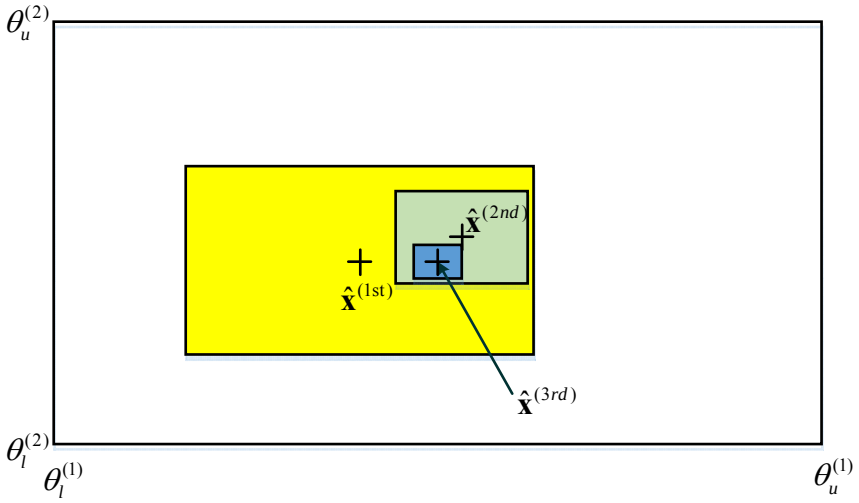


Figure 4-4 Dynamic resizing

4.1.2 *Dynamic Sampling*

This section covers the details of each stratification or sampling technique. Additionally, the convergence criteria along with resampling steps are covered.

Selected grid-based stratification (SGBS) is the baseline method that provides accurate results when enough models are chosen to accurately stratify the parameter space. The model count is typically higher for SGBS. The execution is proportionally slower than the LHS with the lower model count of the sampled model space. The user must choose between a high accuracy computationally complex SGBS and the lower accuracy less computationally complex LHS method.

4.1.2.1 Selected Grid-Based Stratification

SGBS simply divides each parameter space $\boldsymbol{\theta}^{(i)}$ into the defined number of strata n_i for that parameter to define the parameter values $\boldsymbol{\theta}^{(i)} = \left[\theta_1^{(i)} \quad \theta_2^{(i)} \quad \ldots \quad \theta_{ni}^{(i)} \right]$. The parameters may be divided into an equal number of strata where all n_i become n; however, that setting $n_i = n$ is not a requirement. Figure 4-5 provides an example of the first dimension divided into 11 strata and the second dimension is divided into 6 strata. Kalman filters are created at the intersection of each parameter value. This particular SGBS will require 66 (11x6 according to equation (3-4)) Kalman filters to implement.

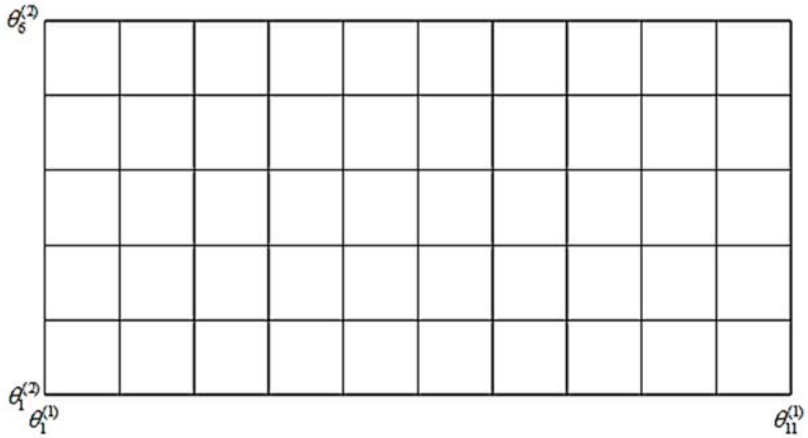


Figure 4-5 Selected grid-based stratification example

SGBS is similar to Maybeck's moving-bank with several differences. First, SGBS approaches the "bank" differently. The moving-bank method has a set of predefined Kalman filter models that are turned on or off for the estimate where SGBS defines the Kalman filter models online dynamically from the bounds of the individual parameters. Furthermore, the current set of SGBS KF models is completely redefined from adjusted parameter bounds.

4.1.2.2 Latin Hypercube Sampling

Latin Hypercube Sampling is an alternative to SGBS using the sampling technique originally used for Monte Carlo type methods. See 3.5.3 for full details of Latin Hypercube Sampling. LHS is implemented according to the four definitions listed below:

1. randomly distributed within uniform strata (according to equation (3-9))
2. centered within uniform strata (according to equation (3-14))
3. randomly distributed within normal (Gaussian) strata (see Figure 3-10)
4. centered within normal (Gaussian) strata (see Figure 3-10)

Approaches 3 and 4 require a mean μ and variance σ^2 of a parameter value rather than upper and lower bounds. The implementation of options 1 and 2 use the MATLAB lhsdesign function and the implementation of options 3 and 4 use the MATLAB lhsnorm function. Each function generates sample values between zero and one. These samples are scaled to match the implementation and shifted to the appropriate values. The sample size for LHS can be determined initially from equation (3-33). However, experimental results have shown that equation (3-33) is too conservative. In practice, LHS sample sizes one third of the value of SGBS sample sizes have worked well.

4.1.2.3 Convergence Criteria

For both SGBS and LHS, the initial approach for determining convergence of the MMAE estimate was to compare the mean squared error (MSE) of the weight changes between time steps to a threshold value. The weights are determined from the posterior pdf (equation (2-45)) which is calculated from the measurement residual (equation (2-39)) and covariance of the residual (equation (2-40)) for each Kalman filter model. Therefore, the approach is tied to the measurement residuals and their covariances. This method proved to work well as will be shown later in the example. (The topic of convergence was revisited for the nonlinear framework.

A more robust test for convergence was added to the more general nonlinear framework for GRAPE. This convergence method is discussed in section 5.3. The same approach is also applicable to the linear framework.)

4.1.2.4 Resampling

Currently, resampling is triggered by a positive flag generated from the MSE convergence criterial. The original windows are larger covering the entire parameter space as illustrated in Figure 4-4. The window resizes to a selectable percentage of the previous window size. Twenty percent of the previous window size worked well in practice. (Again, a more robust test for convergence was added to the more general nonlinear framework for GRAPE. This method and resample rules are discussed in section 5.3. These methods are also applicable to the linear framework.)

4.2 Application to Fault Detection and Diagnosis

GRAPE naturally lends itself for use in FDD. Assuming that both the nominal and faulty regions have been previously defined in Table 4-2, fault detection is a simple lookup comparison of the parameter estimate from the defined parameter based behaviors. Fault identification and isolation is a similar task. The fault can only be identified down to the fault behaviors where the parameters are part of the fault definition. If the fault definitions are not unique, then there will be more than one fault candidate.

Table 4-2 Fault behavior definitions required for FDD

For fault detection:

- Defined parameter space behavior boundary limits (upper and lower) for each parameter for the following defined cases
  - Nominal operation behavior: $B_{nominal}$
  - Faulty operation behavior: B_{faulty}
  - Safe operation is assumed to be: $B_{safe\ operation} = B_{nominal} \cup B_{faulty}$
- Assume failure when system is outside of the B_{safe} operation space

For fault diagnosis (isolation):

- Full parameter space behavior definitions for all faults individually B_{fi}
- Faults may overlap; however, unique identification may not be possible if they do

4.3 Application for System Identification

GRAPE can be used for system general system identification when there is no knowledge of the parameters values but a basic dynamic understanding of the system is known. The bounds must initially cover all possible values of the system. If the initial run does not converge, then the range may be extended further. The more generalize nonlinear GRAPE has this capability to move the window towards nominal values outside of the current window. This feature can also be implemented in the linear GRAPE framework.

4.4 Linear GRAPE Framework Results

A simplified hydroelectric servo model is used to demonstrate the principles introduced and to study the GRAPE method. The model and multiport block diagram are illustrated in Figure 4-6. The basic assumption include:

- amplifier is not power limited (exceeds required needs of valve operation)
- hydraulic pressure is sufficient for valve operation

The defining differential equation is derived from the free body diagram of the mass the servo is designed to move. The resultant equation is

$$m\ddot{x} + \left(b + RA^2\right)\dot{x} + \left(k + AG_pG_ah\right)x = AG_pG_aCu - F_L - \left(\pm f_c\right) \qquad (4\text{-}1)$$

The details of each variable are provide below in

Table 4-3. Equation (4-1) was derived from the system relationship of the component model in Figure 4-7, and the free body diagram and the equation development in Figure 4-8. It can be clearly seen that the system is of the form of equation (2-4) with the addition of a load force F_L and coulomb friction. For the purposes of simplicity, these two terms are assumed to equal zero. Furthermore, all variables are assumed to be a value that results in the ideal form of equations (2-8) through (2-10). The input is assumed to be sinusoidal. Therefore, the final model of the system is the harmonic oscillator of equation (4-2).

$$\ddot{x} + c\dot{x} + kx = \sin(t) \qquad (4\text{-}2)$$

Furthermore, the state-space model becomes

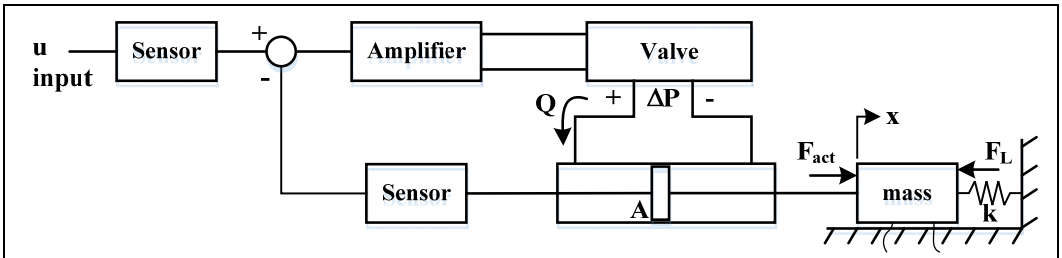
$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k & -c \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w \tag{4-3}$$

with the measurement model

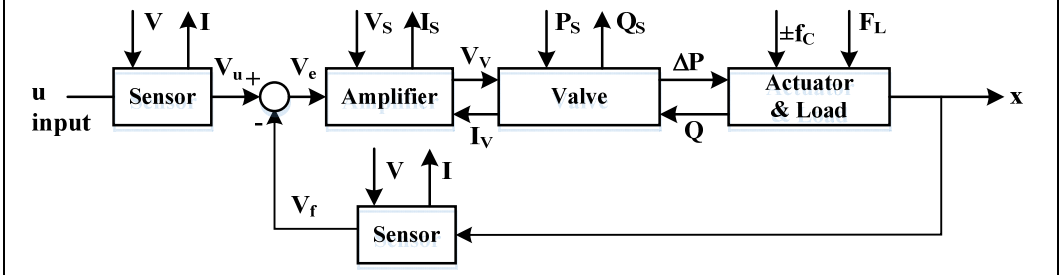
$$\tilde{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} v \tag{4-4}$$

Table 4-3 Variable descriptions for hydroelectric servo

Variable	Description
A	piston area
B	damping coefficient
C_s	input sensor gain
G_a	amplifier gain
G_p	piston valve gain
h	feedback sensor gain
K	spring coefficient (stiffness of connection)
R	piston valve impedance
$(\pm f_C)$	coulomb friction
F_L	load force
m	mass
u	input

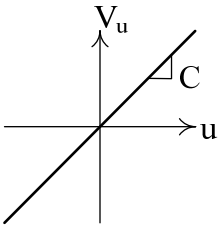


(a) hydroelectric servo component model



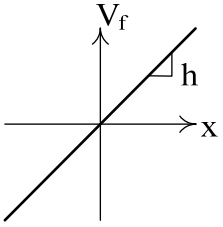
(b) multiport block diagram

Figure 4-6 Hydroelectric servo model



$V_u = Cu$

(a) input sensor



$V_f = hx$

(b) input sensor

$V_v = G_a\left(V_u - V_f\right)$

(c) amplifier and summing junction

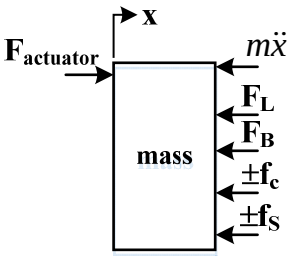
$Q = A\dot{x}$

(d) actuator and load

$\Delta P = G_p V_v - RQ$

(e) valve

Figure 4-7 Component models



(a) mass free body diagram

$$m\ddot{x} = F_{act} - F_B - F_S - F_L - (\pm f_C)$$
$$m\ddot{x} = A\Delta P - b\dot{x} - kx - F_L - (\pm f_C)$$
$$m\ddot{x} = A(G_p V_v - RQ) - b\dot{x} - kx - F_L - (\pm f_C)$$
$$m\ddot{x} = A(G_p(G_a(V_u - V_f)) - R(A\dot{x})) - b\dot{x} - kx - F_L - (\pm f_C)$$
$$m\ddot{x} = A(G_p(G_a(Cu - hx)) - R(A\dot{x})) - b\dot{x} - kx - F_L - (\pm f_C)$$
$$m\ddot{x} = AG_pG_aCu - AG_pG_ahx - RA^2\dot{x} - b\dot{x} - kx - F_L - (\pm f_C)$$
$$m\ddot{x} + (b + RA^2)\dot{x} + (k + AG_pG_ah)x = AG_pG_aCu - F_L - (\pm f_C)$$

(b) equation (4-1) development

Figure 4-8 Model development

A fictitious design goal is a system with behavior identified in Table 4-4.

Table 4-4 Behavior identification of electrohydraulic servo

Limits of explored behavior	$B_{\text{explored}} = \begin{Bmatrix} k \in \Re^{nx1} \mid 0.5 \le k \le 10.0 \\ c \in \Re^{nx1} \mid 0.5 \le c \le 5.0 \end{Bmatrix}$
Nominal operation behavior:	$B_{\text{nominal}} = \begin{Bmatrix} k \in \Re^{nx1} \mid k_{\text{l,nom}} \le k \le k_{\text{u,nom}} \\ c \in \Re^{nx1} \mid c_{\text{l,nom}} \le c \le c_{\text{u,nom}} \end{Bmatrix}$
Faulty operation behavior:	$B_{faulty} = \left\{ (c,k) \middle\vert \begin{matrix} k \in \Re^{nx1} \mid k_{\text{l,faulty}} \le k \le k_{\text{l,nom}}; k_{\text{l,nom}} \le k \le k_{\text{u,faulty}} \\ c \in \Re^{nx1} \mid c_{\text{l,faulty}} \le c \le c_{\text{l,nom}}; c_{\text{l,nom}} \le c \le c_{\text{u,faulty}} \end{matrix} \right\}$
Safe operation	$B_{safe} = \begin{Bmatrix} k \in \Re^{nx1} \mid k_{\text{l,faulty}} \le k \le k_{\text{u,faulty}} \\ c \in \Re^{nx1} \mid c_{\text{l,faulty}} \le c \le c_{\text{u,faulty}} \end{Bmatrix}$

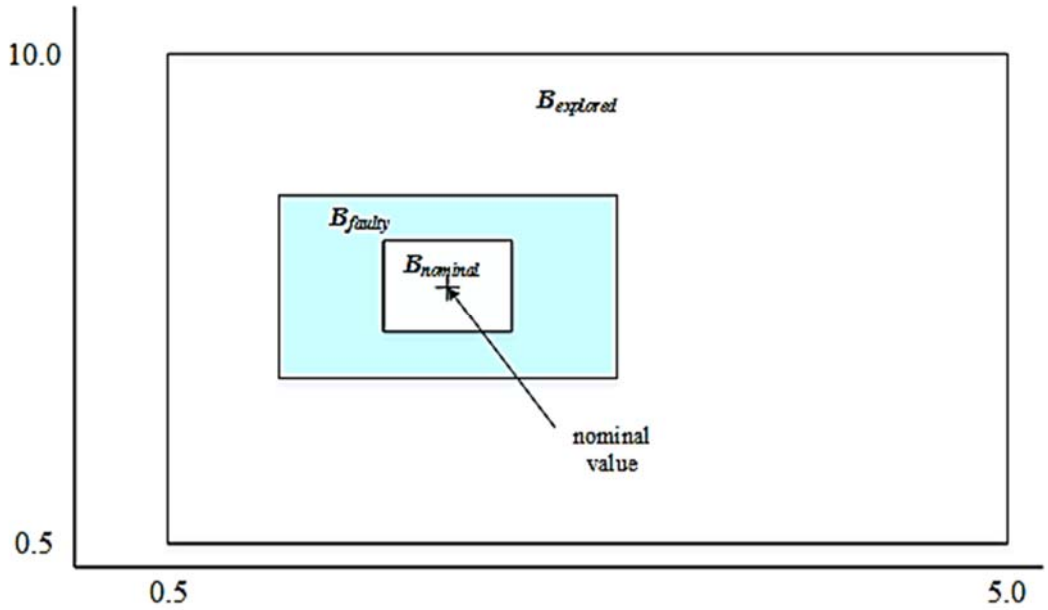


Figure 4-9 Behavior regions

Synthetic data was generated from the model of (4-2) for 100 seconds using the following true values k = 2.5 with each $c = \begin{bmatrix} 0.2 & 0.7 & 1.0 & 1.3 \end{bmatrix}$. The intent was to explore the system throughout the underdamped (c < 1), critically damped(c = 1) and overdamped (c > 1) regions. This synthetic data is presented in Figure 4-10.

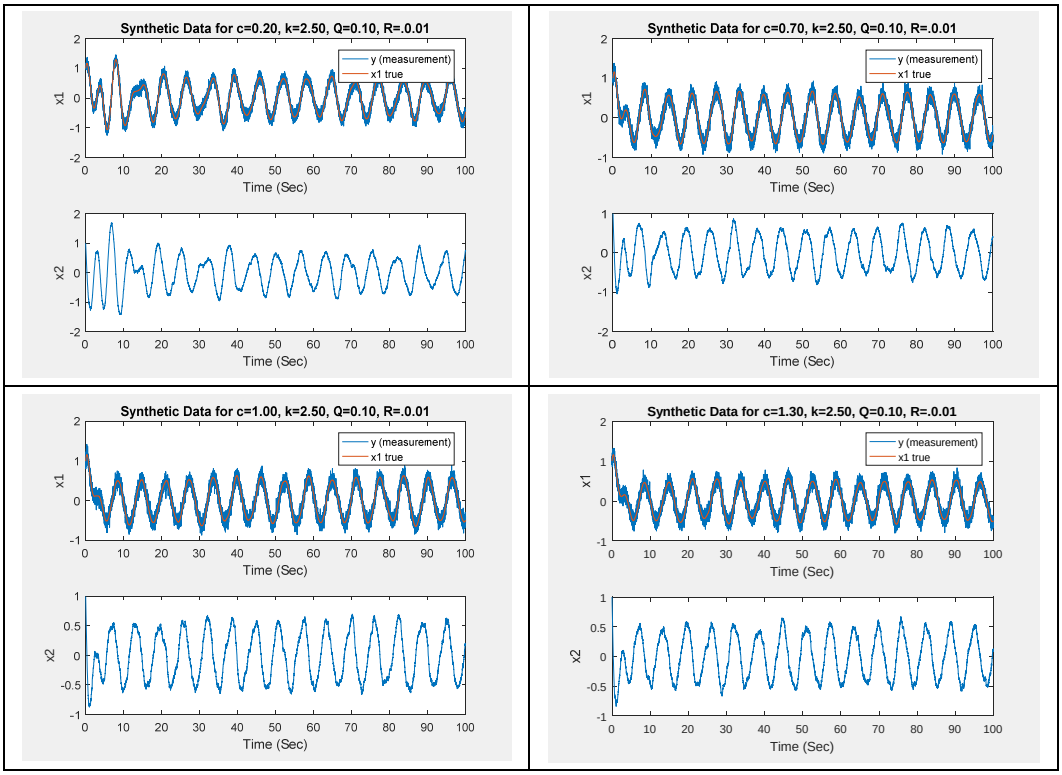


Figure 4-10 Synthetic data

4.4.1 Selected Grid-Based Stratification Results

SGBS was executed with 10 strata per parameter dimension for a total of 100 models. Figure 4-11 through Figure 4-14 represent the respective runs for k=2.5 and $c = \begin{bmatrix} 0.2 & 0.7 & 1.0 & 1.3 \end{bmatrix}$. SGBS performed well for this stratification. The red X marks the points in a simulation where the window was resampled at 20% of its previous size. There are clearly some unnecessary cases of model resampling. The additional convergence criteria to prevent unnecessary resampling will be further explored.

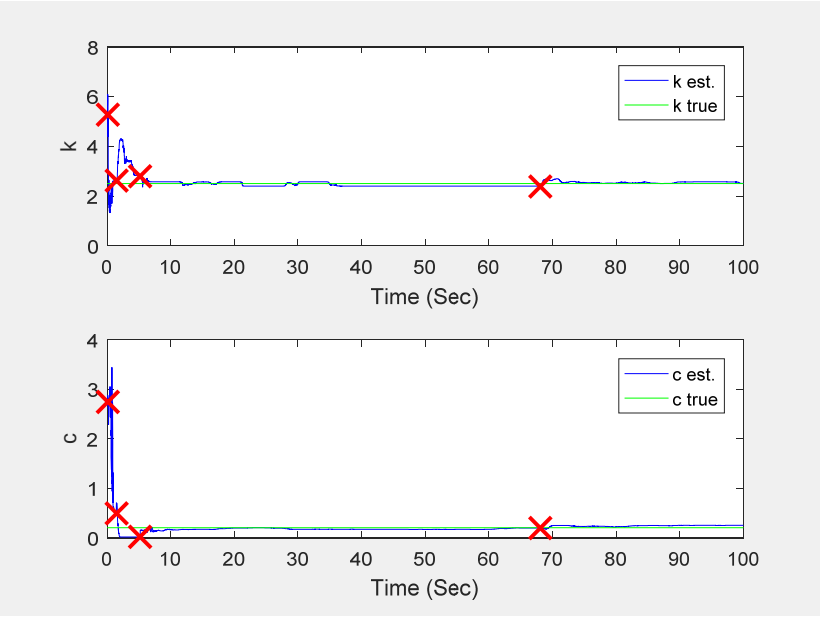


Figure 4-11 SGBS parameter estimates for c=0.20, k=2.5, 100 models

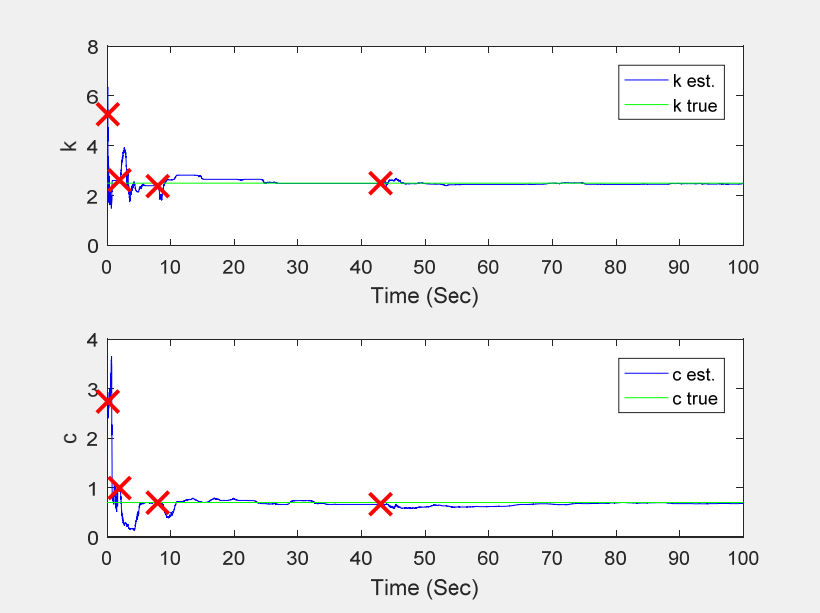


Figure 4-12 SGBS parameter estimates for c=0.70, k=2.5, 100 models

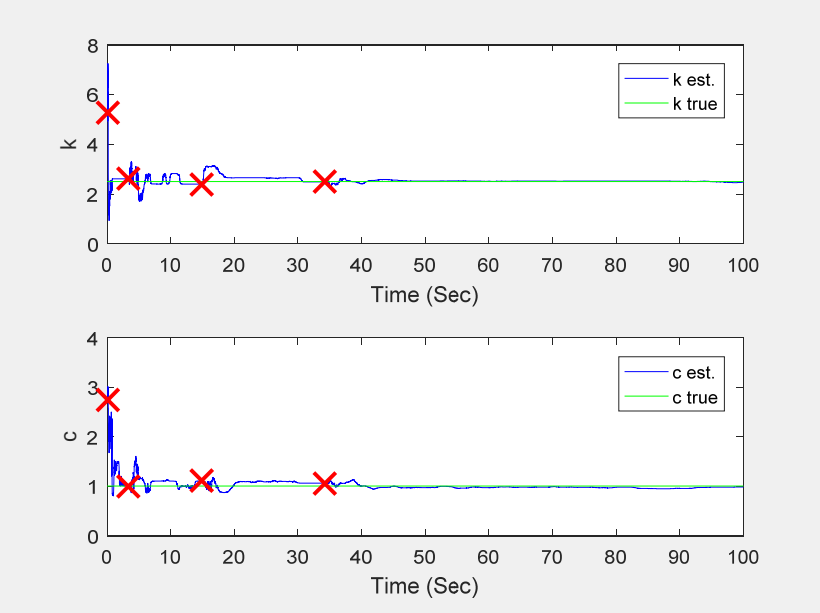


Figure 4-13 SGBS parameter estimates for c=1.00, k=2.5, 100 models

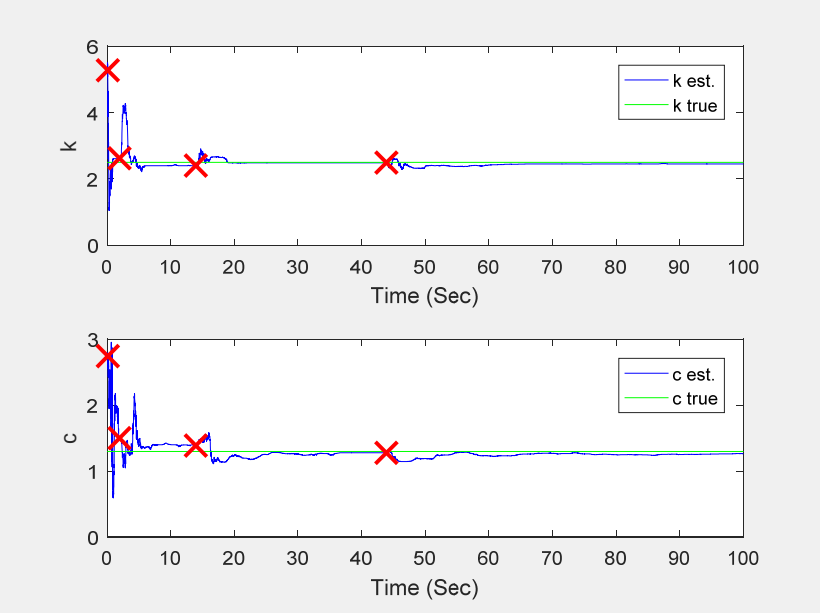


Figure 4-14 SGBS parameter estimates for c=1.30, k=2.5, 100 models

4.4.2 EKF Parameter Estimation using Augmented Parameter Models

For comparison with the SGBS results, a continuous-discrete EKF filter was implemented to estimate the same system by augmenting the states to the system model. While both methods track the parameter well, comparing Figure 4-11 through Figure 4-14 with Figure 4-15 through Figure 4-18 shows that SGBS provides a much smoother result. In some cases, the EKF converges to the region of the actual parameter quicker, but has much more noise present in the measurements for the same process and measurement noise characteristics of the filter.

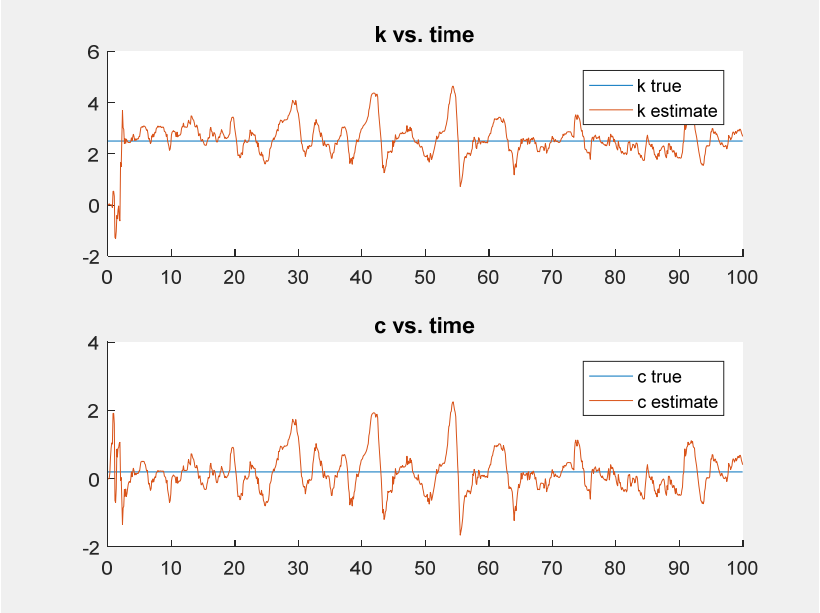


Figure 4-15 EKF parameter estimates for c=0.20, k=2.5

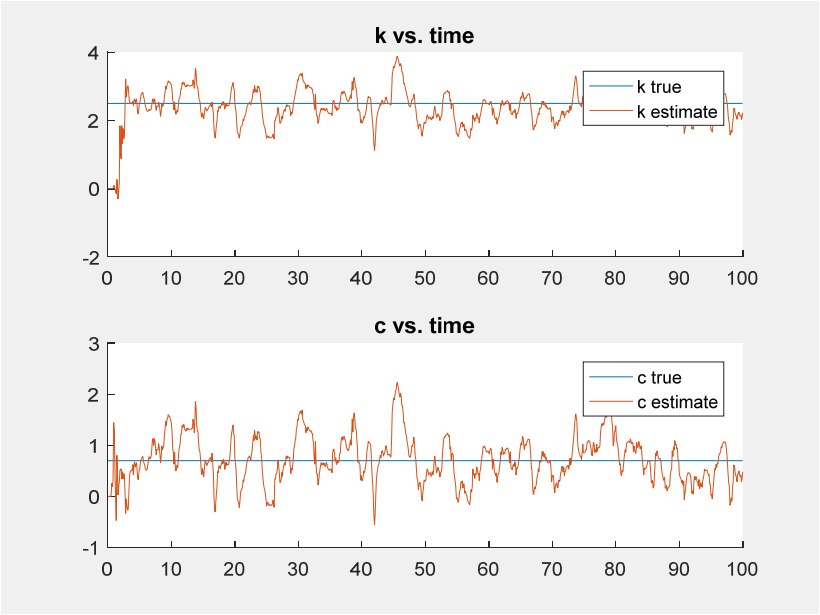


Figure 4-16 EKF parameter estimates for c=0.70, k=2.5

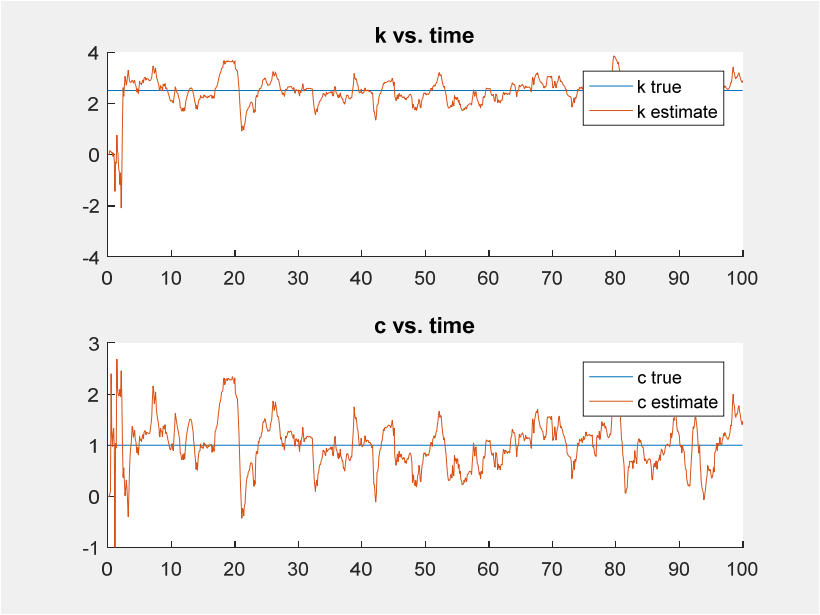


Figure 4-17 EKF parameter estimates c=1.00, k=2.5

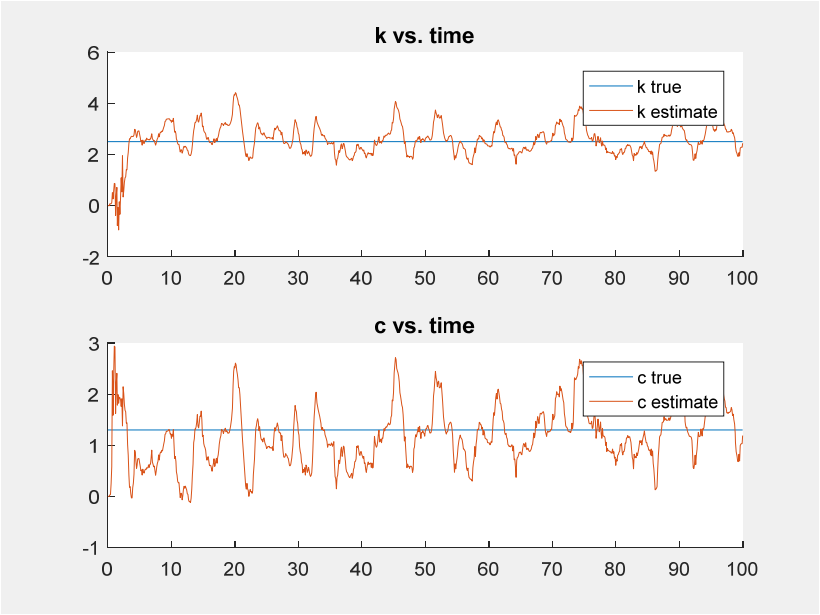


Figure 4-18 EKF parameter estimates c=1.30, k=2.5

4.4.3 Latin Hypercube Sampling Results

The first LHS option (randomly distributed within uniform strata (according to equation (3-9)) was studied in a Monte Carlo fashion. The LHS sample sizes of [20 40 50 60 70 80] were explored with 10 runs each to perform a total of 240 runs. The data was gathered in a MATLAB ".mat" file for further study. The goal of this study is to estimate the mean and variance of the results. Therefore, resampling was turned off. (A single red X marks the first sample point.) Two results from this group of simulations are provided. The first LHS sample run with 20 LHS samples models is shown in Figure 4-19 and Figure 4-20. The second LHS sample run with 50 LHS sample models is illustrated in Figure 4-21 and Figure 4-22. (LHS with resampling is studied in the nonlinear GRAPE framework in Chapters 5 and 6.)

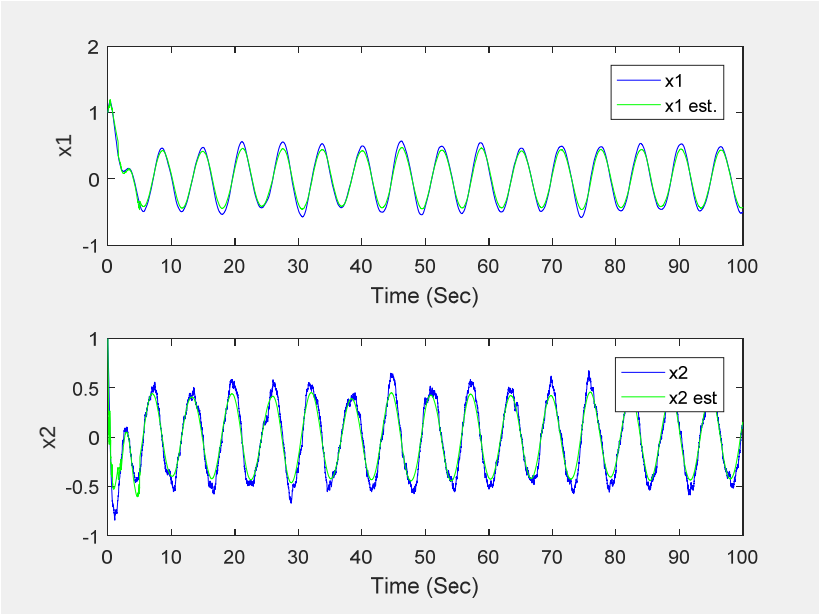


Figure 4-19 LHS state estimate results for c=1.30, k=2.5, 20 models

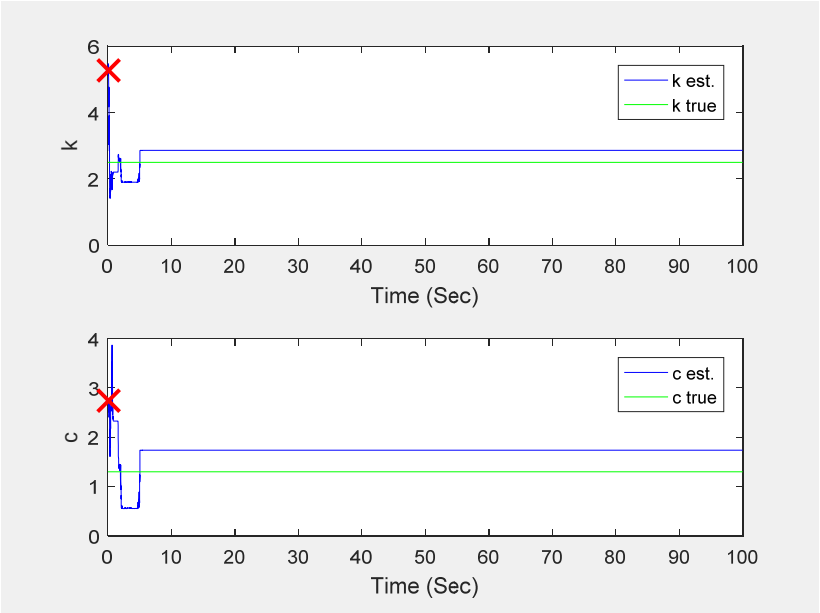


Figure 4-20 LHS parameter estimate results for c=1.30, k=2.5, 20 models

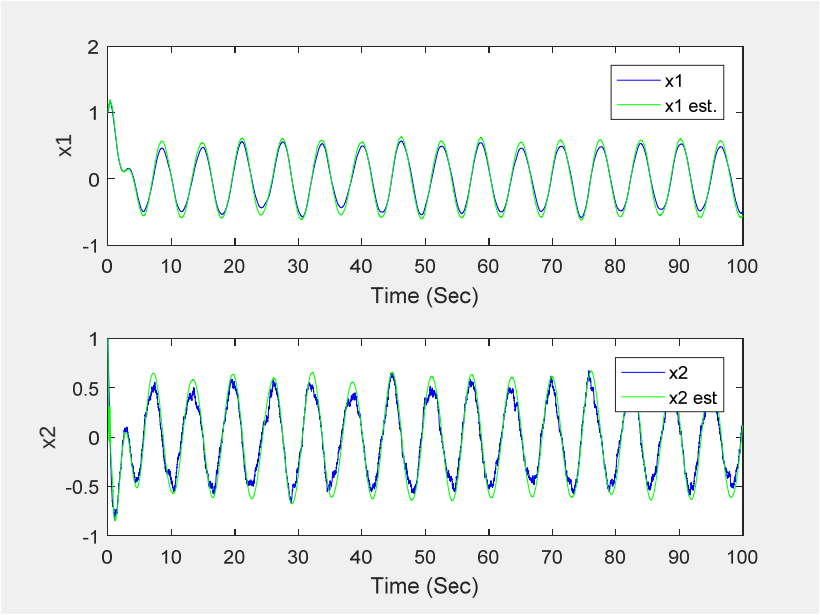


Figure 4-21 LHS state estimate results for c=1.30, k=2.5, 50 models

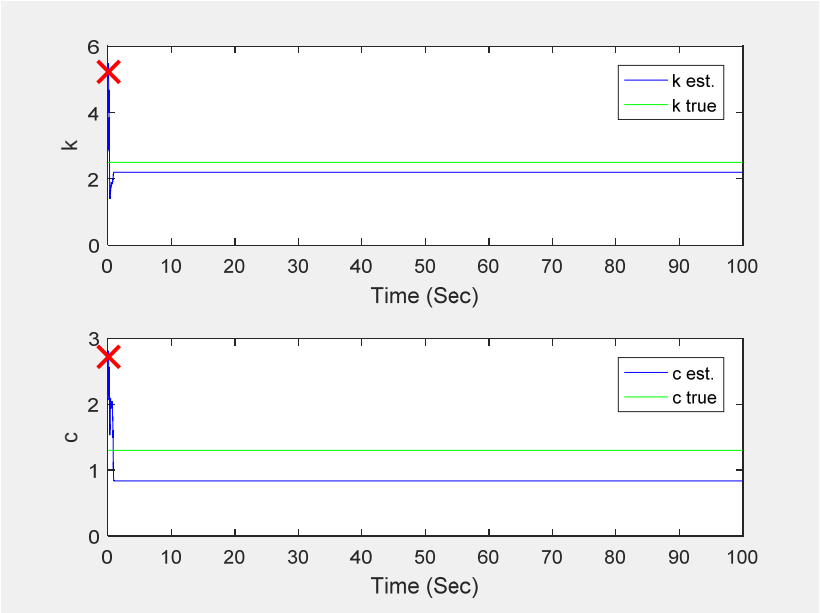


Figure 4-22 LHS parameter estimate results for c=1.30, k=2.5, 50 models

4.4.4 Discussion of Results

The algorithms for both the SGBS and LHS stratification methods have been implemented. Both run well with easily selectable options. Comparing SGBS with the EKF implementation shows that the SGBS method converges to a result and maintains a smooth estimate. Much more noise is present in the EKF estimate of the parameter. There is the possibility of further tuning the EKF to smooth out the parameter estimate, but this has not been explored at the time of this document generation. LHS shows promise from initial sample runs. These initial runs were limited to a Monte Carlo simulation to collect results for evaluating trends in the final estimate. Unfortunately, the samples were not extensive enough to evaluate the mean and variance trends verse the sample size.

CHAPTER 5

Generalized Framework for Nonlinear GRAPE using the Extended Kalman Filter

This chapter covers the extension of GRAPE to a wider range of systems using the continuous-discrete extended Kalman filter (EKF) rather than the Kalman filter. First, the EKF GRAPE algorithm is presented. The FDD approach is then modified for more distinct rules defining each dimensional range for both the SGBS and LHS methods. These procedures enable GRAPE to either narrow its focus on converged values within a sample range or expand the range in the appropriate direction to track parameters outside of the current sample range bounds. Each criterion defines the specific behavior when the MMAE model converges and a resample is considered. New convergence criteria and resampling approaches are presented based on the derivatives of the state estimates. Convergence is determined from a measure of the parameter derivatives using a moving average window. All rules for initial sampling, resampling behavior and convergence criteria are tunable to match the desired behavior goal. Chapter 6 presents the EKF GRAPE algorithm implementation in the context of the Duffing oscillator.

5.1 EKF GRAPE

The EKF allows the exploration of more general nonlinear systems of equations (5-1) through (5-3).

$$\dot{\mathbf{x}}(t) = \mathbf{f}\left(\mathbf{x}^{(i)}(t), \boldsymbol{\theta}^{(i)}(t), u(t), t\right) + \mathbf{G}(t)\mathbf{w}(t), \quad \mathbf{w}(t) \sim N\left(0, \mathbf{Q}^{(i)}(t)\right) \tag{5-1}$$

$$\mathbf{x}_{k+1}^{(i)} = \mathbf{x}^{(i)}\left(t+k\right) \tag{5-2}$$

$$\tilde{\mathbf{y}}_k = \mathbf{h}\left(\mathbf{x}_k^{(i)}\right) + \mathbf{v}_k, \quad \mathbf{v}_k \sim N\left(0, \mathbf{R}_k^{(i)}\right) \tag{5-3}$$

(The integration of equation (5-1) determines the value at the next time step, k+1, for equation (5-2).) Extending MMAE using EKF models has been shown previously in literature by both Crassidis and Junkins' text [53] and Aguiar's review of "sum of Gaussians" filters in [47]. Aguiar provides a summary of several potential approaches beyond just the EKF approach to overcome the limitation of linear MMAE models. Those include "sum of Gaussians" filters using banks of extended Kalman filters, and other related Bayesian-based algorithms such as multiple-hypothesis tracking, bootstrap filters, unscented Kalman filters, particle filters. [47]

This research follows a straightforward extension of GRAPE for nonlinear models. The approach follows the same methodology presented in 2.5. The Kalman filter Models either the discrete-time Kalman filters of Table 2-5 or continuous-discrete EKFs of Table 2-6.

Equations (2-41) through (2-44) show that the EKF models can be used directly for calculating the posterior pdf in equation (2-45). Therefore, equation (2-45) simply replaces step 3.d of Table 4-1. The state estimates are used directly from the individual EKFs**.**

Figure 5-2 and Table 5-1 illustrate a more generalized approach of EKF GRAPE (as opposed to the linear implementations of Figure 4-3 and Table 4-1). The modification of the algorithm is illustrated in the "init. 1" and "3.1 Propagate" steps. The "init. 1" step replaces the linear discrete-time models with a generalized nonlinear set of models. These models can represent the linear systems; however, the implementation uses a variable step 4th order Runge-Kutta (RK4) differential equation solver approach rather than just looping through the models using matrix multiplication. Therefore, there is a significant cost in computational complexity that can be avoided with linear systems by using discrete-time linear models.

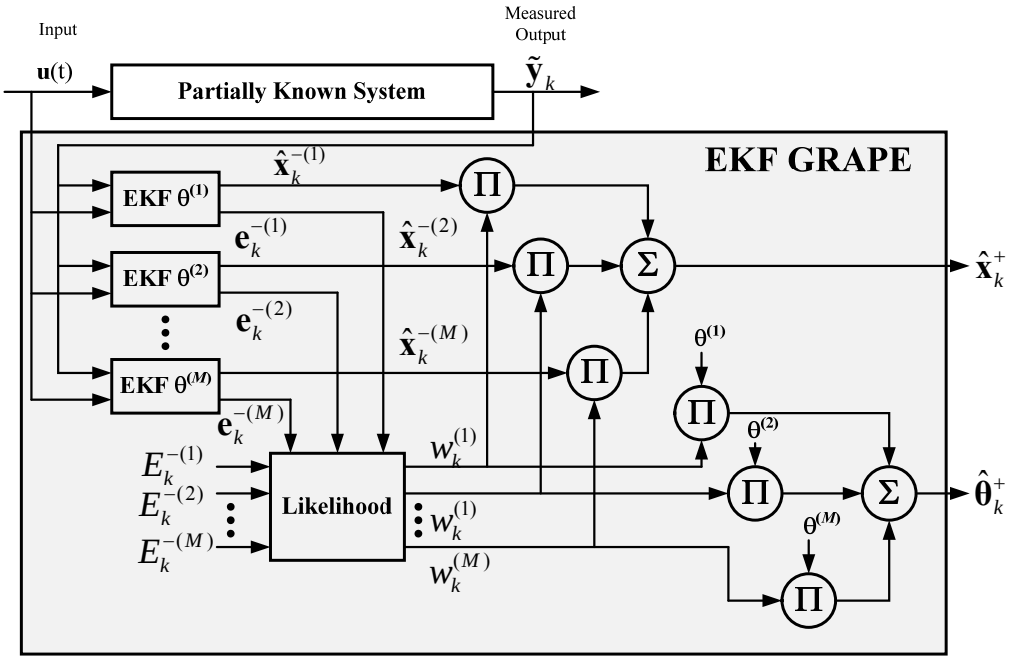


Figure 5-1 Nonlinear GRAPE high-level summary

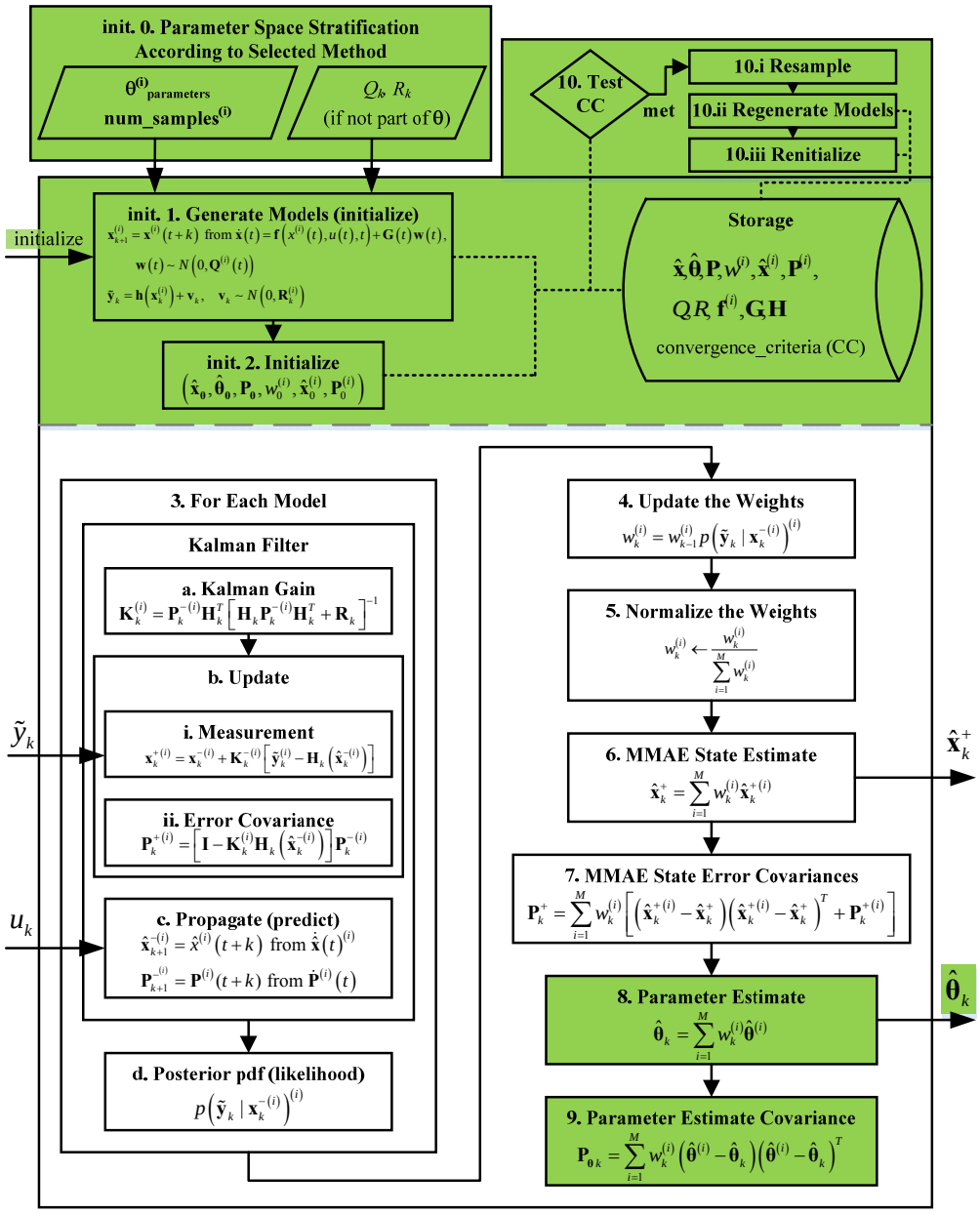


Figure 5-2 GRAPE EKF MMAE high-level summary

Table 5-1 GRAPE EKF algorithm detailed steps

0. Parameter Space Stratification According to Selected Method
    a. SGBS
    b. LHS
        i. randomly distributed within uniform strata
        ii. centered within uniform strata
        iii. randomly distributed within normal (Gaussian) strata
        iv. centered within normal (Gaussian) strata
1. Generate models $\left(\boldsymbol{\Phi}^{(i)}\right)$ to represent parameter sets of interest $\left(\boldsymbol{\theta}^{(i)} \in \boldsymbol{\theta}\right)$

    $\mathbf{x}_{k+1}^{(i)} = \mathbf{x}^{(i)}\left(t+k\right)$ from

    $$\dot{\mathbf{x}}\left(t\right) = \mathbf{f}\left(x^{(i)}\left(t\right), u\left(t\right), t\right) + \mathbf{G}\left(t\right)\mathbf{w}\left(t\right), \quad \mathbf{w}\left(t\right) \sim N\left(0, \mathbf{Q}^{(i)}\left(t\right)\right)$$

    $$\tilde{\mathbf{y}}_k = \mathbf{h}\left(\mathbf{x}_k^{(i)}\right) + \mathbf{v}_k, \quad \mathbf{v}_k \sim N\left(0, \mathbf{R}_k^{(i)}\right)$$

2. Initialize, initial values for:
    a. MMAE state estimate $\left(\hat{\mathbf{x}}_{\mathbf{0}}\right)$
    b. MMAE parameter estimates $\left(\hat{\boldsymbol{\theta}}_{\mathbf{0}}\right)$
    c. Parameter estimate covariances $\left(\mathbf{P}_{\mathbf{0}} = E\left\{\hat{\mathbf{x}}_{\mathbf{0}}\hat{\mathbf{x}}_{\mathbf{0}}{}^{T}\right\}\right)$
    d. KF model state estimates $\left(\hat{\mathbf{x}}_0^{(i)}\right)$
    e. KF model covariances $\left(\mathbf{P}_0^{(i)} = E\left\{\hat{\mathbf{x}}_0^{(i)}\hat{\mathbf{x}}_0^{(i)T}\right\}\right)$
    f. Initialize filter weights $w_0^{(i)} = 1/M$
3. For each model:
    a. Calculate the Kalman Gain

    $$\mathbf{K}_k^{(i)} = \mathbf{P}_k^{-(i)}\mathbf{H}_k^T\left[\mathbf{H}_k\mathbf{P}_k^{-(i)}\mathbf{H}_k^T + \mathbf{R}_k\right]^{-1}$$

    b. Update
        i. Perform the measurement update

        $$\mathbf{x}_k^{+(i)} = \mathbf{x}_k^{-(i)} + \mathbf{K}_k^{-(i)}\left[\tilde{\mathbf{y}}_k^{(i)} - \mathbf{H}_k\hat{\mathbf{x}}_k^{(i)}\right]$$

        ii. Update the error covariance

        $$\mathbf{P}_k^{+(i)} = \left[\mathbf{I} - \mathbf{K}_k^{(i)}\mathbf{H}_k\right]\mathbf{P}_k^{-(i)}$$

Table 5-1 GRAPE EKF algorithm detailed steps (continued)

c. Propagate the next state (predict)

$$\dot{\hat{\mathbf{x}}}(t)^{(i)} = \mathbf{f}\left(x^{(i)}(t), u(t), t\right) + \mathbf{G}(t)\mathbf{w}(t) \Rightarrow \hat{\mathbf{x}}_{k+1}^{-(i)} = \hat{x}^{(i)}(t+k)$$

$$\dot{\mathbf{P}}^{(i)}(t) = \mathbf{F}^{(i)}(t)\mathbf{P}^{(i)}(t) + \mathbf{P}^{(i)}(t)\mathbf{F}^{(i)T}(t) + \mathbf{G}(t)\mathbf{Q}(t)\mathbf{G}^{T}(t)$$

$$\Rightarrow \mathbf{P}_{k+1}^{-(i)} = \mathbf{P}^{(i)}(t+k)$$

$$\mathbf{F}^{(i)}(t) \equiv \left.\frac{\partial \mathbf{f}^{(i)}}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}(t), u(t)}$$

d. Calculate the posterior pdf (likelihood)

$$p\left(\tilde{\mathbf{y}}_k \mid \mathbf{x}_k^{-(i)}\right)^{(i)} = \frac{1}{\left[\det\left(2\pi \mathbf{E}_k^{-(i)}\right)\right]^{1/2}} \exp\left\{-\frac{1}{2}\mathbf{e}_k^{-(i)T}\left(\mathbf{E}_k^{-(i)}\right)^{-1}\mathbf{e}_k^{-(i)}\right\}$$

where

$e_k^{-(i)}$ is the measurement residual and measurement covariance is

$$\mathbf{E}_k^{-(i)} \equiv E\left\{\mathbf{e}_k^{-(i)}\mathbf{e}_k^{-(i)T}\right\} = H_k^{(i)}P_k^{-(i)}H_k^{(i)T} + R_k^{(i)}$$

4. Update the weights

$$w_k^{(i)} = w_{k-1}^{(i)} p\left(\tilde{\mathbf{y}}_k \mid \mathbf{x}_k^{-(i)}\right)^{(i)}$$

5. Normalize the weights

$$w_k^{(i)} \leftarrow \frac{w_k^{(i)}}{\sum_{i=1}^{M} w_k^{(i)}}$$

6. Calculate MMAE State Estimate

$$\hat{\mathbf{x}}_k^{+} = \sum_{i=1}^{M} w_k^{(i)} \hat{\mathbf{x}}_k^{+(i)}$$

7. Calculate the MMAE State Error Covariances

$$\mathbf{P}_k^{+} = \sum_{i=1}^{M} w_k^{(i)} \left[\left(\hat{\mathbf{x}}_k^{+(i)} - \hat{\mathbf{x}}_k^{+}\right)\left(\hat{\mathbf{x}}_k^{+(i)} - \hat{\mathbf{x}}_k^{+}\right)^{T} + \mathbf{P}_k^{+(i)}\right]$$

Table 5-1 GRAPE EKF algorithm detailed steps (continued)

8. Calculate the Parameter Estimate

$$\hat{\boldsymbol{\theta}}_k = \sum_{i=1}^{M} w_k^{(i)} \hat{\boldsymbol{\theta}}^{(i)}$$

9. Calculate the Parameter Estimate Error Covariance,

$$\mathbf{P}_{\boldsymbol{\theta}\, k} = \sum_{i=1}^{M} w_k^{(i)} \left(\hat{\boldsymbol{\theta}}^{(i)} - \hat{\boldsymbol{\theta}}_k \right) \left(\hat{\boldsymbol{\theta}}^{(i)} - \hat{\boldsymbol{\theta}}_k \right)^T$$

10. Check the Convergence Criteria

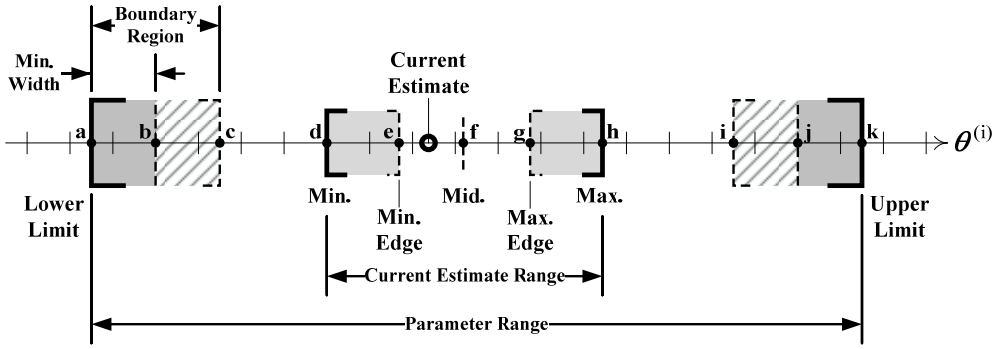
11. Repeat 3 through 10 until the measurements are depleted

5.2 FDD Approach for EKF GRAPE

For the EKF GRAPE approach for FDD, each dimension is handled separately in a common parameter range method. Range values are initialized prior to executing the algorithm. The parameter range options are provided in Table 5-2. These settings provide both the initial values and the range settings. Resampling behavior depends on the location of the current estimate along with the parameter range settings. Figure 5-3 provides an example of a assumed parameter range with its initial range estimate and range settings. The current estimate marks the GRAPE estimate for the parameter.

The parameter range covers the valid parameter values. The current estimate range reflects the boundary of the sample region covered by the GRAPE models. The current estimate range is a subset of the parameter range. The current estimate range can be as large as the entire parameter range or as small as the minimum width. Resampling is triggered with a resample flag. The current

estimate range will change according to the location of the current estimate and the predefined options that set the minimum width on both ends of the parameter range along with the minimum and maximum edge values within the current estimate range. A parameter space is represented by a group of parameter ranges representing each unknown parameter. Figure 5-4 illustrates a three-dimensional parameter space for the parameters $\theta^{(1)}$, $\theta^{(2)}$ and $\theta^{(3)}$. Each parameter characteristic is independently defined.



Range Interval	Description
[a,k]	parameters space limits
[a,b]	minimum width at the lower limit
[a,c] and [i,k]	boundary width
[b,j]	region of normal operation
[d,h]	current estimate range
[d,e]	lower edge range
[e,g]	center range
[g,h]	upper edge range
[j,k]	minimum width at the upper limit

Figure 5-3 GRAPE parameter range regions

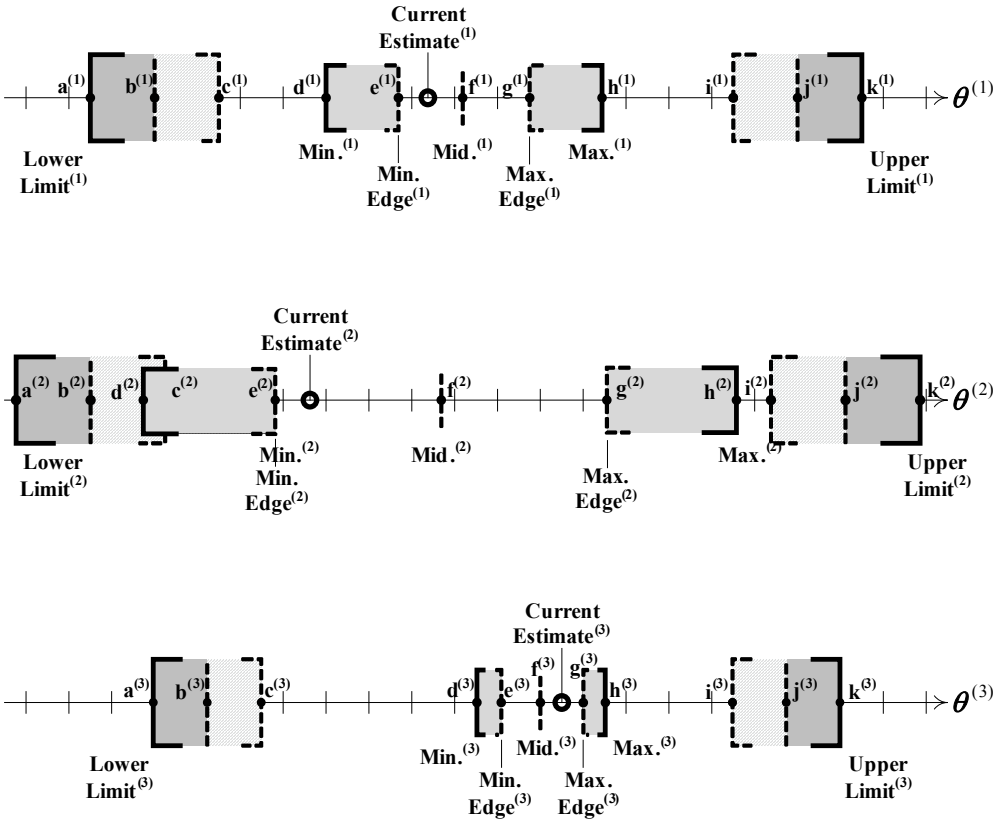


Figure 5-4 GRAPE range regions for 3-dimensional parameter space

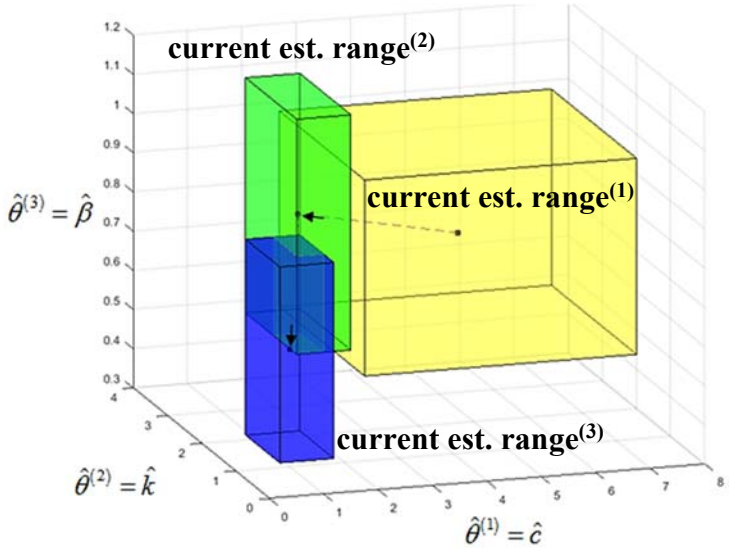


Figure 5-5 GRAPE range regions for 3-dimensional parameter space

5.2.1 Resample Behavior within Center Range

In general, convergence within the center range will reduce the current range estimate by multiplying its width by the "resample.reduce" multiplier. This is illustrated in Figure 5-6.a. The resampled estimate range has reduced and is centered on the current estimate. This reduction reflects normal narrowing of the tracking window. When a resample is initiated near a limit, the width may be reduced by the proximity to closest limit. Figure 5-7.a illustrates a width reduction near the lower limit of the parameter range. The resample of Figure 5-7.a would normally expand the window size because the current estimate is below the minimum edge value. However, it is near the lower limit so both a center range and edge range resample will behave the same here.

Figure 5-6.b illustrates an expanding range resample. The current estimate is below the minimum edge; therefore, the range is expanded by the resample.enlarge multiplier. Again, the expansion will be adjusted when the estimate is near the parameter limits.

Table 5-2 GRAPE range parameter options

Parameter	Description	Point/Interval on Figure 5-3
sample_type	**'SGBS'** or **'LHS'**	N/A
uniform_samples	number of SGBS samples (defined for each dimension)	N/A
limit.lower	lower limit boundary of parameter range	a
limit.upper	upper limit boundary of parameter range	k
limit.smallest_percent	smallest width as percent of overall parameter range	[a,b] and [j,k]
range.min	initial value set, resample values are calculated as (**range.mid-range.width/2**); may be adjusted when near lower limit	d
range.max	initial value set, resample values are calculated as (**range.mid+range.width/2**) may be adjusted when near upper limit	h
range.start_time	time when current (re)sample began	N/A
range.sample_index	count of (re)samples	N/A
resample.reduce	multiplier for resampling within center range (normally between .2 and .8)	N/A
resample.edge	percent of window width minimum and maximum border where next parameter resample width is enlarged rather than reduced	determines e and g
resample.enlarge	multiplier for resampling within lower and upper edge ranges	N/A

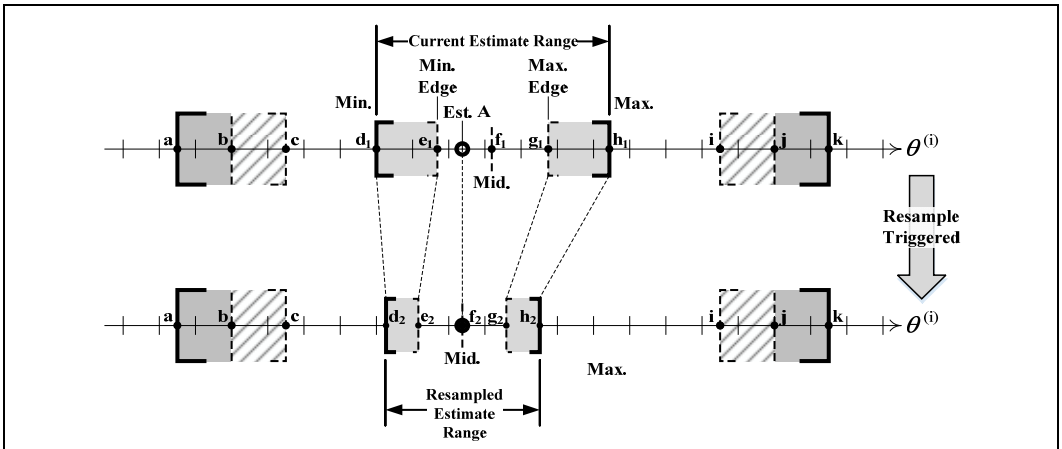
Table 5-3 GRAPE range parameter calculated values

Calculated Parameter	Description	Point/Interval on Figure 5-3
num_models	total number of samples (**LHS_num** or product of each dimensional **uniform_samples** for SGBS)	N/A
limit.width	**limit.upper - limit.lower**	[d,h]
limit.smallest_width	**limit.width * limit.smallest_percent**	[a,b] and [j,k]
range.start_time	recorded time when current (re)sample began	N/A
range.sample_index	index counter of resamples (initialized to 1 on the first sample)	N/A
range.width	initially calculated as **range.max + range.min**; resample increases or decreases base on the location of the current sample	[d,h]
range.mid	initially calculated as **(range.max + range.min)/2;** starts as current parameter estimate at resample; may be adjusted based on location within parameter range when near lower and upper limits	f
range.min_edge	**range.min + resample.edge*range.width**	e
range.max_edge	**range.max - resample.edge*range.width**	g

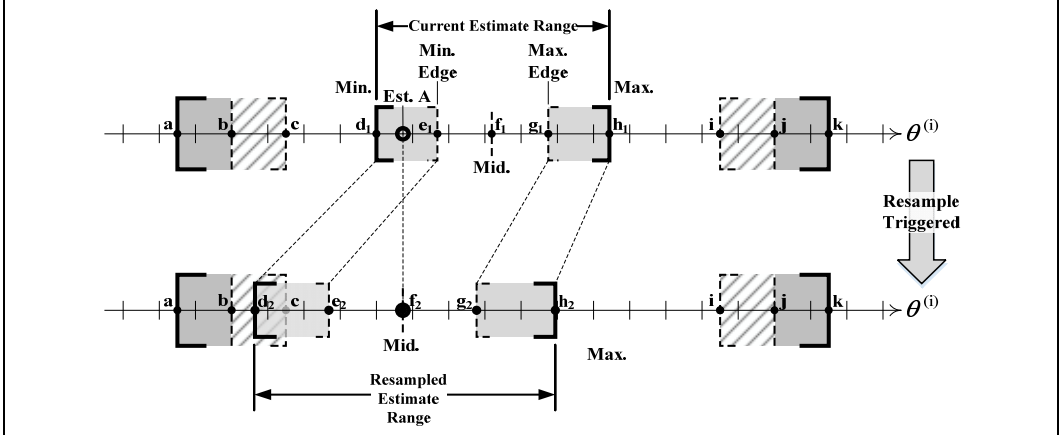
5.2.2 Resample Behavior near Parameter Limits

As mentioned in 5.2.1, the resampling behavior is modified when the current estimate is near the range limits. In cases where the current estimate is within a region approaching the boundary limit, the distance from the boundary limit defines the new current estimate range width. This is illustrated in Figure 5-7.a. The estimate range width is now a maximum of twice the distance to the lower limit. For this case, the current estimate remains the center of the new current estimate range. When the current estimate is within the minimum estimate width

of a range limit, the resampled current estimate width is set to twice the minimum width value. This was an implementation choice to minimize the chance of getting trapped near a limit. Near the limit, the range width cannot get smaller. Additionally, the center is always at the minimum width from the edge, because the center of the range width near the limit cannot move any closer to the limit. However, the center can move away, and the range width can grow as it does.

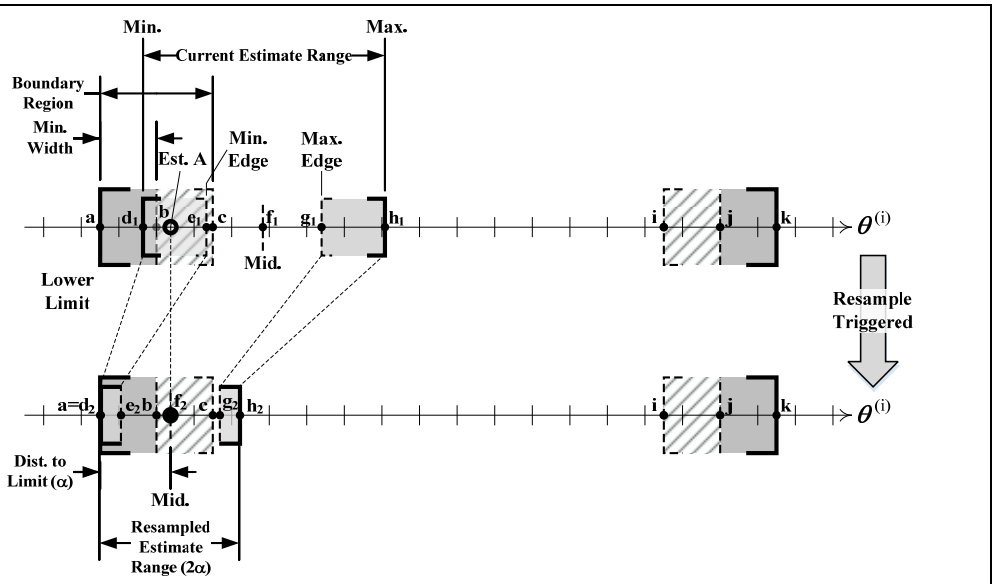


a. reduction resample

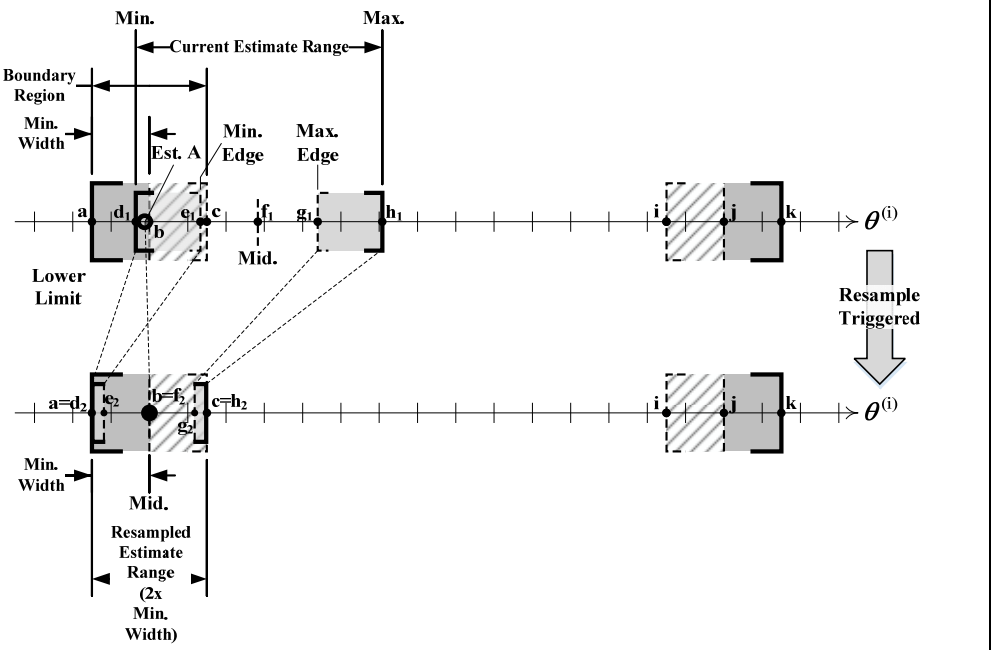


b. expansion resample
(similar behavior when current estimate is within the upper edge range)

Figure 5-6 Resampling in region of normal operation



a. resample near limit
(distance to limit (α) defines the new estimate range (2α))



b. resample within minimum width of limit
(new estimate range set to 2 times minimum width)

Figure 5-7 Lower boundary region resample (upper is mirrored)

5.3 Updated Convergence Determination Approach

Previously in the linear approach, convergence was measured from a mean square error of the MMAE weights. Essentially, when the weights converged, then the system was considered converged. There was difficulty in finding an appropriate value to set as a threshold. The EKF GRAPE method updated the convergence criteria to track the derivative of the parameter estimates. (This update may also be applied to the linear GRAPE approach.)

5.3.1 Simple Moving Average of Parameter Estimate Derivative

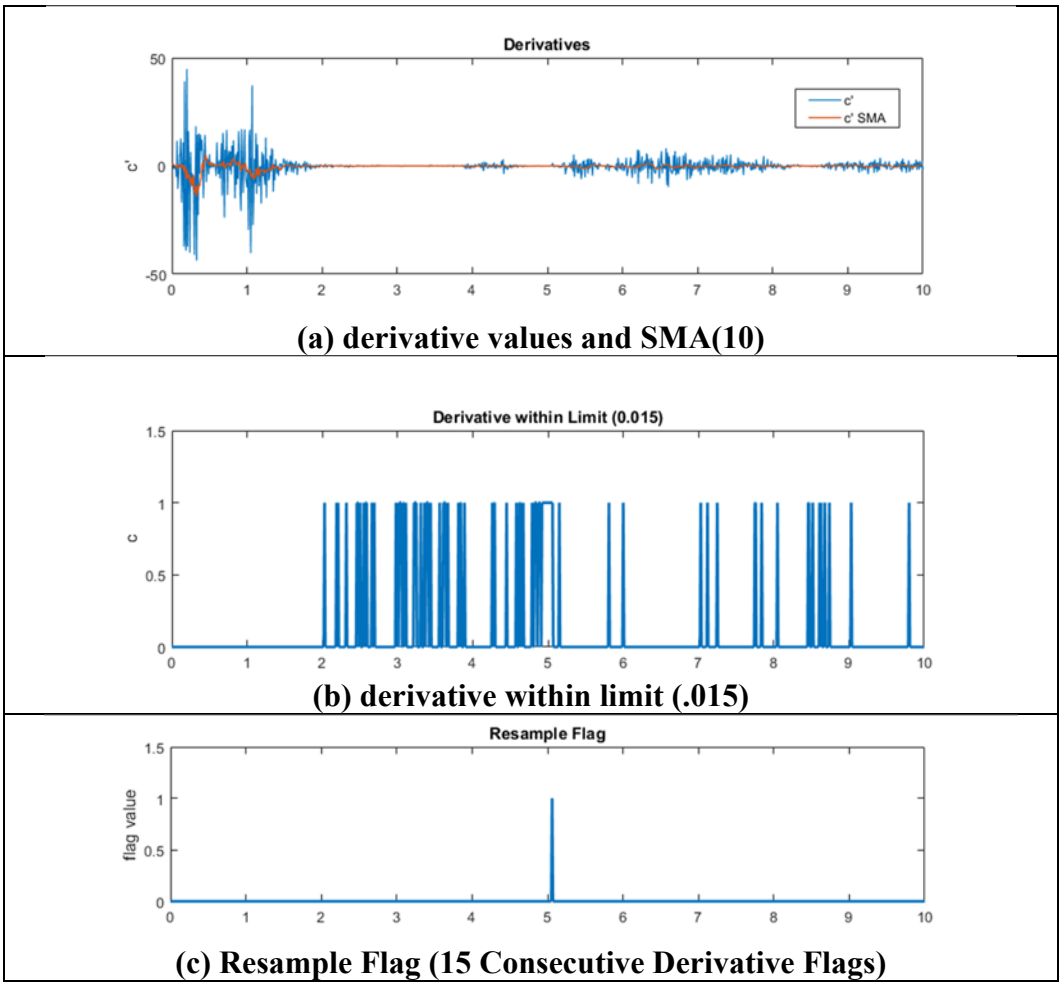
Each parameter estimate is tracked. The backward difference method using prior two parameter estimates, equation (5-4), is used to estimate the derivatives.

$$\dot{\hat{\theta}}_k^{(i)} \simeq \frac{3\hat{\theta}_k^{(i)} - 4\hat{\theta}_{k-1}^{(i)} + \hat{\theta}_{k-2}^{(i)}}{2\Delta t} \tag{5-4}$$

(This approach is derived from combining the Taylor series expansion evaluated at the prior time step *k*-1 and the one before that at *k*-2 and solving for the first derivative ignoring terms higher than the second order derivative. It is equivalent to the backward finite-divided-difference formula in Figure 23-2 of [126].)

The derivative measurements can be dominated by noise. Figure 5-8 illustrates how the noisy data is converted into a threshold flag. Figure 5-8(b) shows the threshold flags being consistently set when using the noisy derivative data of Figure 5-8(a). Therefore, the measurements are smoothed using a simple

moving average (SMA) window. The SMA approach is shown in equation (5-5). Figure 5-8(c) illustrates the SMA window using 10 previous samples.



(a) derivative values and SMA(10)

(b) derivative within limit (.015)

(c) Resample Flag (15 Consecutive Derivative Flags)

Figure 5-8 Noisy derivative to resample flag

$$SMA_k\left(y_k, n\right) = \begin{cases} y_k & if\ k = 1 \\ \dfrac{SMA_{k-1} \cdot \left(k-1\right) + y_k}{k} & if\ 1 < k \le n \\ SMA_{k-1} + \dfrac{y_k}{n} - \dfrac{y_{k-n}}{n} & if\ k > n \end{cases} \tag{5-5}$$

$$SMA_k\left(\dot{\hat{\theta}}_k^{(i)}, n\right) \tag{5-6}$$

5.3.2 Criteria for Convergence

Table 5-4 provides the convergence criteria options along with a summary the stored and calculated values. The SMA derivative flag is set when a specified number of prior SMA derivative values are less than a threshold value. Figure 5-9 illustrates the case for where the derivatives of three parameters are tracked. Their flag values are individually set based on the convergence criteria. The resample flag is set to one to trigger a resample when the convergence rule is met. In this case, the convergence rule is represents when any individual SMA derivative has converged.

Table 5-4 GRAPE convergence options and calculated values

Options	
Parameter	**Description**
`SMA_num`	number of past samples used for Simple Moving Average (SMA) calculation
`SMA_flag_limit`	number of past SMA values ≤ **`derv_limit`** required for a true flag (1)
`derv_limit`	limit on the derivative (slope) of the parameter estimate
`derv_limit_redux`	multiplier for decreasing **`derv_limit`** each time a new sample is taken (set to 1 for no reduction); used to make it more difficult to resample each time a new sample is taken
`delay_resample`	delay resample a minimum of this value in hundredths of a second (set to zero for no delay)
`resample_rule`	currently there are 3 resample rules: 1 = **`SMA_derv_flag`** set for at least one dimension 2 = **`SMA_derv_flag`** set for at least two dimensions 3 = **`SMA_derv_flag`** set for at least three dimensions
`forced_resample`	forces resample at fixed time interval in hundredths of a second (set to a large value for no effect); used for debugging purposes only

Table 5-4 GRAPE convergence options and calculated values (continued)

Calculated Values	
Parameter	**Description**
`derv`	derivative calculated from equation (5-4)
`SMA_derv`	simple moving average derivative value calculated from equation (5-5) for each parameter dimension
`SMA_derv_within_limit`	set to 1 if **`SMA_derv ≤ derv_limit`**, otherwise 0; stored for each parameter dimension
`SMA_derv_flag`	true if **`SMA_flag_limit`** is met on the current dimension, stored for each parameter dimension
`resample_flag`	true if passes **`resample_rule`**

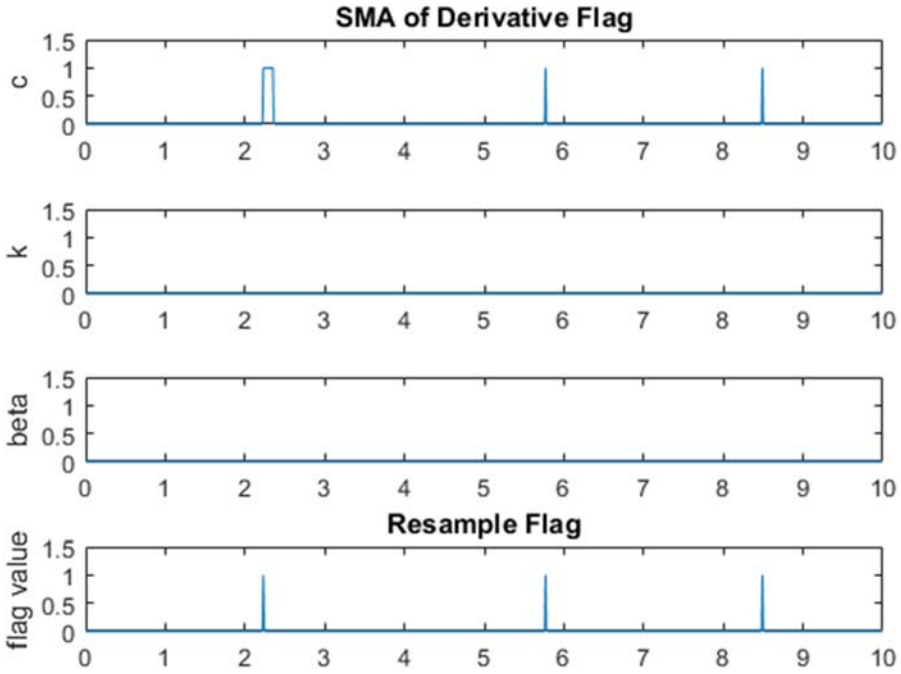


Figure 5-9 Example using resample rule 1

5.3.3 Resample Rules

Individual parameter estimate convergence criteria are combined to trigger a resample based on the resample rule selected. A delay modifier is added to set a

minimum resampling interval. This modifier is the "delay_resample" value which prevents resampling in increments of sample steps. For the current implementation, the samples are taken every one-hundredth of a second.

There were three resample rules implemented in this research. Rule 1 triggers a resample with any parameter estimate convergence. Rules 2 and 3 require parameter estimate convergence for a minimum of two and three parameter dimensions before resampling is triggered. Each rule also enforces the minimum delay to prevent resampling to soon after the prior resample.

Rule 1 is the default rule used. It works well when a single parameter tends to converge faster than the others. Additionally, this rule has shown good results when the initial current range estimate for a parameter is outside the actual value. The current estimate of that parameter will converge towards the edge of the current range estimate and trigger an expansion of the current range estimate centered on the value. This moves it towards the actual parameter value. Using, resample rules 2 or 3 will needlessly cause a delay in the system convergence as the actual parameter is not within the current parameter samples. Testing has shown that rule 1 combined with an appropriate delay typically works well. (Specific results covering the Duffing Oscillator are covered in Chapter 6.) An alternate rule 4 was explored but not implemented. Rule 4 was defined to explore the total derivative magnitude from equation (5-7).

$$SMA\left(\dot{\theta}_k^{total}, n\right) = \sqrt{\sum_{i=1}^{n} SMA_k\left(\dot{\theta}_k^{(i)}, n\right)^2} \tag{5-7}$$

5.4 Closing Remarks on EKF GRAPE

The EKF framework formalizes the parameter tracking process. These additions to the GRAPE framework are also applicable to the linear implementation. The updated procedures enable GRAPE to either narrow the focus to converged values within a parameter range or expand the range in the appropriate direction to track parameters outside the current parameter range boundary. Each rule defines the specific behavior when the MMAE model converges. Convergence is determined from the parameter derivatives using a simple moving average window to filter the noise. The system is tunable to match the desired performance goal for initial sampling, resampling and convergence criteria. Chapter 6 explores the EKF framework using the Duffing oscillator.

CHAPTER 6

Exploring the Duffing Oscillator using GRAPE

This chapter applies the EKF GRAPE framework to the Duffing oscillator. There are many nonlinear oscillatory systems to consider for study. The Duffing oscillator is commonly studied oscillator that adds a cubic stiffness term to the simple harmonic oscillator. The Duffing oscillator was chosen because it has common mechanical analogies and is similar in form to the harmonic oscillator previously presented in Chapter 2 and simulated in section 4.4 using the spring-mass-damper mechanical representation. This chapter provides an overview of the duffing oscillator with some relevant research. The specific form of the Duffing equation is presented along with the derivation of EKF filter model. A specific model is chosen from a wide range of coefficient possibilities. This model is studied through the EKF GRAPE framework. The results are discussed, and the justification is provided for the general GRAPE tuning statements of Chapter 5.

6.1 Overview of the Duffing Oscillator

The Duffing oscillator is a typical system used to study nonlinear dynamics. It was popularized and received its name from Georg Duffing a German engineer during the early twentieth century. He produced several publications included one text [127] on his equation. The Duffing oscillator has been used to model the following physical processes: stiffening springs, beam buckling, nonlinear

electronic circuits along with several other applications in superconducting amplifiers and ionization waves in plasma.[128, 129]

Kovacic and Brenna provide a thorough overview of the historical evolution the Duffing equation in Chapter 1 of [129]. The Duffing oscillator received sporadic focus until 1970. Since then, journal paper titles including "Duffing" have steadily grown from 8 to 212 in 2009 (see Figure 1.4, pg 15 of [129]).

The Duffing oscillator has come to describe an entire group of equations that add a cubic nonlinear stiffness term to the standard linear second-order harmonic differential equation. Duffing's original system of study was the second order vibration system that included linear viscous dampening, either free or forced harmonic vibration along with quadratic and cubic force terms.[129] Today, the most common form of the Duffing equation is described by the following differential equation with the cubic stiffness term.

$$\ddot{x} + \delta\dot{x} + \alpha x + \beta x^3 = \gamma\cos(\omega t) \tag{6-1}$$

where the constants are defined as:

- γ is the driving force amplitude,
- ω is the driving force frequency,
- δ is the damping, α is stiffness and
- β is nonlinearity in the restoring force.

The system simplifies to a spring-mass-damper if β is zero. The appearance of βx^3 changes the behavior of the system entirely. A paraphrase of effects of βx^3 is summarized below from [128]:

- an analytical solution is no longer available
- superposition principle is no longer valid (linear combination of solutions does not provide a solution)
- the equation has several qualitative behaviors related to limit cycles, sensitivity to initial conditions, strange attractor, fractal structures, bifurcation phenomena (left to the reader to further explore in [128])

6.2 Chosen form of Duffing Oscillator

The simulations focus will be on small β ($0 \leq \beta \leq 1$) as the system becomes chaotic when β>1.

6.3 EKF Model of Duffing Oscillator

The chosen form of the Duffing oscillator is provided below,

$$m\ddot{x} + c\dot{x} + kx + \beta x^3 = \gamma \sin(\omega t) \tag{6-2}$$

where m = mass, c = viscous damping, k is linear stiffness and β is the cubic stiffness term. For purposes later shown, the input is represented as a general function, $f(\omega,t)$, of frequency, ω, and time, t.

$$m\ddot{x} + c\dot{x} + kx + \beta x^3 = \gamma \cdot f(\omega,t) \tag{6-3}$$

Dividing both sides of equation (6-3) by m provides the form used to derive the EKF models.

$$\ddot{x} + (c/m)\dot{x} + (k/m)x + (\beta/m)x^3 = (\gamma/m) \cdot f(\omega,t) \tag{6-4}$$

The states are chosen as $x_1 = x$ and $x_2 = \dot{x}$; therefore, the state space representation is

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -(c/m)x_2 - (k/m)x_1 - (\beta/m)x_1^3 \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{\gamma}{m} \end{bmatrix} f(\omega, t) \tag{6-5}$$

with measurement of

$$\tilde{y} = \begin{bmatrix} 1 & 0 \end{bmatrix} x = x_1 \tag{6-6}$$

Equations f and h are defined as

$$f(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \end{bmatrix} = \begin{bmatrix} x_2 \\ -(c/m)x_2 - (k/m)x_1 - (\beta/m)x_1^3 \end{bmatrix} \tag{6-7}$$

$$h(x) = x_1 \tag{6-8}$$

Per the continuous-discrete Extended Kalman Filter, Table 2-6, the Jacobians are

$$F = \frac{\partial f}{\partial x} = \begin{bmatrix} \dfrac{\partial f}{\partial x_1} & \dfrac{\partial f}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -(k/m) - 3(\beta/m)x_1^2 & -(c/m) \end{bmatrix} \tag{6-9}$$

$$H = \frac{\partial h}{\partial x} = \begin{bmatrix} \dfrac{\partial h}{\partial x_1} & \dfrac{\partial h}{\partial x_2} \end{bmatrix} = \begin{bmatrix} 1 & 0 \end{bmatrix} = \text{const.} \tag{6-10}$$

6.4 Range of Values Considered

A general assumption is the mass m is 1. Therefore, there are three coefficients of concern. The EKF GRAPE framework is set up such that $\theta^{(1)} = c$, $\theta^{(2)} = k$ and $\theta^{(3)} = \beta$. The value of β *is* chosen to be less than one to prevent chaotic behavior. With the constraint n β, further constraints must be considered for c and

k to ensure the system to displays behavior that changes when any of the parameters.

Both k and β are stiffness coefficients in equation (6-3). Figure 6-1 illustrates that ranges of $0.1 < k \leq 1.0$ and $0 \leq \beta < 1.0$ are reasonable to consider when the desired effect is different system behavior when either of the coefficients change value. The coefficient c is also limited to values between 0 and 5 to ensure that the system is not overly damped.

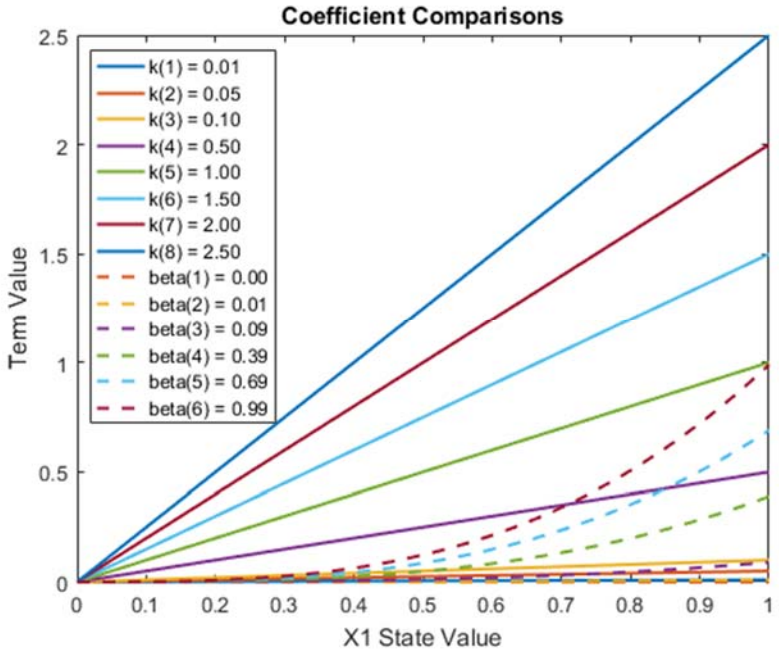


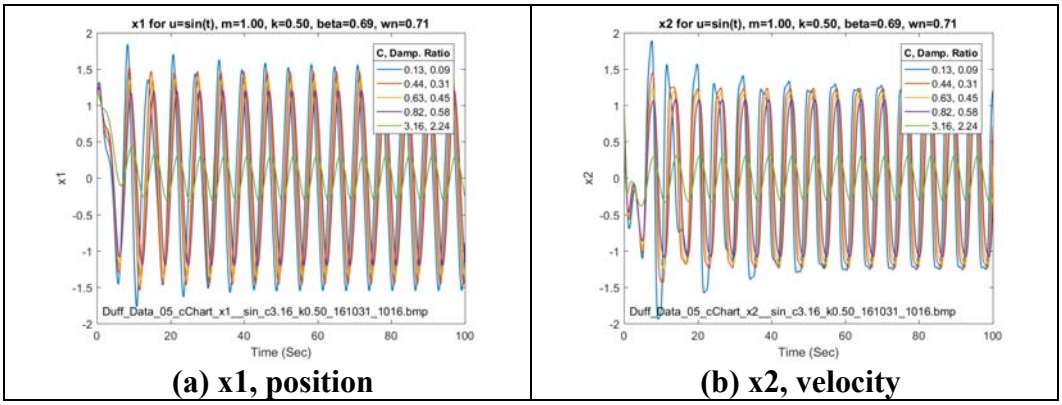
Figure 6-1 Contribution of kx and βx^3 terms

Table 6-1 provides a summary of the values considered for study. Each system was evaluated with β values from the set [0, 0.01, 0.09, 0.39, 0.69, 0.99]. Figure 6-2 is sample plot from the evaluation of changing c with $k =$ 0.5 and

$\beta = 0.69$. Figure 6-3 shows the effect of the range of β values with $c = 0.1265$ and $k = 0.5$.

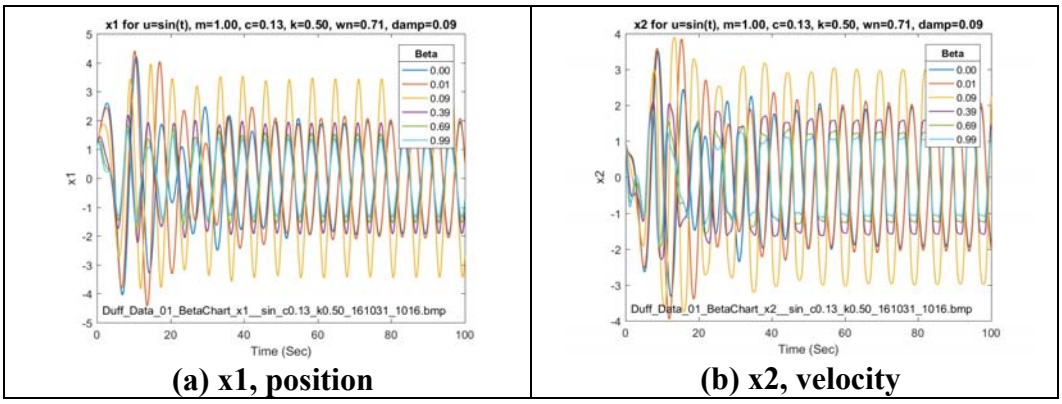
Table 6-1 Coefficient values considered for duffing equation

m	k	c	ω_n	ζ (damping ratio)	Damped Description
1	2.5	0.63246	1.581	0.2	under
1	2.5	2.21359	1.581	0.7	under
1	2.5	3.16228	1.581	1	critically
1	2.5	4.11096	1.581	1.3	over
1	2.5	15.81139	1.581	5	over



(a) x1, position **(b) x2, velocity**

Figure 6-2 Evaluating the effect of changing *c*



(a) x1, position **(b) x2, velocity**

Figure 6-3 Evaluating the effect of changing *c*

6.4.1 Harmonic Excitation

Through the study, the focus was narrowed to the final system with c = 0.1265, $k = 0.5$ and $\beta = 0.69$. Synthetic data generated from the following model.

$$\dot{x} = \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \end{bmatrix} = \begin{bmatrix} x_2 \\ -(c/m)x_2 - (k/m)x_1 - (\beta/m)x_1^3 \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{\gamma}{m} \end{bmatrix} \sin(\omega t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w \quad (6\text{-}11)$$

$$\tilde{y} = x_1 + v \quad (6\text{-}12)$$

where the process noise is of the form $w \sim N(0, r)$ and the measurement noise is of the form $v \sim N(0, q)$. The characteristics of the final system are shown in Table 6-2 and Figure 6-4.

Table 6-2 Characteristics of final duffing oscillator system

Coefficient (Characteristic)	Value
m (mass):	1
c (viscous damping):	0.1265
k (linear stiffness):	0.5
beta (cubic nonlinear stiffness):	0.69
gamma (driving amplitude):	1
omega (driving frequency):	1
q (process noise):	1.0
r (measurement noise):	0.01

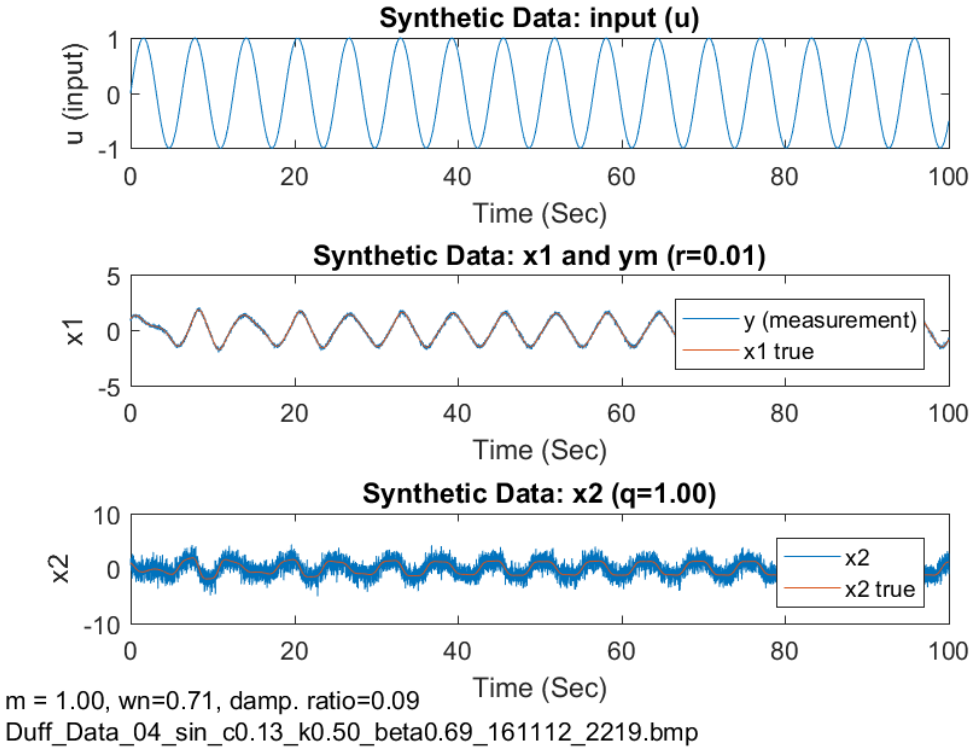


Figure 6-4 Chosen synthetic system

6.4.2 Nonpersistent Excitation

The ideal system for MMAE and other parameter estimation techniques is a system with output measurements and persistent input variation. Since the input is known or measured, there is constant information from the residual calculation from each model. This information is used by the MMAE approach to adjust the parameter estimate weights. This is why an oscillator system is a good system to study. For some systems, the input may become a constant steady state value u_{ss} or

zero. For this case of the nonexistent (zero) input, the system achieves a steady-state value defined by the static gain G_S.

$$G_S\left(u=0\right)=\frac{\gamma}{m} \tag{6-13}$$

For the case of a constant input, u_{ss} the steady state result is simply the static gain multiplied by u_{ss}.

$$G_S\left(u=u_{ss}\right)=\frac{\gamma}{m}\left(u_{ss}\right) \tag{6-14}$$

Both values are not unique concerning an MMAE system with unknown parameters $\theta = [c, k, \beta]$. Each system can achieve this value.

This means that there is no residual information for the MMAE process to adjust the weights and ultimately the parameter estimates. This leads to two observations. First, the system must converge within a time of varying input. Secondly, to verify parameter estimates during a period of unchanging input, some type of synthetic input must be used to generate information to update the MMAE estimates. For the purpose of this research, two signals, the pulse and doublet are evaluated to update the MMAE estimates.

Figure 6-5 provides a comparison of the pulse and doublet with the persistent sinusoidal. Each signal is of the form $\gamma \cdot f\left(\omega,t\right)$ with a constant amplitude γ and a constant frequency ω. Figure 6-6 and Figure 6-7 respectively provide the synthetic simulations of each.

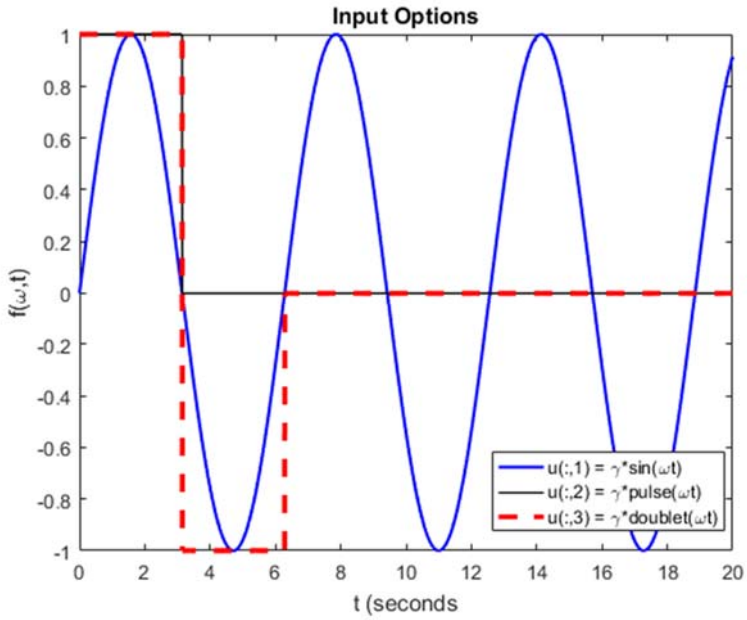


Figure 6-5 Sine, pulse and doublet input with constant frequency(ω) and amplitude (γ)

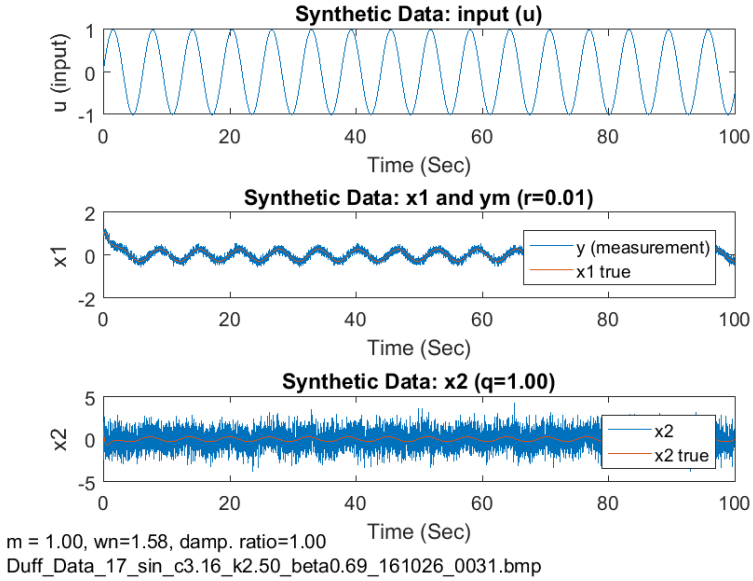


Figure 6-6 Pulse input synthetic data

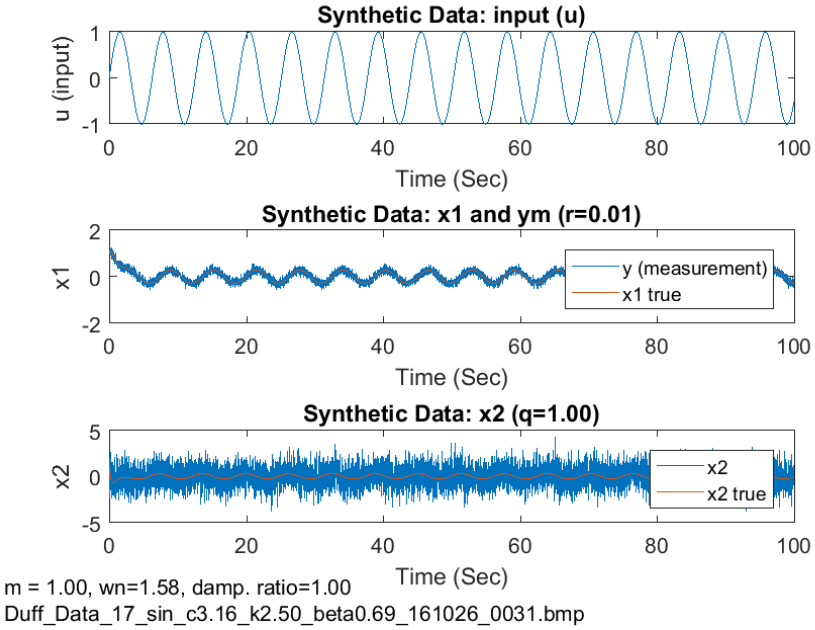


Figure 6-7 Doublet input synthetic data

6.5 Determining a General Configuration for EKF GRAPE

EKF GRAPE simulations were performed on a number of system configurations. Those simulations lead to a generalized approach using tuned values. The values for the general approach of the EKF GRAPE for the Duffing oscillator is recorded in Table 6-3. The following subsections show modifications to the tuned values to show why the values related to the simple moving average window, derivative convergence threshold and resample delay Table 6-3 were chosen. Furthermore, the resample rules are demonstrated. Each example provided in this section was performed on an SGBS simulation.

Table 6-3 General configuration of EKF GRAPE

Model Coefficients	Value/ Limits	Initial Range
m (mass):	1	known value
c (viscous damping):	$\theta^{(1)}$	unknown parameter 1
k (linear stiffness):	$\theta^{(2)}$	unknown parameter 2
β (cubic nonlinear stiffness):	$\theta^{(3)}$	unknown parameter 3
γ (driving amplitude):	1	known value
ω (driving frequency):	1	known value
P_0 (intitial EKF covariance)	$0.1 \cdot I^{2x2}$	from tuning trials
R (process noise)	0.01	from tuning trials
Q (measurement noise)	0.1	from tuning trials
Initial Conditions	**Value**	**Notes**
`range.start_time`	0	initial start time
`range.sample_index`	1	initial index
$\hat{\mathbf{x}}_0$	$[1,1]^T$	
$\mathbf{P}_0$	$P_0 \cdot \mathbf{I}^{2x2}$	$P_{0,SGBS}=1e^{-3}$; $P_{0,LHS}=1e^{-2}$
$\hat{\boldsymbol{\theta}}_0$	$\left[\hat{\theta}_0^{(1)} \quad \hat{\theta}_0^{(2)}\right]^T$	calculated as the midpoint of the initial range
weights (MMAE)	**1/num_trials**	calculated
EKF GRAPE Characteristics	**Value**	**Notes**
`sample_type`	**`'SGBS'`** or **`'LHS'`**	refer to specific simulation for type
`num_trials`	125	5 per parameter for SGBS
	40	for LHS
`limit.lower`	[0.1,0.001, 0]	for $[\theta^{(1)}, \theta^{(2)}, \theta^{(3)}]$ respectively
`limit.upper`	[10,10, 0.99]	
`limit.smallest_percent`	0.1	
`range.min`	[0.5, 0.01, 0]	ideally, most parameter ranges should span actual value (c chosen specifically to demonstrate tracking outside of range)
`range.max`	[5, 2.5, 0.99]	
`resample.reduce`	0.8	
`resample.edge`	0.1	
`resample.enlarge`	1.2	
Convergence Criteria	**Value**	**Notes**
`SMA_num`	**`10`**	
`SMA_flag_limit`	**`15`**	
`derv_limit`	**`0.015`**	generally [0.01, 0.02]; 0.015 or lower for LHS
`derv_limit_redux`	**`0.95`**	
`delay_resample`	**`50`**	typically 200 for LHS
`resample_rule`	**`1`**	

6.5.1 Tuning the EKF Models

There are three parameters used to tune the EKF models, **R** (process noise covariance), **Q** (measurement noise covariance) and $\mathbf{P}_0$ (initial state estimate covariance, typically initialized to a constant, p_0, times the identity matrix as shown in equation (6-15)). In general, these terms are matrices. For the current system of study Q and R are scalar. The values Q and R were determined experimentally by simulating combinations of Q and R to test the performance of the EKF models in the GRAPE algorithm. These simulations covered both the SGBS and LHS sampling approaches. Sample models were run at the values identified in Figure 6-8. The best (Q,R) sets for SGBS and LHS are recorded in Table 6-3.

$$\mathbf{P}_0 = p_0 \cdot I \tag{6-15}$$

Table 6-4 Approach for tuning EKF filters

Component	Approach
R (process noise Covariance)	$Q = E\left\{w\,w^T\right\}$, measure of trust in model; decrease value to increase trust in EKF models
Q (measurement noise covariance)	$R = E\left\{v\,v^T\right\}$, measure of trust in measurements; decrease value to increase trust in measurements
$\mathbf{P}_0$ (initial state estimate covariance)	$\mathbf{P}_0 = p_0 \cdot \mathbf{I}^{2x2}$, measure of covariance of initial value; decrease value until 3σ bounds decrease over simulations

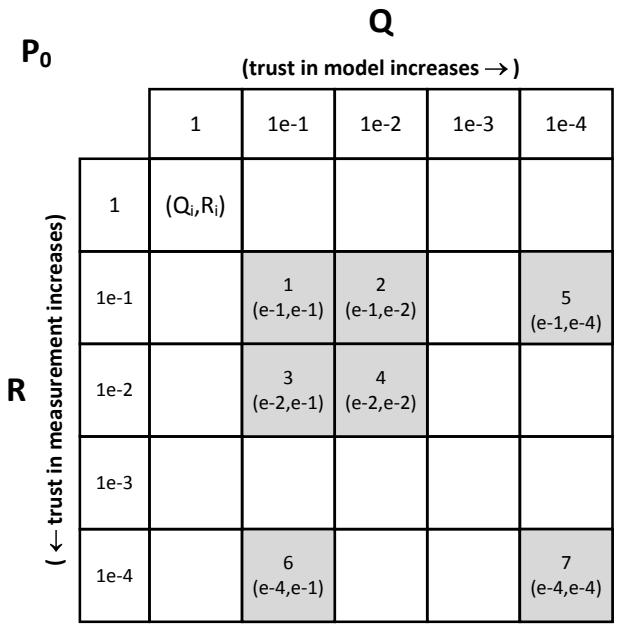
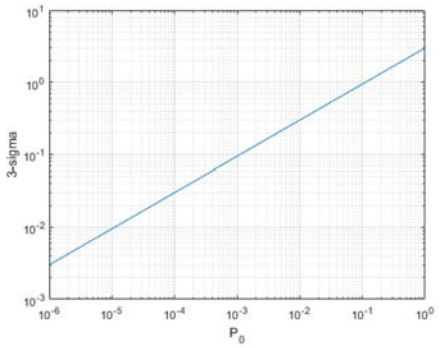


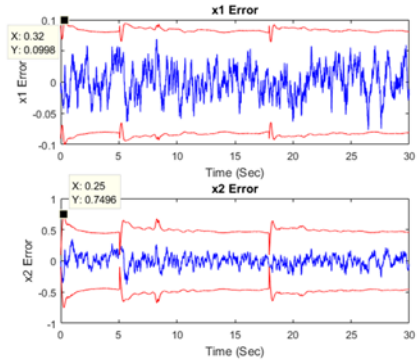
Figure 6-8 EKF tuning process

Positive and negative 3-sigma (3σ) boundaries are calculated by multiplying 3 times the square root of the diagonals of the state estimate covariance, **P**, at each time step. These 3σ values should bound the state estimate errors, also called residuals. The goal is to have an initial covariance such that the 3σ bounds decreases over each sample region. While this approach worked well for LHS, this process deviated slightly for the SGBS. By decreasing P_0 even further, the SGBS approach was found to track nonpersistent signals better. This was not the case for LHS. The models would diverge if P_0 was set too low.



$$+3\sigma_i = +3 \cdot \sqrt{\mathbf{P}_{0,ii}}$$

(a) relationship between 3σ and P_0



$3\sigma_{max} \approx .75$; therefore, $P_0 = (3\sigma_{max})^2$

from $\pm 3\sigma_i = \pm 3 \cdot \sqrt{\mathbf{P}_{0,ii}}$

(b) selecting 3σ from results (SGBS run)

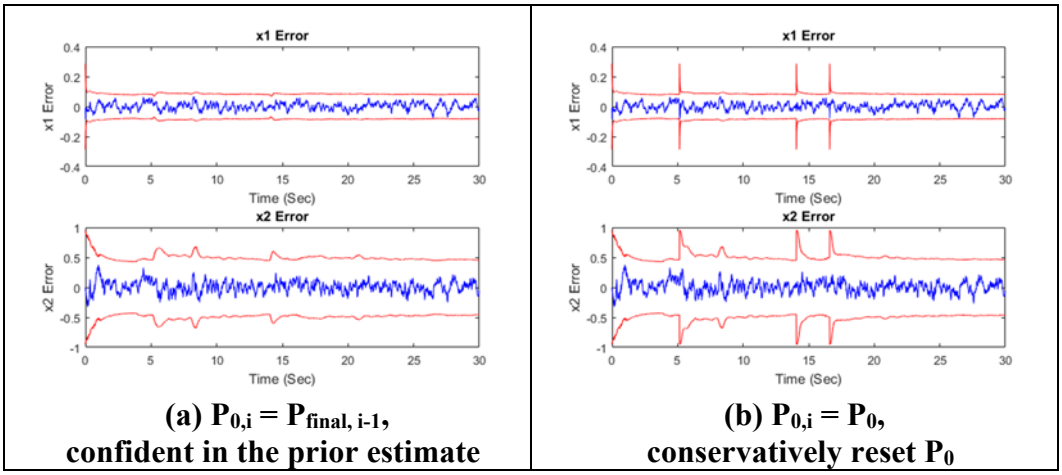
Figure 6-9 Selecting P_0

6.5.2 Option to Use Prior State Estimate Covariance

Another option was explored concerning the initial state estimate covariance, $\mathbf{P}_{0,i}$, for each sample region. The standard approach resets each $\mathbf{P}_{0,i}$ to $\mathbf{P}_0$ during the initialization of a resample. This approach is the conservative option ignored the information available from the prior sample set. It is also the option to use when the desired effect of $\mathbf{P}_0$ is required for the low values of $\mathbf{P}_0$ used to aid the system to rapidly converge for nonpersistent signals for SGBS. The new approach, added late in the research effort, assumes confidence in the prior sample set. This second option uses the prior sample set final covariance, $\mathbf{P}_{i-1,final}$, for the initial value for the covariance, $\mathbf{P}_{0,i}$, for the current resample set.

There are subtle differences in the GRAPE response between the confident and conservative settings for $\mathbf{P}_0$. The most noticeable difference is illustrated in the

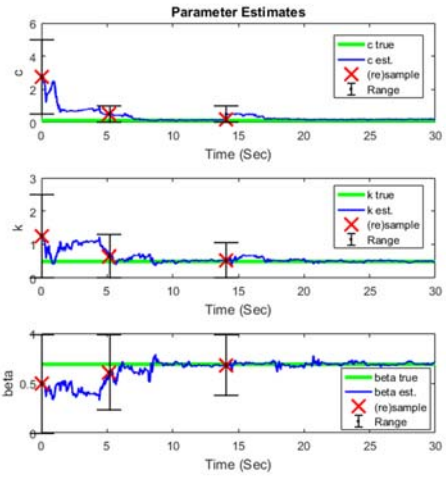
state estimate 3σ bounds. The confident approach provides a smooth almost constant 3σ boundary after the first resample (see Figure 6-10(a)). The conservative response has sharp changes in the 3σ boundary as prior information about the covariance is ignored and reset to the initial value (see Figure 6-10(b)). The effect of this change is subtle is most cases. The confident approach may reduce unnecessary resamples as shown in Figure 6-11. Here the conservative approach, illustrated by Figure 6-11(b), adds an unnecessary resample at approximately 17 seconds when compared to the confident approach of Figure 6-11(a).



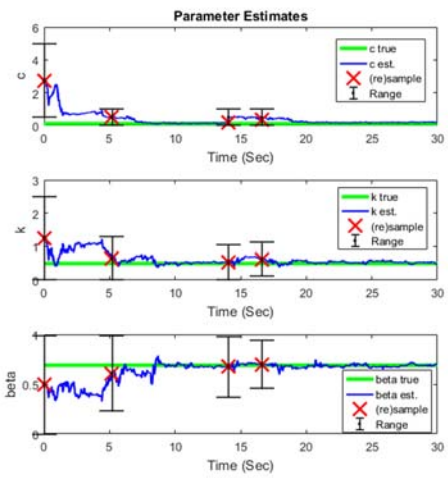
(a) $P_{0,i} = P_{final,\ i-1}$, confident in the prior estimate

(b) $P_{0,i} = P_0$, conservatively reset P_0

Figure 6-10 Effect of P_0 on SGBS state estimate covariances



(a) $P_{0,i} = P_{final, i-1}$, confident in the prior estimate



(b) $P_{0,i} = P_0$, conservatively reset P_0

Figure 6-11 Effect of P_0 on SGBS parameter estimates

6.5.3 Combined Effect of Convergence Threshold and Delay

The SMA derivative estimate is compared to a convergence threshold (derv_limit) for each time step. If the value is met, then a flag (SMA_derv_within_limit) is set for that instance. Once the present number of consecutive flags is reached (controlled by SMA_flag_limit), then parameter resample_flag is set.

It is important for the derivative threshold value to be both attainable and low enough to prevent too frequent resampling. This is illustrated in Figure 6-12. Here the derivative threshold was set to 0.1. This value was too high. The system resampled too often and would not properly converge.

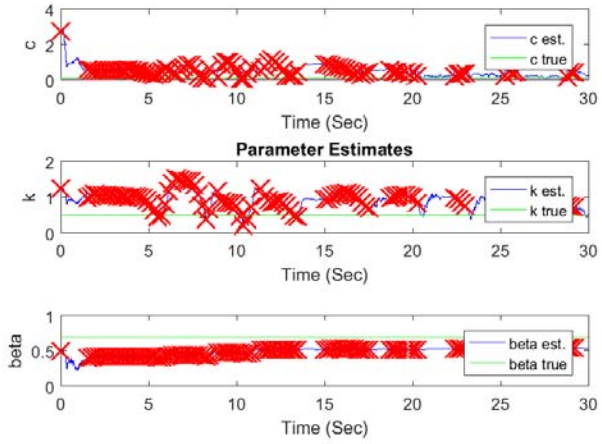


Figure 6-12 System with frequent resampling (threshold to high)

The threshold, derv_limit, may also be set to low and needlessly delay a resample as illustrated in Figure 6-13. Here the estimate for c has clearly converged around 2 seconds; however, the resample is not performed until approximately 14 seconds. In this case, the boundary of the range for *c* is preventing it from approaching the true value. Once a resample is processed, the system converges to the true value of *c*.

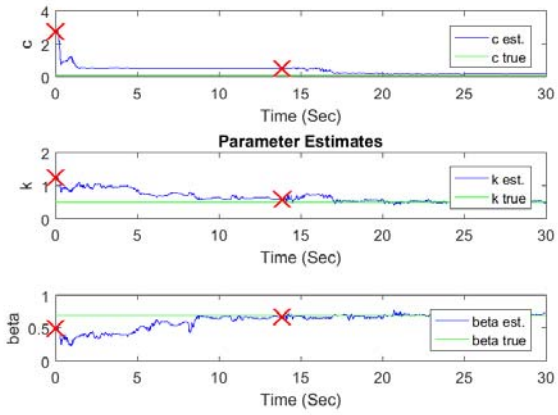


Figure 6-13 System with infrequent resampling (threshold to low)

A delay was implemented to minimize the risk of resampling too often when the threshold derv_limit was set to high. The delay value must be considered along with the desired response of the GRAPE system. A delay too high will needlessly delay resampling as shown in Figure 6-14. In this case, the sampling is primarily driven by the delay. Ideally, the derivative threshold is low enough to properly measure convergence, and the delay acts as a barrier to prevent the next resample from occurring too fast.

Another factor was added to the implementation to make it more difficult to resample by reducing the threshold value after each resample. The derv_limit_redux multiplier is multiplied to the derv_limit for after each resample. Ideally, derv_limit_redux should be near one. For long run with many resamples, the value can reduce to an unattainable value.

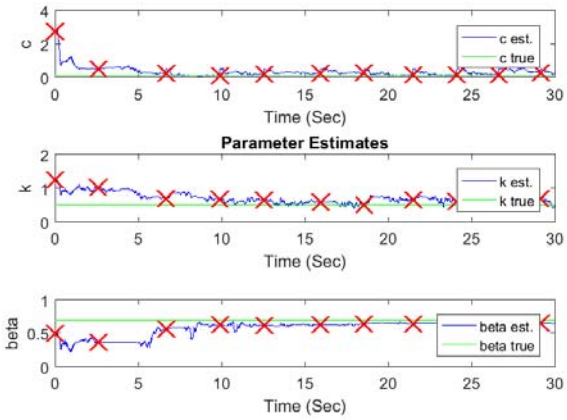


Figure 6-14 System with nearly fixed interval resampling

The ideal configuration is one that resamples when a parameter estimate converges but does not resample again until the system has had time to converge on another estimate. A reasonable system is shown in Figure 6-15. Here, the system resamples upon convergence. The first resample moves the *c* range estimate as the true value of *c* was not within the original range. The second resample is converges close to the true value of *c*. The windows for *k* and β are now moved closer to their true values. Subsequent resampling continue to move the estimates consistently closer to the true values. There is one issue to note for the estimate of *c*. The true value of *c* is near the lower limit for that parameter dimension. The window size has also reached is lower width parameter. The true value of *c* is at the lower end of that window. When a resample occurs, error is initially introduced as the weights of the MMAE process are all initialized to 1/(number of models). This causes the initial estimate to be in the center of the range. The MMAE process will rapidly overcome this error.

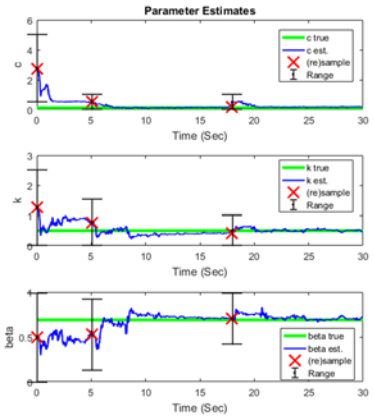
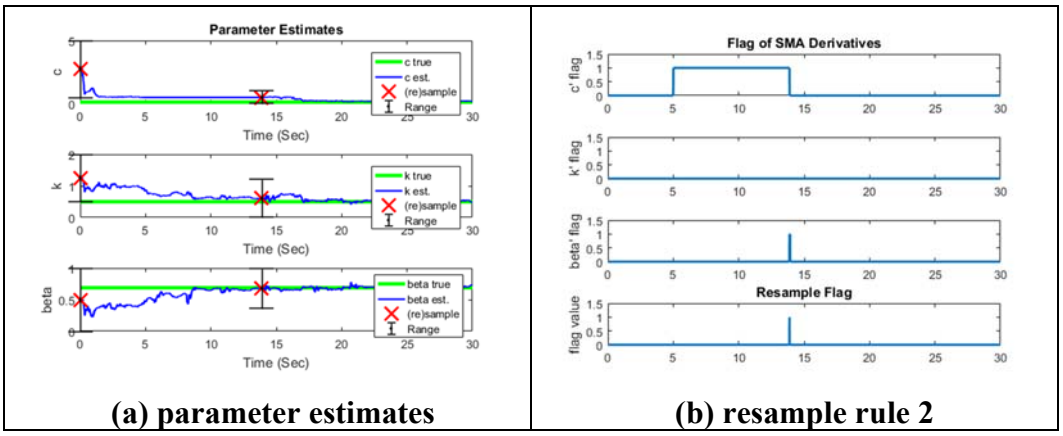


Figure 6-15 System with reasonable interval resampling

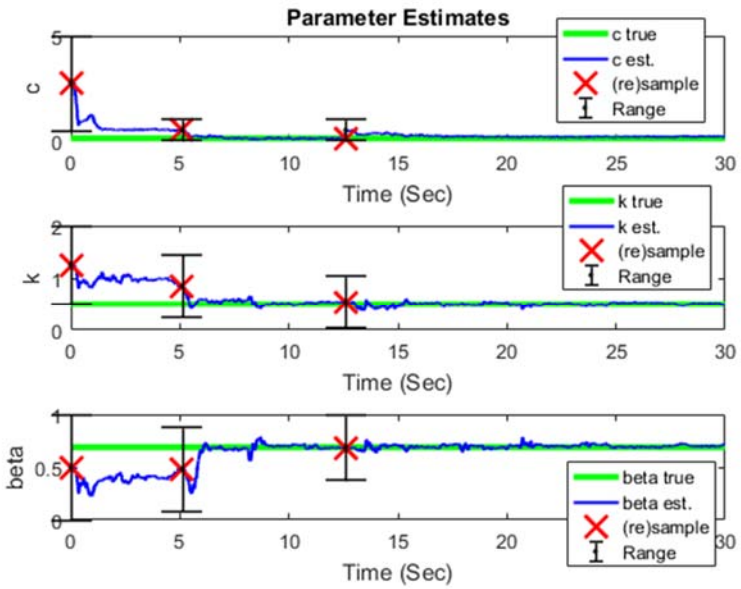
6.5.4 Resampling Rules

Rule-based resampling determines the final decision to resample or not. At the present time, there are three rules. The rule number represents the required number of dimension resample flags to be set to trigger a system resample. For the three-dimensional system implemented for the Duffing oscillator, the available rules are 1, 2 and 3. Testing has shown that the starting point should be rule 1 requiring just one dimension to converge. Otherwise, the resample will likely be excessively delayed as shown in Figure 6-16. For this case, both rule 2 and a delay of 5 seconds is used. Resampling was excessively delayed until a second convergence flag was reached at approximately 14 seconds. This prevented the *c* parameter from resampling and moving closer to its true value. (Recall, the initial range of the parameter estimate for *c* did not cover its true value, see Table 6-3.) A better result is obtained from Rule 1 with a 5-second delay as displayed in Figure 6-17.

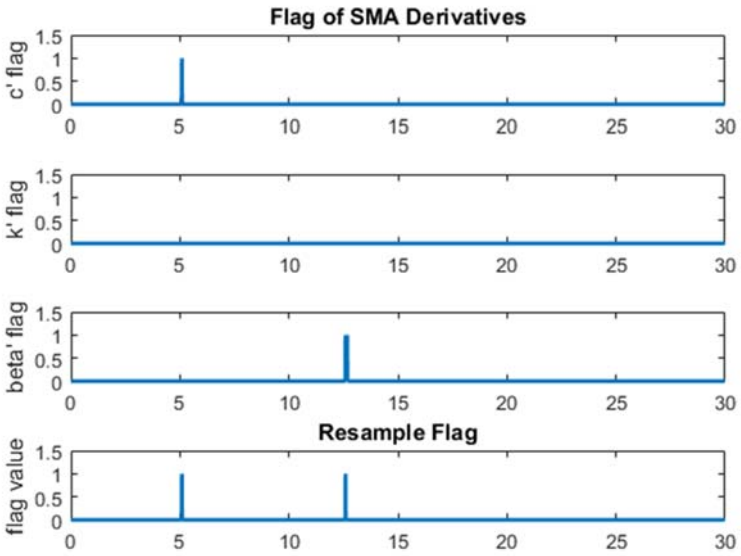


(a) parameter estimates **(b) resample rule 2**

Figure 6-16 Rule 2 resampling



(a) Parameter Estimates



(b) Derivative Flags and Resample Rules

Figure 6-17 Resampling with rule 1

6.6 Simulations

Section 6.5 provides the discussion of an adequate set configuration options, summarized in Table 6-3, for executing EKF GRAPE on the Duffing oscillator system of Table 6-2. The values of Table 6-3 were determined from qualitatively evaluating hundreds of simulations. With the synthetic data that already includes a predefined set of noise, the results of specific configuration SGBS are deterministic. In other words, a particular simulation will result in the same results with the same settings. For LHS the sampling is random. Therefore, the simulations will not perform the same.

The configuration of Table 6-3 is now demonstrated with results for both the SGBS and LHS sampling approaches for both persistent sinusoidal input along with the pulse and doublet inputs. Recall, the standard input model for the nonlinear test was the Duffing oscillator system of equations (6-11) and (6-12) with the configuration of Table 6-2. This system was demonstrated in Figure 6-4 for sinusoidal input, Figure 6-6 for the pulse input and Figure 6-7 for the doublet input.

The results are presented in the following manner. First, the results of persistent excitation are presented for both the SGBS and LHS sampling methods using a sinusoidal input signal. Second, a pulse with the same frequency is discussed. This is followed by the doublet input signal. A summary of the referenced simulation figures is provided in Table 6-5. Each representative simulation is present in three figures. The first figure provides the state estimates

$\hat{\mathbf{x}}$ and the residuals with the 3σ bounds. The second figure provides the parameter estimates $\hat{\boldsymbol{\theta}}$. The parameter range boundaries are represented by the interval of the error bars at the initial sample location at t = 0 and each subsequent resample. All points of sampling are marked with a red x. The third figure illustrates the parameter derivatives and the associated dimension resample flag based on the threshold derv_limit. The third figure also shows the overall GRAPE resample flag per rule 1. These results are presented with the conservative option of resetting $\mathbf{P}_{i,0}$ to $\mathbf{P}_0$ for each resample.

Table 6-5 Simulation summary

Input Signal	Sampling Method	Representative Simulation Figure
sin(t)	SGBS	Figure 6-18 SGBS state estimates and residuals example
sin(t)	SGBS	Figure 6-19 SGBS parameter estimates example
sin(t)	SGBS	Figure 6-20 SGBS parameter derivatives and convergence flags
sin(t)	LHS	Figure 6-21 LHS state estimates and residuals
sin(t)	LHS	Figure 6-22 LHS parameter estimates
sin(t)	LHS	Figure 6-23 LHS parameter derivatives and convergence flags
pulse(t)	SGBS	Figure 6-24 SGBS state estimates and residuals
pulse(t)	SGBS	Figure 6-25 SGBS parameter estimates
pulse(t)	SGBS	Figure 6-26 SGBS parameter derivatives and convergence flags
pulse(t)	LHS	Figure 6-27 LHS state estimates and residuals
pulse(t)	LHS	Figure 6-28 LHS parameter estimates
pulse(t)	LHS	Figure 6-29 LHS parameter derivatives and convergence flags
doublet(t)	SGBS	Figure 6-30 SGBS state estimates and residuals
doublet(t)	SGBS	Figure 6-31 SGBS parameter estimates
doublet(t)	SGBS	Figure 6-32 SGBS parameter derivatives and convergence flags
doublet(t)	LHS	Figure 6-33 LHS state estimates and residuals
doublet(t)	LHS	Figure 6-34 LHS parameter estimates
doublet(t)	LHS	Figure 6-35 Parameter derivatives and convergence flags

6.6.1 Persistent Sinusoidal Input Simulations

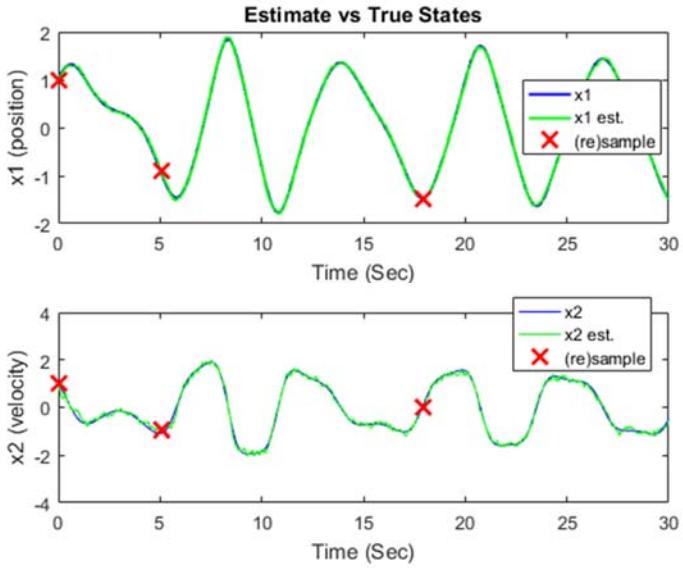
EKF GRAPE performed well for the persistent sinusoidal input and the configuration of Table 6-3. With 125 models, 5 values for each parameter

dimension, SGBS tracks the true states very well. SGBS takes longer to simulate but will converge close to the true value if enough models are used, and the EFK models are appropriately tuned. The LHS will execute much faster directly proportional to the lower model count. The LHS simulations used 40 total models rather than the 125 of SGBS. LHS derivatives will converge much faster in the simulation due to the lower number of models. Therefore, a lower threshold is used. Additionally, a larger delay is set to prevent too frequent resampling from interfering with the convergence to the true values.

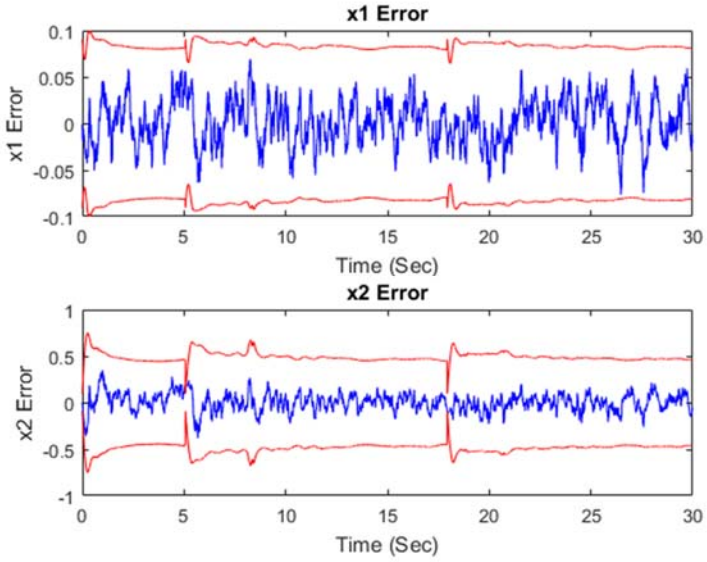
For some rare LHS sample sets, the LHS approach will settle on a system that produces results with the parameters estimates significantly different from the true values (true values of parameters are outside the boundary of the range window for the parameter estimate). In this case, the c estimate converges near the true value, and the k estimate is always high with the β estimate always low. Initial evaluations of these differences point to the system similarity.

The parameter space likely has systems that produce similar results. The MMAE process itself uses the information of the residuals to estimate the states and parameter estimates. There are far fewer samples in the LHS simulations. Therefore, the residuals are typically higher because the simulated models are potentially separated further from the true model. Therefore, the LHS is more likely to converge on an alternate model that produces similar output for the same input than the much higher model count of the SGBS approach. In the case of a

similar model, the state estimates will be accurate but the parameter estimates will not be correct.



(a) state estimates



(b) residuals and 3σ bounds

Figure 6-18 SGBS state estimates and residuals example

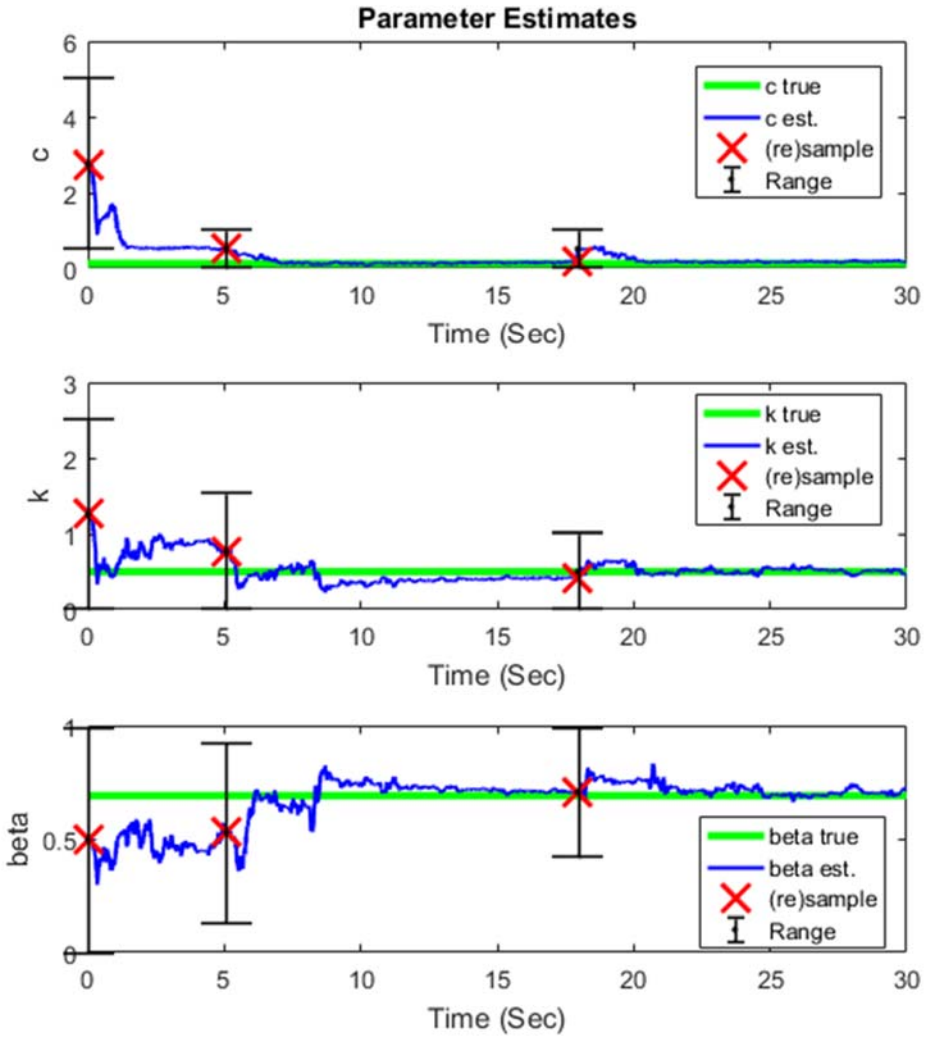
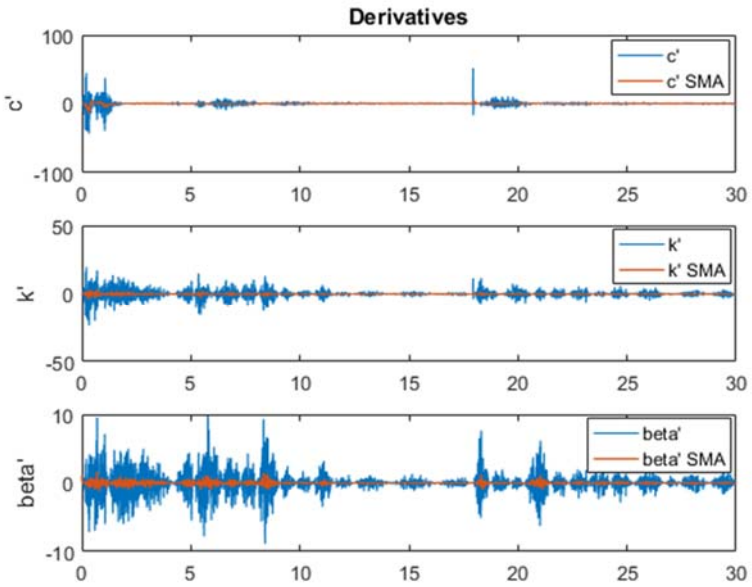
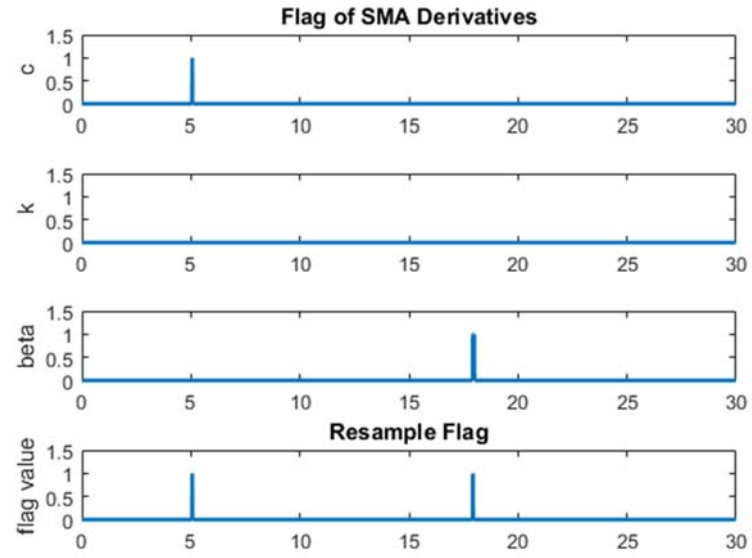


Figure 6-19 SGBS parameter estimates example

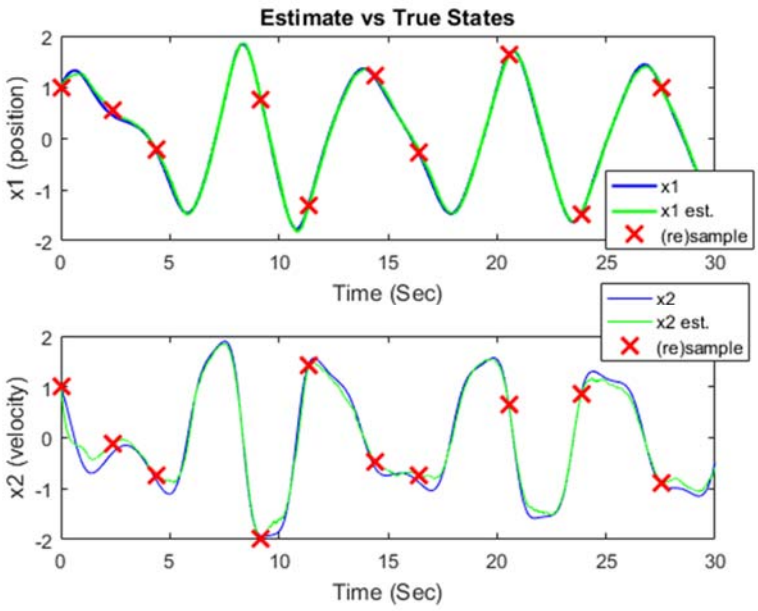


(a) parameter derivatives

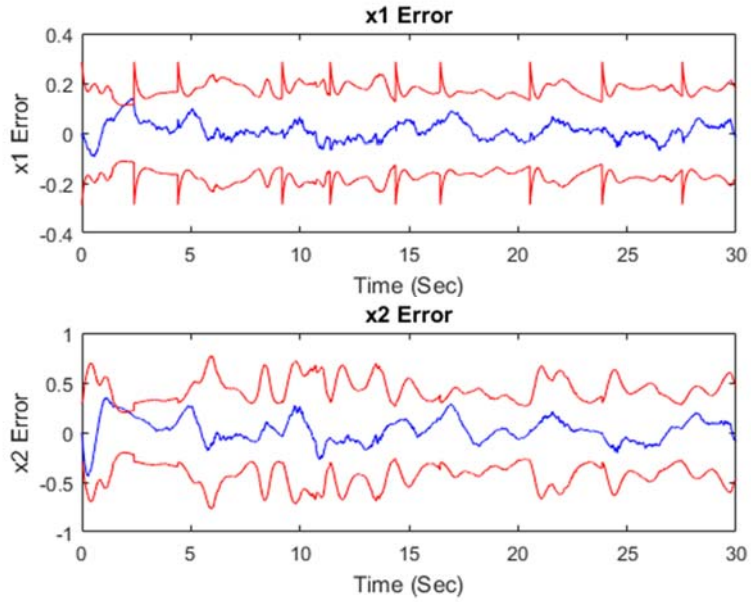


(b) sma derivative and resample flags

Figure 6-20 SGBS parameter derivatives and convergence flags



(a) state estimates



(b) residuals and 3σ bounds

Figure 6-21 LHS state estimates and residuals

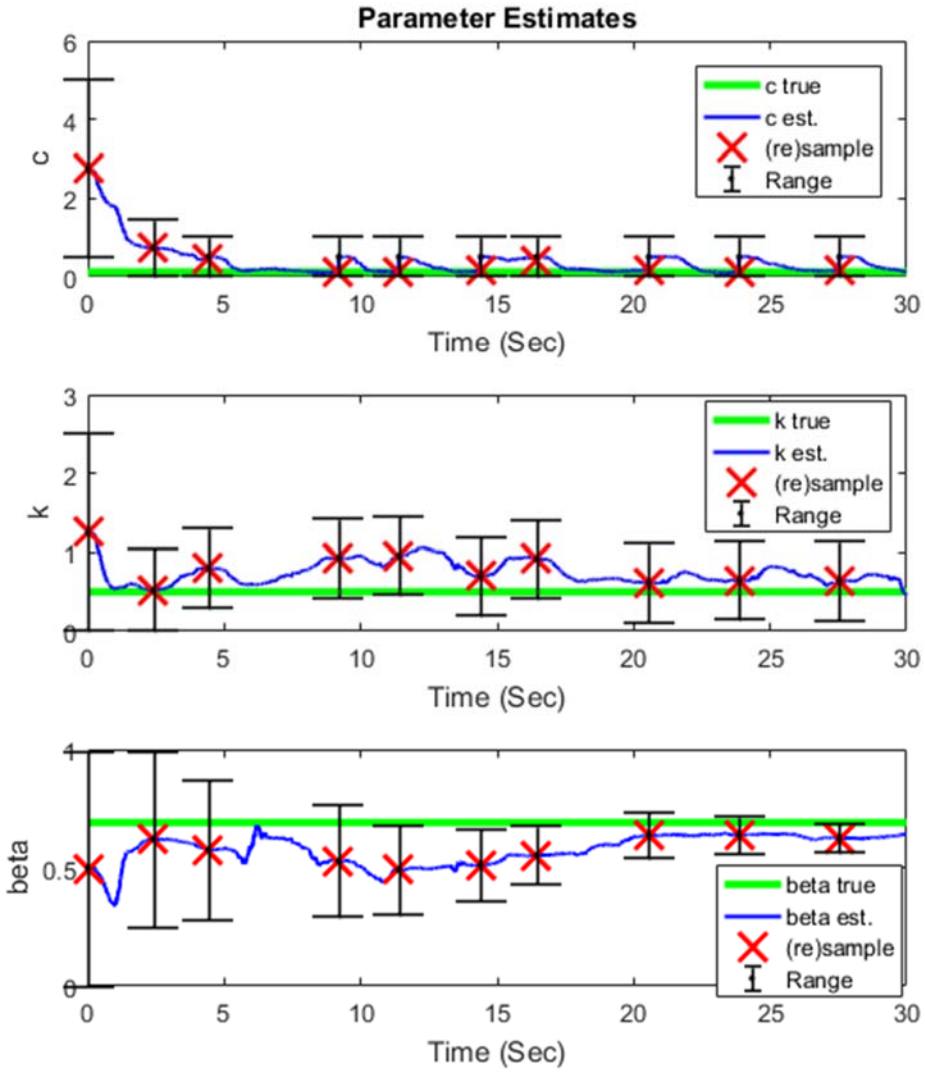
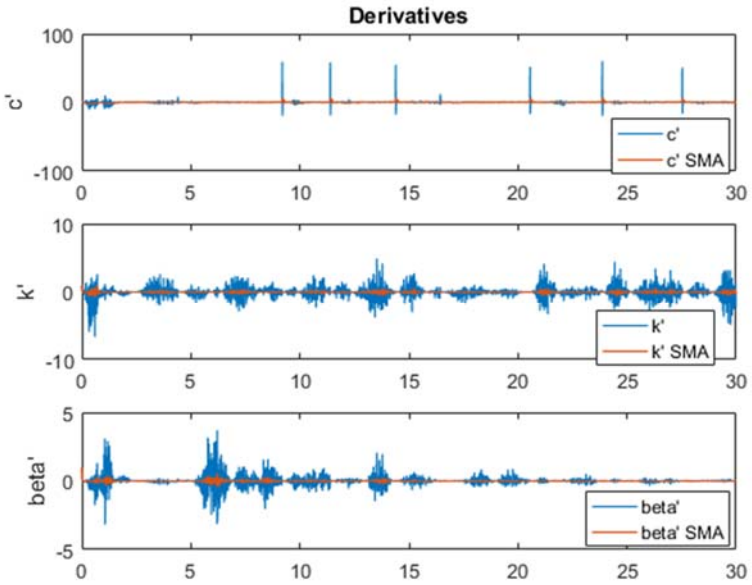
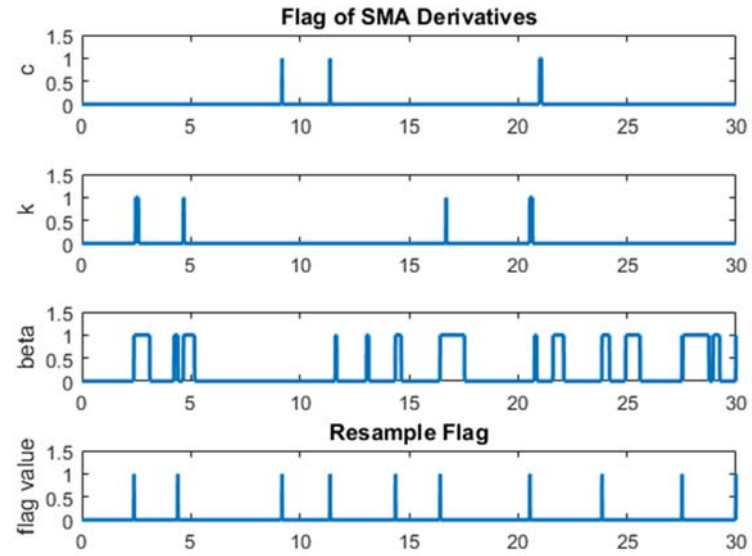


Figure 6-22 LHS parameter estimates



(a) parameter derivatives

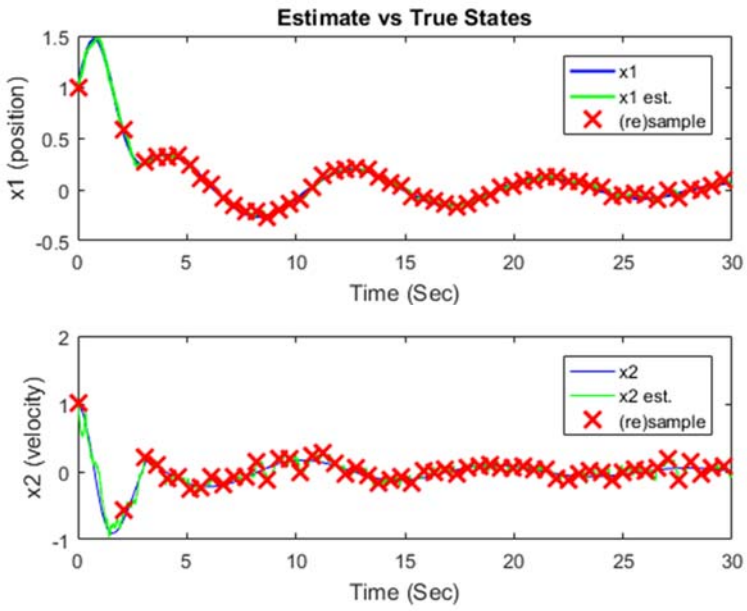


(b) sma derivative and resample flags

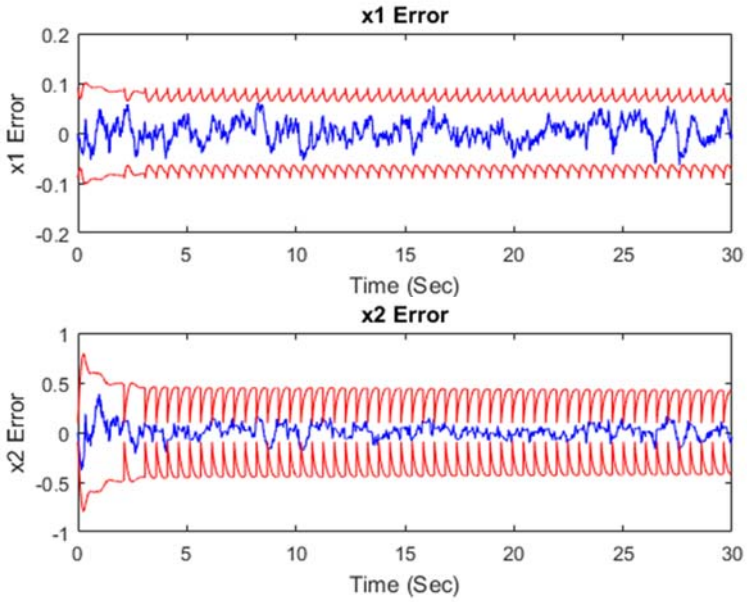
Figure 6-23 LHS parameter derivatives and convergence flags

6.6.2 Pulse Input Simulations

The pulse signal was evaluated to see if it could be introduced when the input and output of the system are constant. In this case, there is no information to update the models (see discussion of section 6.4.2 for further details). A single positive pulse of the same frequency as the sinusoidal signal provides a very short window of residual information for the GRAPE algorithm to update the models. For the SGBS, the models can be tuned to rapidly converge to the true estimates with an artificially very low $\mathbf{P}_0$. However, the information window is not long enough to converge to the true values. Furthermore, there is no additional information from the residuals after the effect of the pulse is over and the estimate will drift. LHS produces similar results. When the pulse signal is present, the GRAPE algorithm approaches an estimate towards the true system value. However, the information is not present long enough for the GRAPE method to converge on the true states.



(a) state estimates



(b) residuals and 3σ bounds

Figure 6-24 SGBS state estimates and residuals

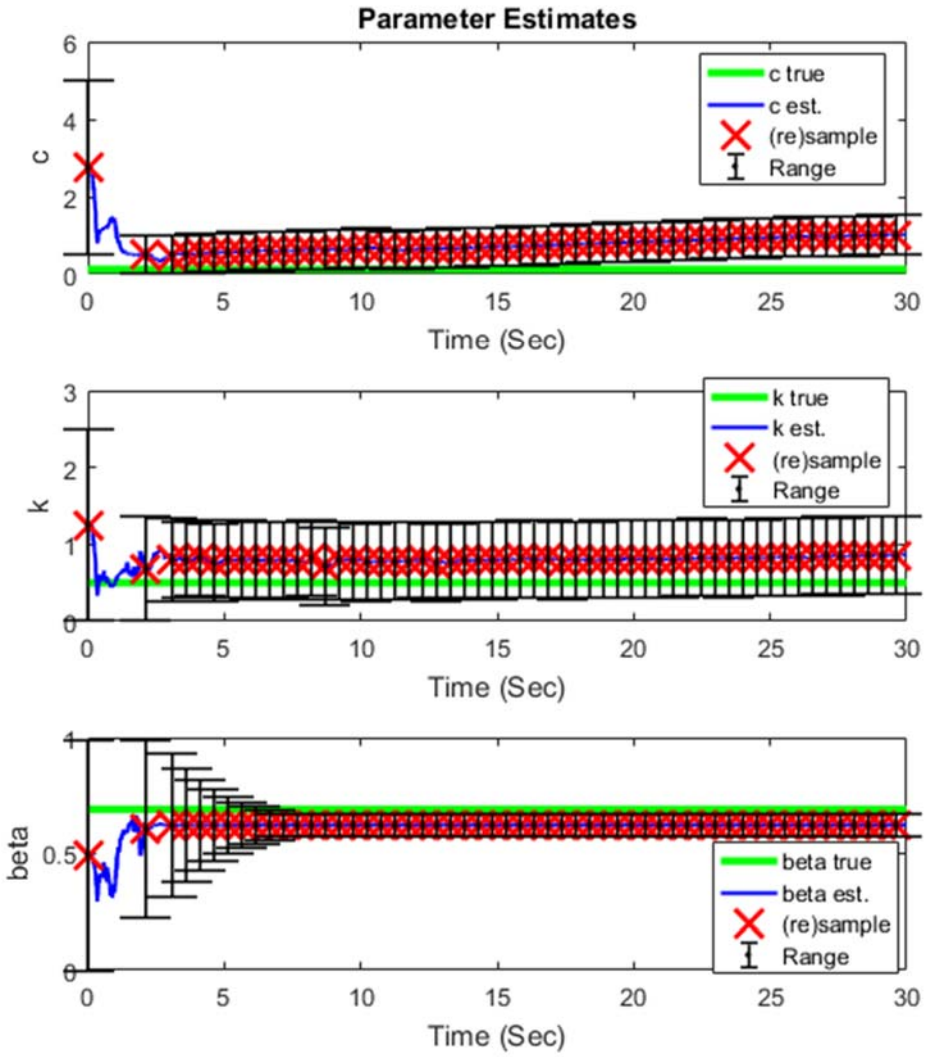
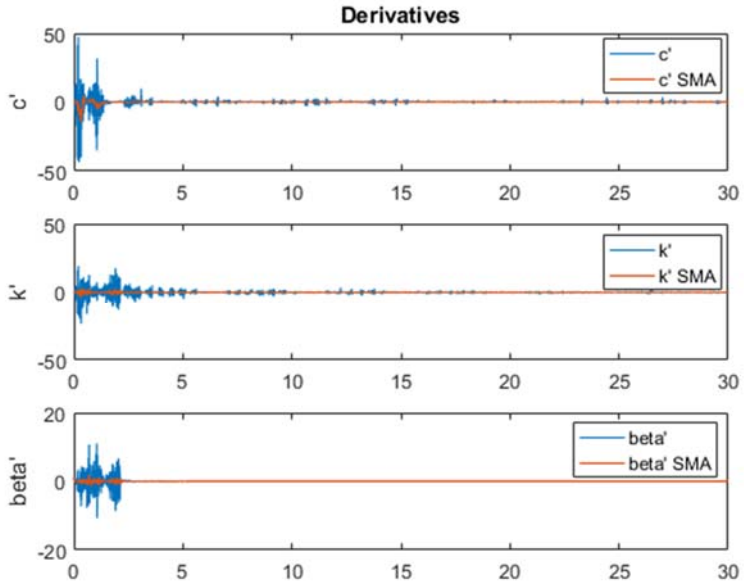
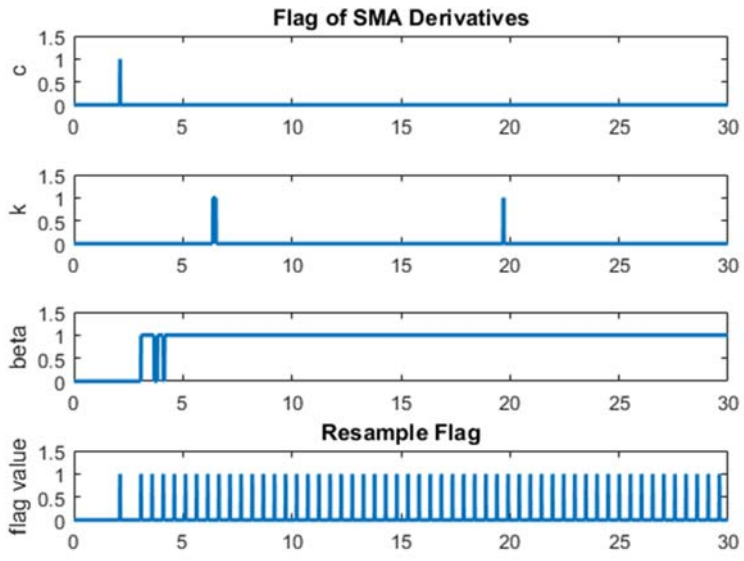


Figure 6-25 SGBS parameter estimates

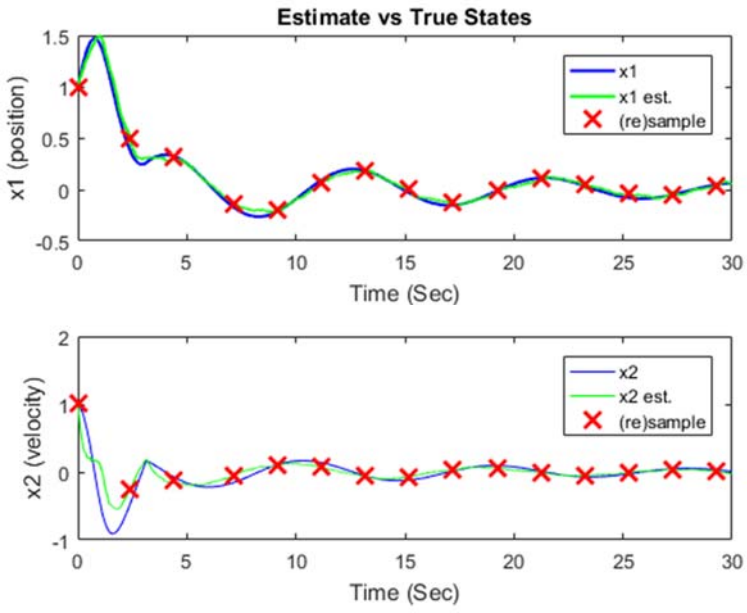


(a) parameter derivatives

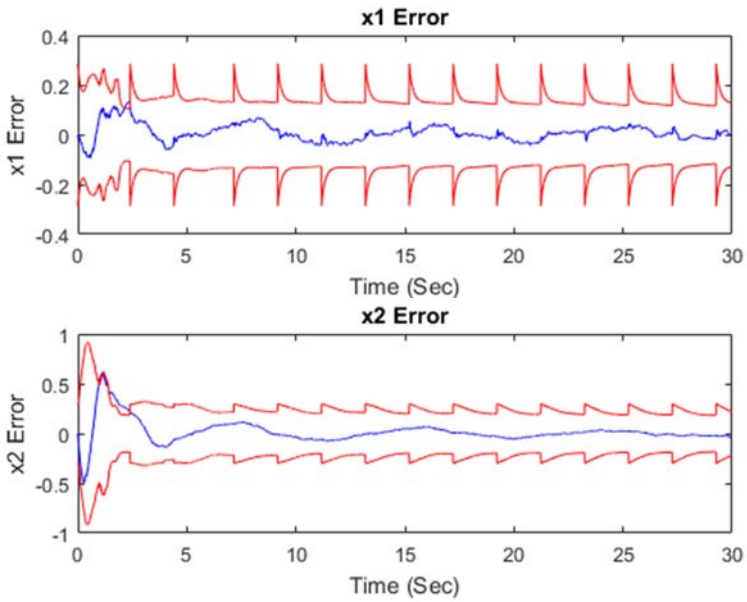


(b) sma derivative and resample flags

Figure 6-26 SGBS parameter derivatives and convergence flags



(a) state estimates



(b) residuals and 3σ bounds

Figure 6-27 LHS state estimates and residuals

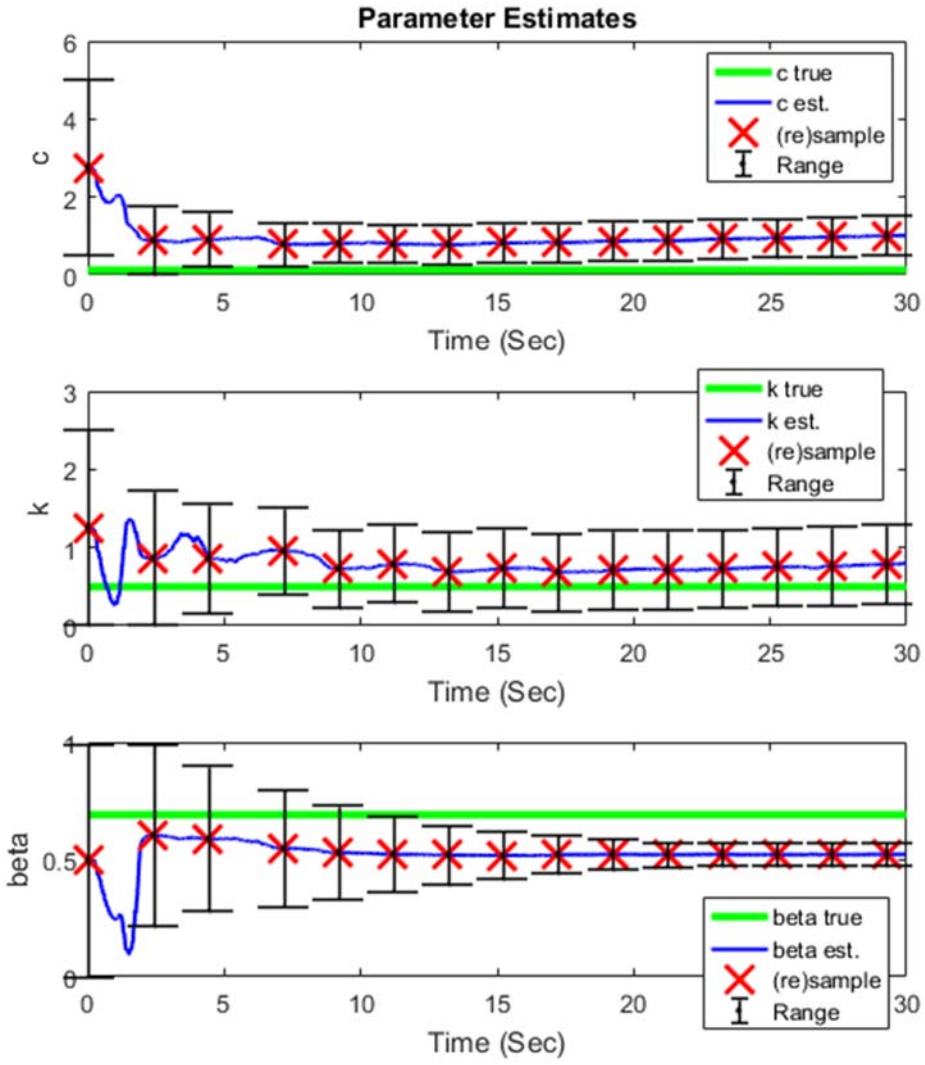
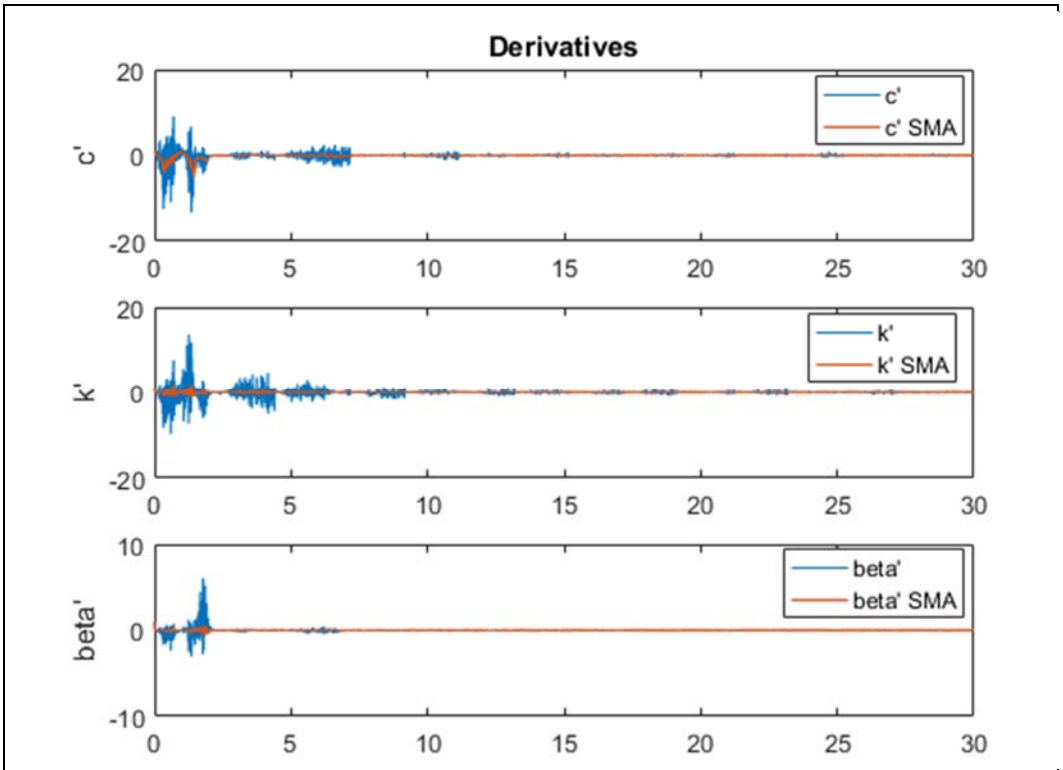
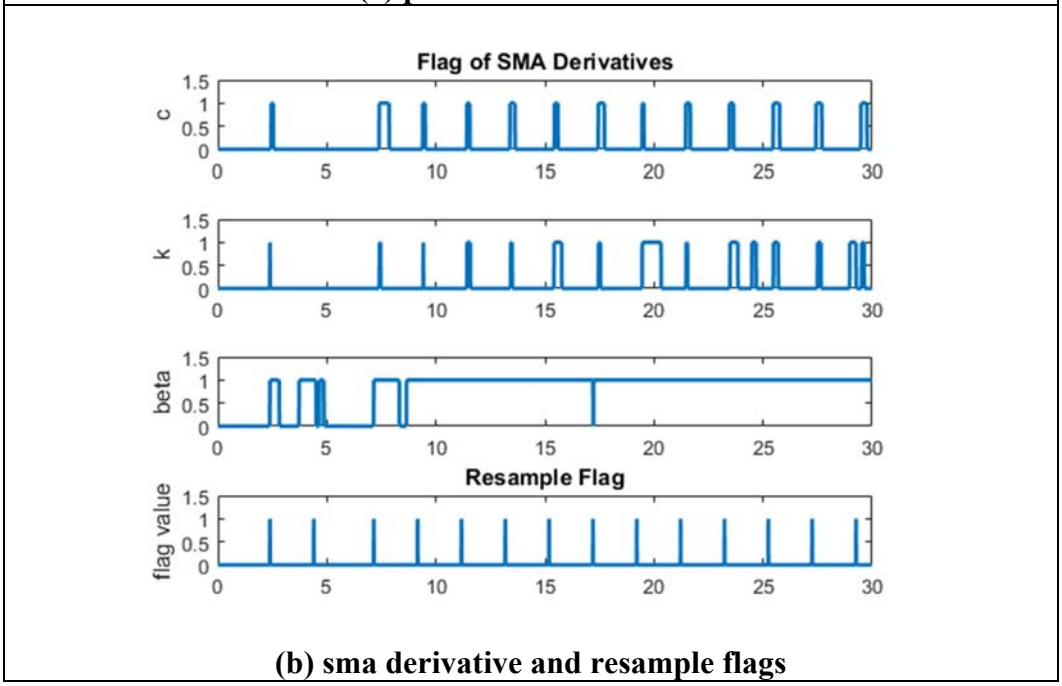


Figure 6-28 LHS parameter estimates



(a) parameter derivatives

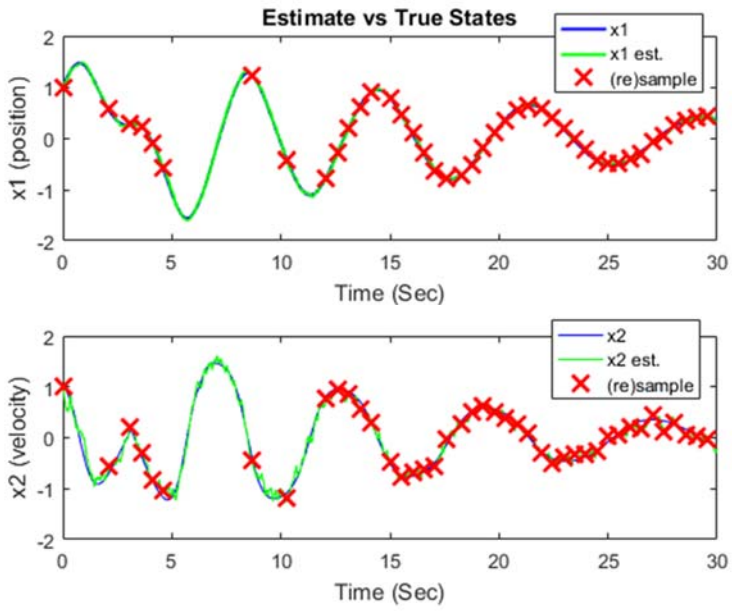


(b) sma derivative and resample flags

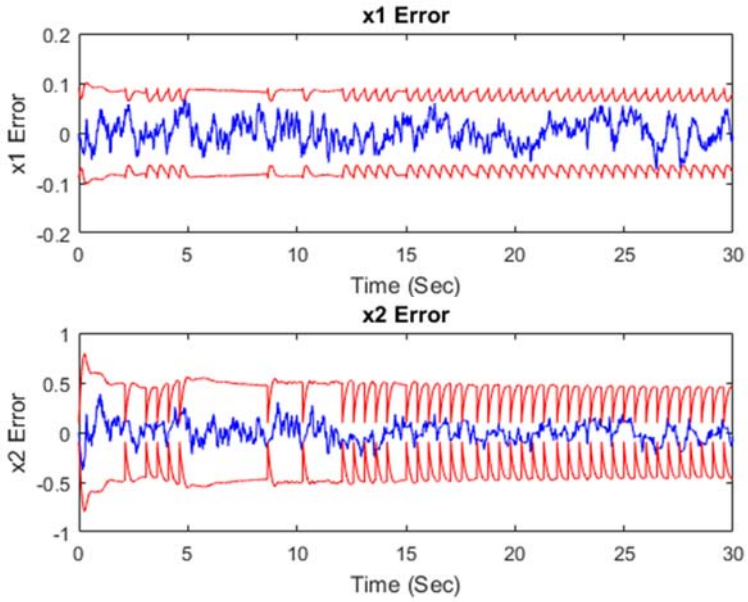
Figure 6-29 LHS parameter derivatives and convergence flags

6.6.3 Doublet Input Simulations

The double provides an information window twice as long as the pulse. Therefore, both the SGBS and the LHS sampling methods do a better job of converging to the true system parameters. When the double signal is not present, there is no further residual information available for the GRAPE algorithm to use. Ultimately, inserting an artificial signal when a system has a steady state constant input and corresponding output will allow the GRAPE algorithm to properly update the parameter estimates as long as the information window is long enough for the estimates to converge. The duration of the required artificial input is dependent on the system parameters that were set to define the desired performance. For the configuration of Table 6-3, a two cycle doublet should suffice for a parameter estimate update. However, the system can drift after the artificial signal subsides. Alternatively, the GRAPE algorithm could be modified to recognize a steady-state input/output situation and halt all processing until more information is available. This potential modification is left for future effort.



(a) state estimates



(b) residuals and 3σ bounds

Figure 6-30 SGBS state estimates and residuals

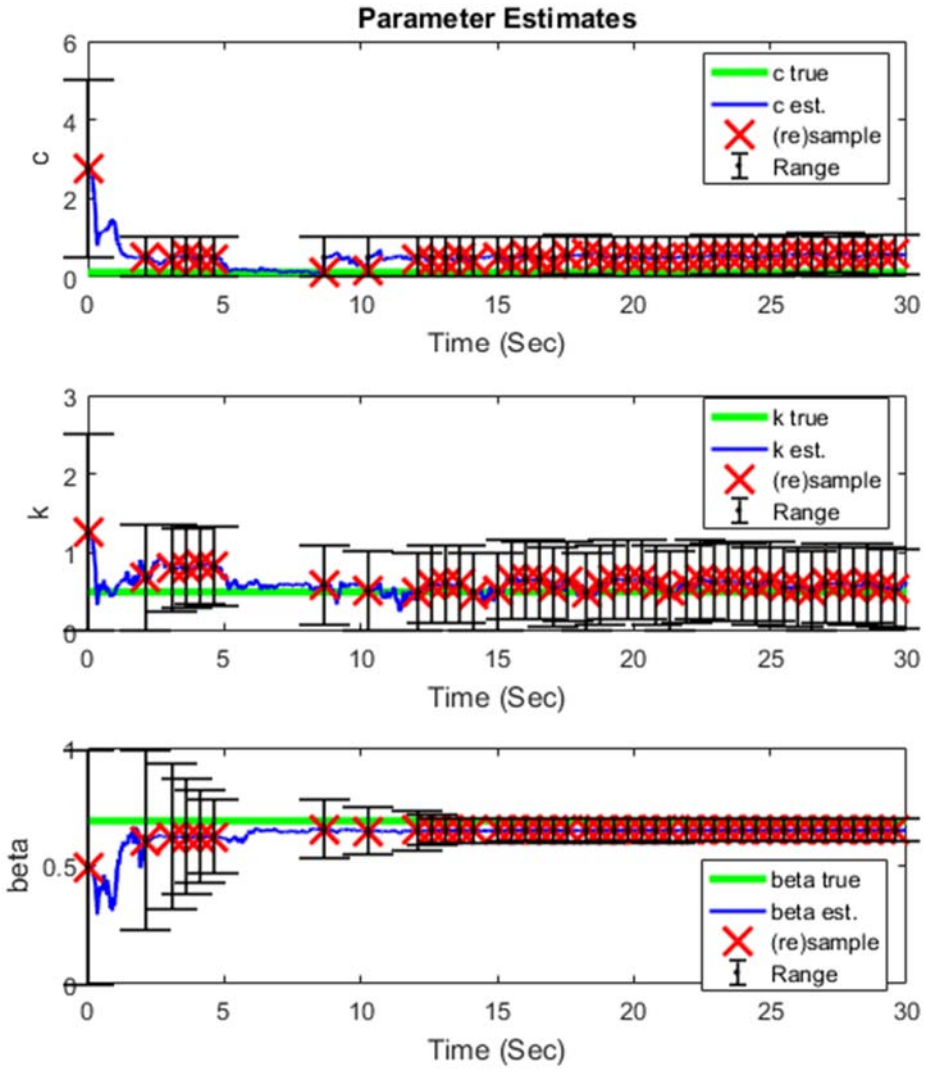
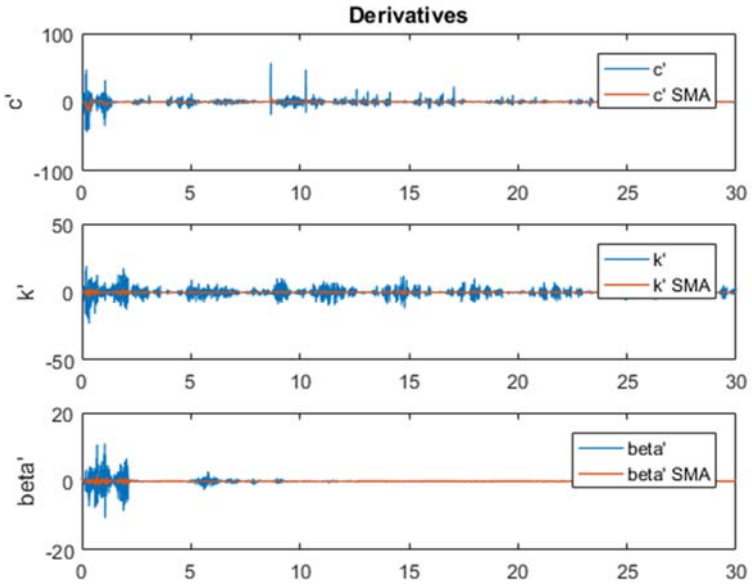
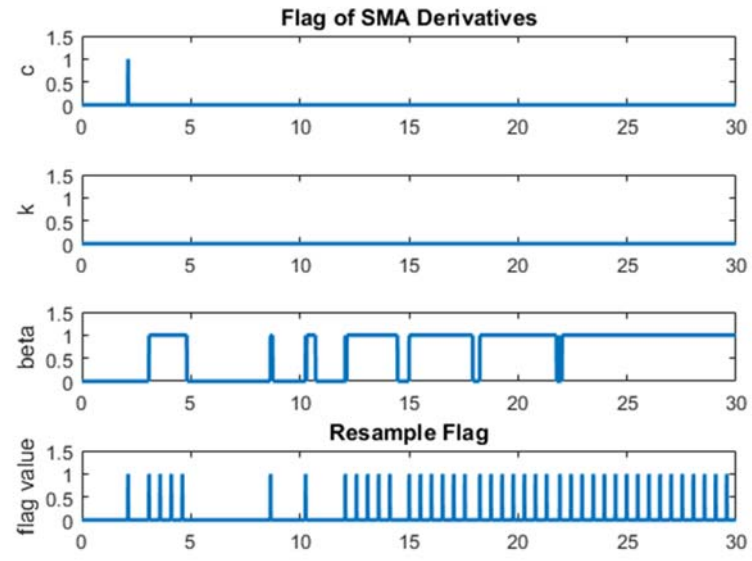


Figure 6-31 SGBS parameter estimates

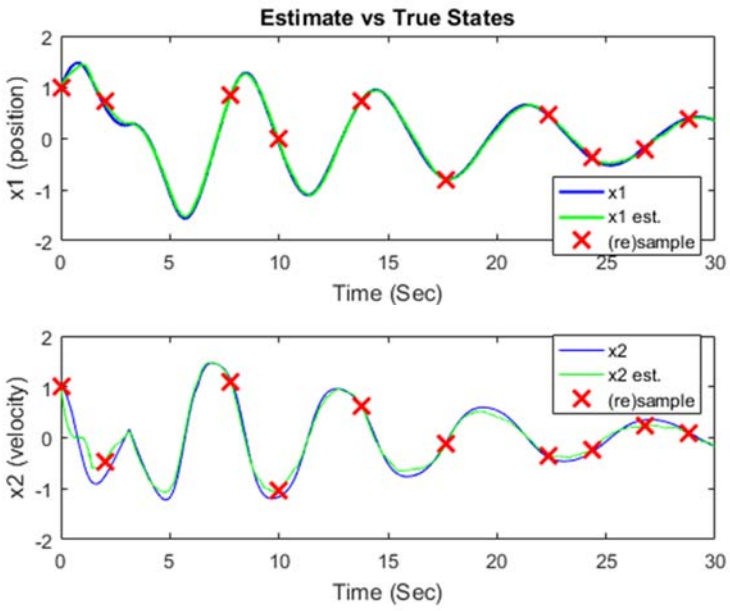


(a) parameter derivatives

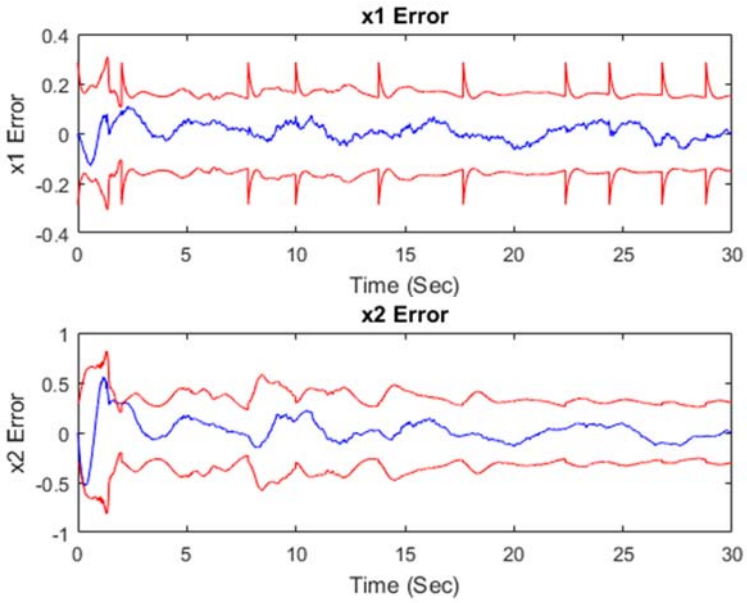


(b) sma derivative and resample flags

Figure 6-32 SGBS parameter derivatives and convergence flags



(a) state estimates



(b) residuals and 3σ bounds

Figure 6-33 LHS state estimates and residuals

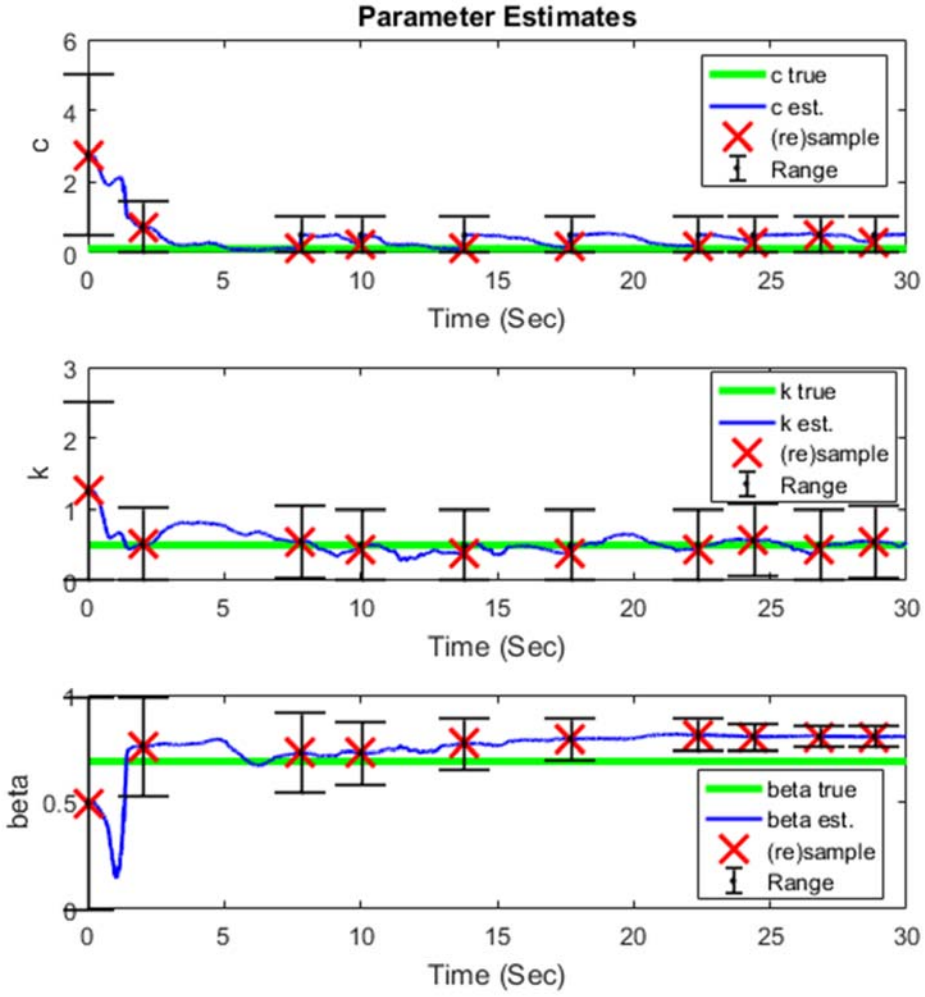
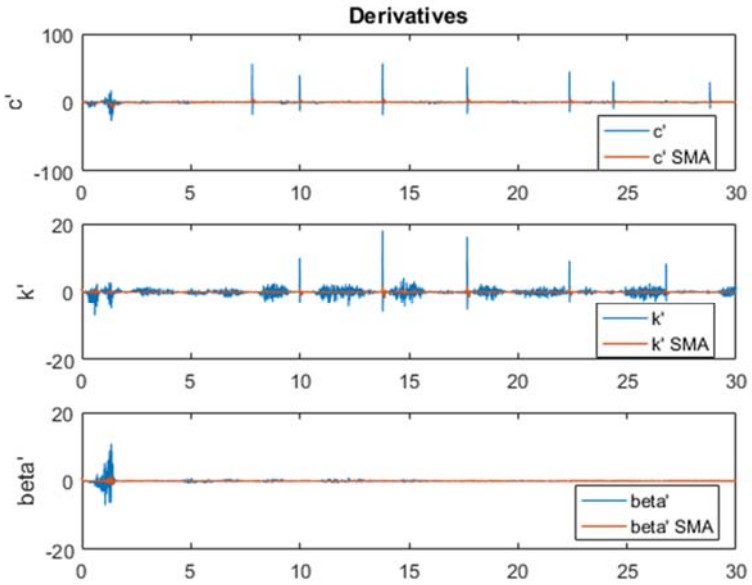
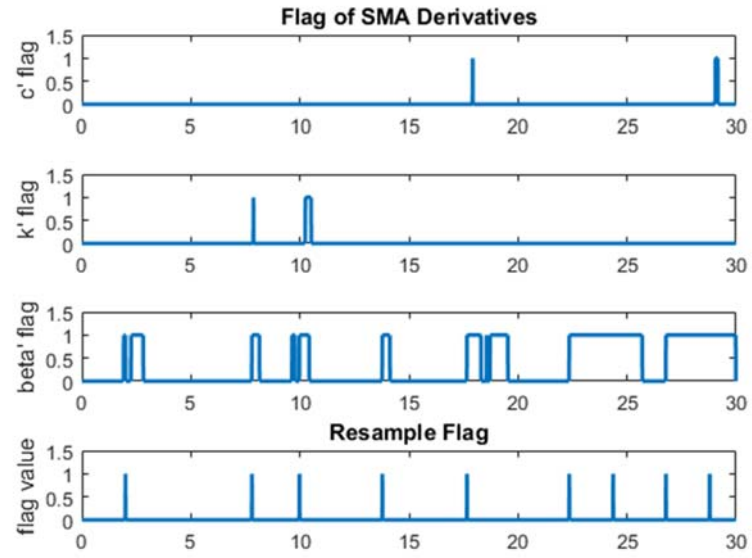


Figure 6-34 LHS parameter estimates



(a) parameter derivatives

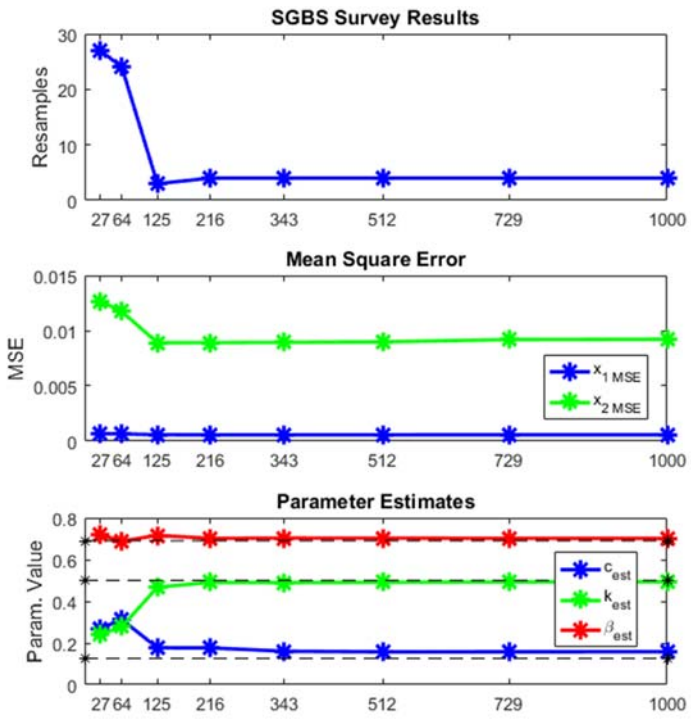


(b) sma derivative and resample flags

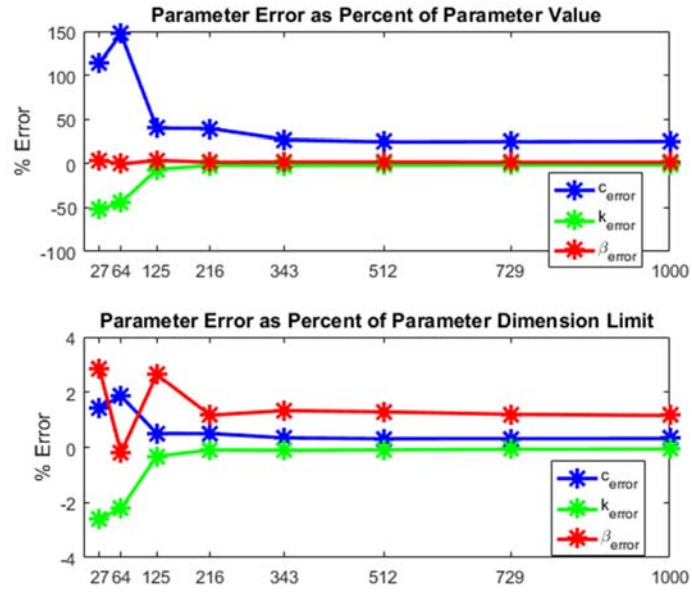
Figure 6-35 Parameter derivatives and convergence flags

6.6.4 Simulation Summary

The SGBS approach is considered the baseline. SGBS guarantees accurate results as long as there are enough stratum and the EKF filters have been properly tuned. In general, the SGBS will provide the best mean squared error when comparison to the true values store when the simulated input/output data was generated. SGBS provides the best final parameter estimate and error. The accuracy comes at the cost of computation time. To close out the SGBS evaluation, a study was performed using the hypothesis model size of [3^3, 4^3, 5^3, 6^3, 7^3, 8^3, 9^3, 10^3] for the 3 parameter space of the Duffing oscillator under consideration. There were only minor improvements after 5^3 (125) models. The diminishing returns do not justify a higher model count for the Duffing oscillator system considered. This conclusion is only valid for the of Duffing oscillator with characteristics of Table 6-2, and the parameter space and the ranges of Table 6-3. Figure 6-36 provides a summary of the effect of increasing the number of models. The number of resamples stabilizes at 125 samples. The mean squared error (comparing to the stored true values) levels out at 125 samples with only minor improvement for higher sample counts. The parameter estimates also see little improvement when the model count is increased as shown in the parameter values and the relative error. The best measure of relative errors is error as a percent of the parameter dimension limit. This compares the error to the total size of the parameter space considered.



(a) SGBS survey results



(b) relative error

Figure 6-36 Parameter derivatives and convergence flags

6.7 Summary of GRAPE Simulations using the Duffing Oscillator

The nonlinear EKF GRAPE framework of chapter 5 was applied to the Duffing oscillator system to estimate the parameters for damping, c, linear stiffness, k, and nonlinear cubic stiffness, β. An appropriate set of parameters was chosen for the true system with parameter and measurement noise introduced to generate a synthetic system with input and output values. Many simulations were executed to characterize the system, tune the EKF models and determine appropriate parameter options for both the SGBS and LHS sampling methods. The SGBS approach will converge close to the true parameter values as long as enough models are used. The model count is the product of the number of strata desired for each dimension. The LHS approach uses a much lower model count and will generally produce similar results. However, the lower model count of the LHS approach results in models potentially further away from the true values. In general, this causes the residual values to be higher. As a result, the LHS GRAPE approach may converge on a model that produces similar results with different parameters. Therefore, caution must be taken to ensure the LHS approach is tuned appropriately. Nonpersistent signals were evaluated to determine if a short pulse or doublet would provide sufficient residual information to update a parameter estimate. The simulation results suggest that the approach is feasible for a doublet as long as the estimate converges quickly in respect to the duration of the doublet. For the configuration explore, a two cycle double should suffice.

CHAPTER 7

Summary and Closing Remarks

7.1 Summary of Research Achievements

This research effort has concluded with the development of a new framework to monitor system health by concurrently estimating the unknown system parameters along with the state estimates. The framework is called GRide-based Adaptive Parameter Estimation (GRAPE). This new approach is a derivative of multiple-model adaptive estimation (MMAE). GRAPE is initially implemented in a linear framework using simple Kalman filter models. A more generalized GRAPE formulation is presented using extended Kalman filter (EKF) models to represent nonlinear systems. GRAPE can handle both time invariant and time varying systems.

The primary contribution of this research is three parts. The first part is the adaptation of MMAE to a formalized framework that supports system health monitoring. In this framework, each parameter dimension is treated independently. The overall limits and range boundaries for each dimension can be set to values that match the chosen FDD approach.

The second research contribution is the way GRAPE handles resampling. Literature has shown that previous research efforts have focused on using predefined banks of filter models. The GRAPE approach samples and resamples from the parameter space according to the desired FDD goals. Each resample

model set is newly generated, not carrying any prior models forward. The range of the new models depends on convergence location of the prior sample set and the GRAPE configuration. Information from the covariance can be store for use during the resampling initialization.

The third contribution is sampling approach itself. GRAPE provides two methods to sample the parameter space. The selected grid-based sampling approach is similar to prior methods presented in literature in that the strata are uniformly divided according to the desired number of models. This approach converges well as long as the number of stratum is sufficient to produce the desired accuracy. The drawback of the SGBS approach is the number of models required is the product of the number of strata for each dimension. The Latin Hypercube Sampling approach was introduced to determine the locations of models and minimize the model count required. This reduces the computational complexity. The LHS sampling approach works well in most cases. However, the LHS approach can converge on similar model combinations producing errors in the parameter estimates. This is due to the sparser spacing of the LHS models and higher residual values. These higher residual values allow the MMAE process to converge on parameters that produce similar state estimates. Care must be taken to ensure that LHS is adjusted through the GRAPE configuration to produce the desired results.

GRAPE can be tuned to match the desired performance goals for initial sampling, resampling and convergence criteria. The method will narrow its focus on progressively smaller dimensional ranges until a preset lower width limit is reached. Additionally, GRAPE will expand each dimensional window as appropriate to track parameters outside of the current range boundaries.

The GRAPE framework has been demonstrated on both linear and nonlinear systems. The linear GRAPE framework was performed on the simple-spring-mass damper system for the estimation of the damping coefficient and spring stiffness coefficient. The nonlinear GRAPE framework was performed on the Duffing oscillator which adds a cubic stiffness term to the harmonic equation for the spring-mass-damper system. The linear framework executes much faster for the same model counts as the equivalent nonlinear framework. The linear models are typically discrete-time Kalman filters (KF) while the nonlinear models are continuous-discrete extended Kalman filters (EKF). The linear discrete-time KFs are simpler and are evaluated with simple matrix multiplication techniques to calculate the next result. The EKFs require the evaluation of Jacobians and integration using an RK4 solver such as the MATLAB ode series of functions.

7.2 Future Work

This research concludes with a working framework to perform FDD using GRAPE. There are many areas to direct future research. Table 7-1 provides a summary of potential future work in this research area.

Table 7-1 Summary of potential future work/modifications to GRAPE

No.	Description
1	Incorporate GMMAE
2	adjust the initial weights when a minimum window is near the limit for a dimension to correct the initial error introduced
3	Halt resampling when input/output have reached steady state
4	Add forced resample coupled with double-cycle doublet to resample during steady-state input/output
5	Prevent resampling when all dimensions are within the center range of the smallest width
6	Investigate GRAPE response to alternate input signals
7	Investigate system similarity and relationship to LHS model spacing to identify when residuals are high enough to allow MMAE to diverge from true value
8	Fully implement hybrid SGBS/LHS sampling
9	Expand for concurrent GRAPE and Neural Network fault signature identification of sensor failures
10	Implement GRAPE on a physical system with real-time tests

The first identified area is to implement the concepts of generalized MMAE (GMMAE) into GRAPE. The core to GRAPE is MMAE. GMMAE improves the likelihood function used to update the MMAE weights. It uses the correlation between measurements by using the past history of the residuals. The current GRAPE algorithm already stores the residuals. It is only a matter of implementing GMMAE to update the core MMAE process. MMAE should improve the rate of convergence according to research in [61], [62], [63] and [64].

The second area to improve GRAPE is the performance near the upper and lower limits of an individual parameter space. Currently, the region is controlled by a minimum width. Within the regions [a,b] and [j,k] of Figure 5-7, the center of the resample region is not the last estimate. This causes an initial error as shown in

the third sample point of Figure 7-1. GRAPE overcomes this error over time. However, skewing the initial MMAE weights at a resample to produce an initial estimate at the last estimate will improve the performance for this case.

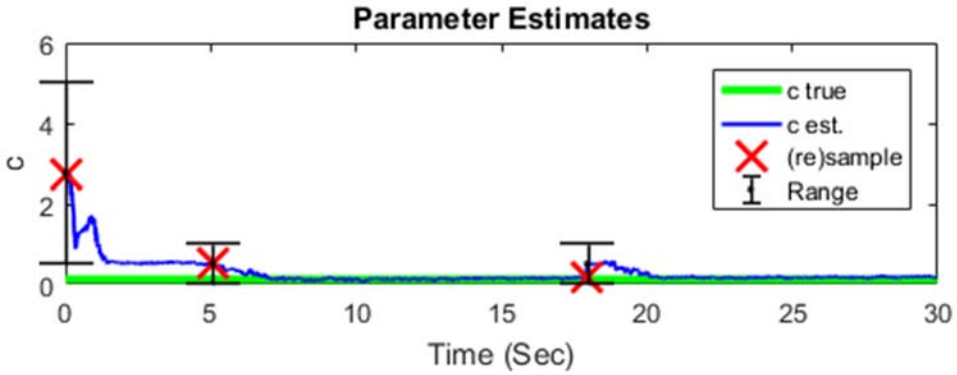


Figure 7-1 System with reasonable interval resampling

The third area to improve GRAPE is to halt resampling when the input and output have become static. (Section 6.4.2 discusses the issues with nonpersistent excitation.) With no input or constant input, the system is stable, and there is no information in the residuals. Therefore, there is nothing for the MMAE process to use to update the models. Therefore, the input/output pair should be examined for static conditions in which model updates should be halted. This will maintain the last estimate and prevent estimate drift.

The fourth area for improvement is to add a formal forced resample feature. This feature was used for debugging purposes while developing GRAPE. It is a valuable tool that should be formalized in the actual GRAPE algorithm. Combining a forced resample with the research already performed on the nonpersistent doublet

signal would be a valuable tool to see if the system has changed while the input is static. A double doublet synthetic input should allow GRAPE to re-identify the system parameters. The forced resample would coincide with the synthetic input.

The fifth area for improvement is to prevent resampling when all dimensional current estimate ranges are both at the minimum width values, and the current estimate is also within the center range region. Identifying this condition and setting a flag would help prevent the needless resampling that is purely controlled by the delay feature after the system has converged near the true value as illustrated in Figure 7-2 starting at approximately 22 seconds.

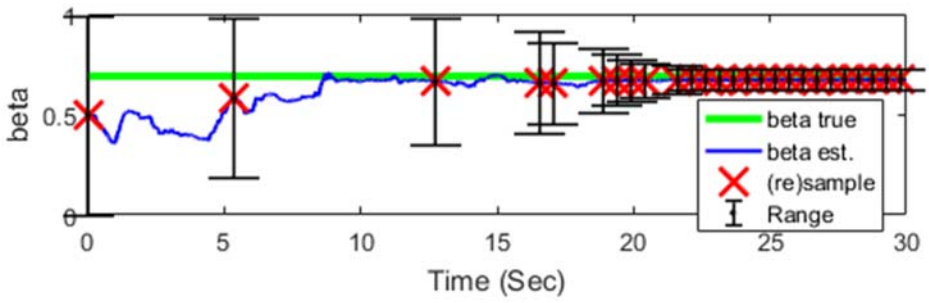
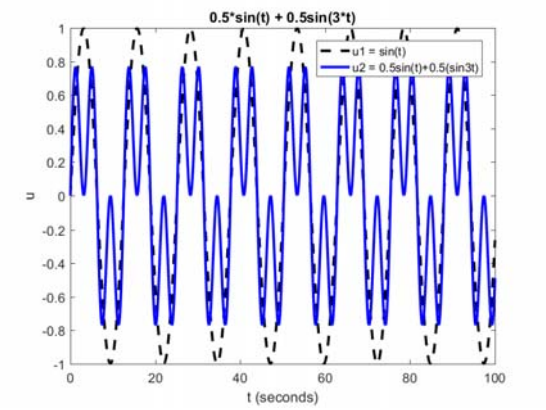


Figure 7-2 Resampling continues after final convergence

The sixth identified area is for future research to investigate alternative input signals. This research focused on three signals, the persistent sinusoidal input, along with the pulse and doublet inputs followed by no input. There is a multitude of other signals to test and study. Additional input signals should be explored. Two such inputs are a combined multiple sinusoidal signal in Figure 7-3(a) and a random amplitude with random duration signal of Figure 7-3(b). The Figure 7-3(a) is of

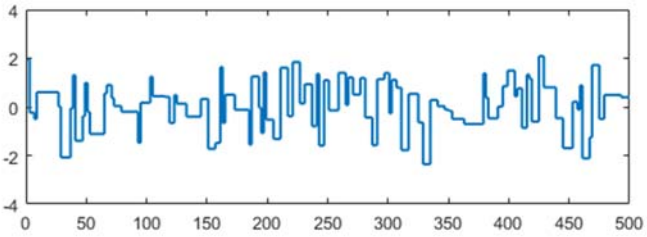
particular interest because it can be represented as a cubic sine function or simple sine functions according to the following trigonometric identity of equation (7-1). Studying this signal would give insight to the behavior of similar input signals. It is hypothesized that similar responses are what cause the rare cases where LHS sampling converges to the wrong system parameters.

$$\sin^3(t) = \frac{3\sin(t) - \sin(3t)}{4} \tag{7-1}$$



(a) alternative sinusoidal input

$$\gamma\sin(\omega t) + \eta\sin(\phi t)$$



(b) random amplitude and duration signal generator of predefined probability

[130] (test of n_random_levels 2.3.2 (pg 40-45))

Figure 7-3 Alternate input signals

The seventh identified area of research is to investigate model similarity and the relationship to sparse LHS sampling and the cases where LHS sampling will converge on a similar model that is away from the true model. As discussed in 6.6.1, this issue is hypothesized to arise from the sparsely populated models of LHS generating higher residuals. The higher residuals allow MMAE to converge on alternate models with similar input/output relationships.

The eighth identified potential improvement is to fully implement the hybrid SGBS/LHS sampling approach. In this hybrid approach, the initial samples would be performed over a wider window using LHS. Once the system has converged to a predefined percentage of the parameter space limits, the next resample would switch to an SGBS approach. The SGBS would continue to narrow its focus. By having uniformly spaced strata only on smaller range windows, the number of models for SGBS could be reduced. Additionally, the SGBS sampling will better track towards the correct parameter when the sparsely placed models of LHS has converged in the wrong region. This should correct the rare cases where the sparse LHS samples converge on a similar model that is outside the parameter range of the true mode.

The ninth identified area of future work is to implement GRAPE on a physical system. Simulations are a method of study and development. However, an actual system would test the capabilities of GRAPE and provide insight for future required research.

The tenth and final area for future research is the area of using a concurrent neural network approach to identify additive sensor fault signatures. This is a hybrid analysis identifying nonlinear fault signatures due to sensor failure using neural networks in conjunction with the GRAPE methods. The approach would recognize the fault signature and subtract the resulting output from the measured output to allow GRAPE to continue to estimate the system parameters. The learned fault could be compared to other faults for identification and reporting. A preliminary literature review was begun on this subject following two main sources, [131, 132]. The ultimate approach to this future research is a hybrid method to combine the GRAPE analysis with neural networks to concurrently estimate the states and system parameters and evaluate potential nonlinear fault signals.

7.3 Closing Remarks

MMAE methods are subjects of renewed interest with the capability of modern computing systems. GRAPE expands on MMAE with the hypothesis that sample models can be applied and resampled without relying on a predefined set of models. GRAPE simply resamples in the region of the prior parameter estimate using one of two methods: the SGBS approach or the LHS approach. GRAPE has generated a framework for FDD. Much potential future work has been identified to expand the capabilities of GRAPE further. The GRAPE algorithm has the versatility for use in wide range of applications outside of the framework for FDD. Potential applications are tracking, system identification and adaptive control.

Appendix A

Acronyms

Table A-1 List of acronyms used

Acronym	Description
ACM	Association for Computing Machinery
AIAA	American Institute of Aeronautics and Astronautics
AEM	abnormal event management
ASME	American Society of Mechanical Engineers
ANOVA	Analysis of variance
CLT	Central Limit Theorem
DGPS	differential Global Positioning System
EKF	Extended Kalman Filter
ENKF	Ensemble Kalman Filter
FDD	Fault Detection and Diagnosis
FTC	Fault Tolerant Control
GLR	Generalized Likelihood Ratio
GMMAE	generalized multiple-model adaptive estimation
GPS	Global Positioning System
GRAPE	Grid-Based Adaptive Parameter Estimation
GS	Static Gain
I/O	Input/Output
IEEE	Institute of Electrical and Electronics Engineers
IID	Independent and Independently Distributed
IMM	Interactive Multiple Model
INS	Inertial Navigation System
KF	Kalman Filter
LLN	Law of Large Numbers
MAP	Maximum A Posteriori Estimation
MBMMAE	Maybeck Moving-Bank Multiple-Model Adaptive Estimation
MDOF	Multiple Degree of Freedom
MLE	maximum likelihood estimation
MMAC	Multiple-Model Adaptive Control
MMAE	Multiple-Model Adaptive Estimator
MMSE	minimized mean square error
MSE	mean squared error
$N(\mu,Q)$	Normal (Gaussian) distribution with mean mu and variance Q
N/A	Not Applicable

Table A-1 List of acronyms used (continued)

ODE	Ordinary Differential Equation
RK4	4^{th} Order Runge-Kutta (variable step size ordinary differential equation solver)
SDOF	Single Degree of Freedom
SGBS	Selected Grid-Based Stratification
SKFB	Standard Kalman Filter Bank
SMA	Simple Moving Average
UAV	unmanned aerial vehicle
UKF	Unscented Kalman Filter

References

[1] Blanke, M., Kinnaert, M., Lunze, J., Staroswiecki, M., and Schröder, J., 2006, "Diagnosis and Fault-Tolerant Control," Springer-Verlag, Berlin, Germany.

[2] Isermann, R., 2006, "Fault-Diagnosis Systems: An Introduction from Fault Detection to Fault Tolerance," Springer-Verlag, Berlin, Germany.

[3] Miljković, D., 2011, "Fault detection methods: A literature survey," 2011 Proceedings of the 34th International Convention on Information and Communication Technology, Electronics and Microelectronics, Opatija, Croatia, MIPRO, pp. 110-115.

[4] Chen, J., and Patton, R.J., 1999, "Robust Model-Based Fault Diagnosis for Dynamic Systems," Springer Science & Business Media, New York.

[5] Stengel, R. F., 1991, "Intelligent Failure-Tolerant Control," IEEE Control Systems, **11** (4) pp. 14-23.

[6] Patton, R. J., 1997, "Fault-tolerant control systems: The 1997 situation," IFAC Symposium on Fault Detection Supervision and Safety for Technical Processes, IFAC SAFEPROCESS '97, Hull, UK, pp. 1033-154.

[7] Venkatasubramanian, V., Rengaswamy, R., Yin, K., and Kavuri, S. N., 2003, "A Review of Process Fault Detection and Diagnosis: Part I: Quantitative Model-Based Methods," Computers & Chemical Engineering, **27** (3) pp. 293-311.

[8] Venkatasubramanian, V., Rengaswamy, R., and Kavuri, S. N., 2003, "A Review of Process Fault Detection and Diagnosis: Part II: Qualitative Models and Search Strategies," Computers & Chemical Engineering, **27** (3) pp. 313-326.

[9] Venkatasubramanian, V., Rengaswamy, R., Kavuri, S. N., and Yin, K., 2003, "A Review of Process Fault Detection and Diagnosis: Part III: Process History Based Methods," Computers & Chemical Engineering, **27** (3) pp. 327-346.

[10] Qi, X., Theillio, D. l., Qi, J., 2013, "Fault diagnosis and fault tolerant control methods for manned and unmanned helicopters: A literature review," 2013 Conference on Control and Fault-Tolerant Systems (SysTol), pp. 132-139.

[11] Dusia, A., and Sethi, A. S., 2016, "Recent Advances in Fault Localization in Computer Networks," IEEE Communications Surveys & Tutorials, **18** (4) pp. 3030-3051.

[12] Ruschmann, M., Wu, N., and Shin, J., 2010, "Actuator Fault Diagnosis Using Two-Stage Extended Kalman Filters," AIAA Guidance, Navigation, and Control Conference, pp. 7703.

[13] Moradi, M., and Fekih, A., 2014, "Adaptive PID-Sliding-Mode Fault-Tolerant Control Approach for Vehicle Suspension Systems Subject to Actuator Faults," IEEE Transactions on Vehicular Technology, **63** (3) pp. 1041-1054.

[14] Yong, J., Hongguang, W., Lijin, F., 2006, "A Combined Logistic and Model Based Approach for Fault Detection and Identification in a Climbing Robot," 2006 IEEE International Conference on Robotics and Biomimetics, pp. 1512-1516.

[15] Yong, J., Hongguang, W., Lijin, F., 2006, "A Novel Approach to Fault Detection and Identification in Suction Foot Control of a Climbing Robot," 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3423-3428.

[16] Stepaniak, M. J., and Maybeck, P. S., 1998, "MMAE-Based Control Redistribution Applied to the VISTA F-16," IEEE Transactions on Aerospace and Electronic Systems, **34** (4) pp. 1249-1260.

[17] Menke, T. E., and Maybeck, P. S., 1995, "Sensor/Actuator Failure Detection in the Vista F-16 by Multiple Model Adaptive Estimation," IEEE Transactions on Aerospace and Electronic Systems, **31** (4) pp. 1218-1229.

[18] Eide, P., and Maybeck, P. S., 1995, "Evaluation of a multiple-model failure detection system for the F-16 in a full-scale nonlinear simulation," Aerospace and Electronics Conference, 1995. NAECON 1995., Proceedings of the IEEE 1995 National, **1,** pp. 531-536 vol.1.

[19] Lane, D. W., and Maybeck, P. S., 1994, "Multiple model adaptive estimation applied to the LAMBDA URV for failure detection and identification," Decision and Control, 1994., Proceedings of the 33rd IEEE Conference on, **1,** pp. 678-683 vol.1.

[20] McNeill, S. S., and Zimmerman, D. C., 2007, "Fault Detection in Open-Loop Controlled Structures," Journal of Guidance, Control, and Dynamics, **30** (5) pp. 1378-1385.

[21] Márton, L., 2015, "Actuator Fault Diagnosis in Mechanical systems—Fault Power Estimation Approach," International Journal of Control, Automation and Systems, **13** (1) pp. 110-119.

[22] Hernández-Alcántara, D., Tudón-Martínez, J. C., Amézquita-Brooks, L., Vivas-López, C. A., and Morales-Menéndez, R., 2016, "Modeling, Diagnosis and Estimation of Actuator Faults in Vehicle Suspensions," Control Engineering Practice, **49** pp. 173-186.

[23] Efimov, D., Zolghadri, A., and Simon, P., 2013, "Improving fault detection by extended kalman filter adjustment for oscillatory failure case in aircrafts," Progress in flight dynamics, guidance, navigation, control, fault detection, and avionics, EDP Sciences, **6,** pp. 393-406.

[24] Belhassine-Cherif, R., and Ghedamsi, A., 2000, "Diagnostic tests for communicating nondeterministic finite state machines," Computers and Communications, 2000. Proceedings. ISCC 2000. Fifth IEEE Symposium on, pp. 424-429.

[25] Varney, P., and Green, I., 2013, "Rotordynamic Crack Diagnosis: Distinguishing Crack Depth and Location," Journal of Engineering for Gas Turbines and Power, **135** (11) pp. 112101.

[26] Hongzhong Ma, Limin Zhang, Huamin Li, 2008, "The application of modal analysis in fault diagnosis of AC motor," Condition Monitoring and Diagnosis, 2008. CMD 2008. International Conference on, pp. 217-220.

[27] Chelidze, D., and Cusumano, J. P., 2004, "A Dynamical Systems Approach to Failure Prognosis," Journal of Vibration and Acoustics, **126** (1) pp. 2-8.

[28] Jia, Y., and Reddy, T. A., 2003, "Characteristic Physical Parameter Approach to Modeling Chillers Suitable for Fault Detection, Diagnosis, and Evaluation," Journal of Solar Energy Engineering, **125** (3) pp. 258-265.

[29] Sheard, A. G., Corsini, A., and Bianchi, S., 2011, "Stall Warning in a Low-Speed Axial Fan by Visualization of Sound Signals," Journal of Engineering for Gas Turbines and Power, **133** (4) pp. 041601.

[30] Zhang, H., Liu, J., and Li, R., 2015, "Fault Detection for Medical Body Sensor Networks Under Bayesian Network Model," 2015 11th International Conference on Mobile Ad-hoc and Sensor Networks (MSN), pp. 37-42.

[31] Charest, J., Beaudoin, J. F., Cadorette, J., Lecomte, R., Brunet, C. A., 2014, "Automatic Channel Fault Detection on a Small Animal APD-Based Digital PET Scanner," IEEE Transactions on Nuclear Science, **61** (5) pp. 2494-2502.

[32] Vega-Hernández, O., Campos-Delgado, D. U., and Espinoza-Trejo, D. R., 2009, "Actuator fault tolerant control for an artificial pancreas," 2009 6th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), pp. 1-6.

[33] Goudar, V., and Potkonjak, M., 2014, "Low-power semantic fault-detection in multi-sensory mobile health monitoring systems," Internet of Things (WF-IoT), 2014 IEEE World Forum on, pp. 32-36.

[34] Jamshidpour, E., Poure, P., and Saadate, S., 2015, "Photovoltaic Systems Reliability Improvement by Real-Time FPGA-Based Switch Failure Diagnosis and Fault-Tolerant DC–DC Converter," IEEE Transactions on Industrial Electronics, **62** (11) pp. 7247-7255.

[35] Mohseni, S., and Namvar, M., 2009, "Fault diagnosis in robot manipulators in presence of modeling uncertainty and sensor noise," 2009 IEEE Control Applications, (CCA) & Intelligent Control, (ISIC), pp. 1750-1755.

[36] McIntyre, M. L., Dixon, W. E., Dawson, D. M., and Walker, I. D., 2005, "Fault Identification for Robot Manipulators," IEEE Transactions on Robotics, **21** (5) pp. 1028-1034.

[37] Zhang, J., Xiong, J., Ren, M. S., Y., and Xu, J., 2016, "Filter-Based Fault Diagnosis of Wind Energy Conversion Systems Subject to Sensor Faults," Journal of Dynamic Systems, Measurement, and Control, **138** (6) pp. 061008.

[38] Doraiswami, R., and Cheded, L., 2013, "Fault Diagnosis of a Sensor Network: A Distributed Filtering Approach," Journal of Dynamic Systems, Measurement, and Control, **135** (5) pp. 051002.

[39] Peng, Z. K., Lang, Z. Q., and Billings, S. A., 2009, "Analysis of Locally Nonlinear MDOF Systems using Nonlinear Output Frequency Response Functions," Journal of Vibration and Acoustics, **131** (5) pp. 051007.

[40] Sohn, H., Farrar, C. R., Hunter, N. F., and Worden, K., 2001, "Structural Health Monitoring using Statistical Pattern Recognition Techniques," Journal of Dynamic Systems, Measurement, and Control, **123** (4) pp. 706-711.

[41] Mjit,M. , 2009, "Methodology for fault detection and diagnostics in an ocean turbine using vibration analysis and modeling," MS, Florida Atlantic University, Boca Raton, Florida.

[42] El Azzouzi, F. A., and Johnson, K., 2012, "Single-sensor-based Fault Detection in Wind Turbines," 50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, pp. 1289.

[43] Meskin, N., Khorasani, K., and Rabbath, C. A., 2009, "Fault diagnosis in a network of unmanned aerial vehicles with imperfect communication channels," AIAA Guidance, Navaigation, and Control Conference, pp. 1-18.

[44] Hu, B., and Seiler, P., 2014, "Certification analysis for a model-based UAV fault detection system," AIAA Guidance, Navigation, and Control Conference, pp. 0610.

[45] Sadeghzadeh, I., and Zhang, Y., 2011, "A review on fault-tolerant control for unmanned aerial vehicles (UAVs)," Infotech@ Aerospace 2011, AIAA 2011-1472, Infotech@Aerospace Conference, St.Louis, MO, pp. 1-12.

[46] Stengel, R.F., 1994, "Optimal Control and Estimation," Dover Publications, New York.

[47] Aguiar, A. P., 2007, "Multiple-model adaptive estimators: Open problems and future directions," Control Conference (ECC), 2007 European, IEEE, pp. 5544-5545.

[48] Vasquez, J. R., and Maybeck, P. S., 2004, "Enhanced Motion and Sizing of Bank in Moving-Bank MMAE," IEEE Transactions on Aerospace and Electronic Systems, **40** (3) pp. 770-779.

[49] Simon, D., 2006, "Optimal state estimation: Kalman, H infinity, and nonlinear approaches," John Wiley & Sons, Hoboken, New Jersey.

[50] Sorenson, H.W., 1985, "Kalman Filtering: Theory and Application," IEEE Press, .

[51] Magill, D., 1965, "Optimal Adaptive Estimation of Sampled Stochastic Processes," IEEE Transactions on Automatic Control, **10** (4) pp. 434-439.

[52] Miller,Mikel M. , 1998, "Modified Multiple Model Adaptive Estimation (M3AE) for Simultaneous Parameter and State Estimation," PhD, Air Force Institute of Technology, Wright-Patterson AFB, Ohio.

[53] Crassidis, J.L., and Junkins, J.L., 2012, "Optimal Estimation of Dynamic Systems," 2nd Ed., CRC Press, Boca Raton, Florida.

[54] Ljung, L., 1999, "System Identification," 2nd Ed., Prentice-Hall, New Jersey.

[55] Steinberg, D.S., 1988, "Vibration Analysis for Electronic Equipment," 2nd Ed., Wiley-Interscience, New York.

[56] Craig, R.R., and Kurdila, A.J., 2006, "Fundamentals of Structural Dynamics," 2nd Ed., John Wiley & Sons, Hoboken, New Jersey.

[57] Karnick, D. A., and Maybeck, P. S., 1987, "Moving bank multiple model adaptive estimation applied to flexible spacestructure control," Decision and Control, 1987. 26th IEEE Conference on, **26,** pp. 1249-1257.

[58] Blasch, E., 2009, "Supervised learning for adaptive interactive multiple model (SLAIMM) tracking," Proceedings of the IEEE 2009 National Aerospace & Electronics Conference (NAECON), pp. 236-243.

[59] Anderson, B., and Moore, J.B., 1979, "Optimal Filtering," Prentice-Hall, Englewood Cliffs, New Jersey.

[60] Sims, F., Lainiotis, D., and Magill, D., 1969, "Recursive Algorithm for the Calculation of the Adaptive Kalman Filter Weighting Coefficients," IEEE Transactions on Automatic Control, **14** (2) pp. 215-218.

[61] Crassidis, J. L., and Cheng, Y., 2006, "Generalized Multiple-Model Adaptive Estimation Using an Autocorrelation Approach," 2006 9th International Conference on Information Fusion, pp. 1-8.

[62] Alsuwaidan, B. N., Crassidis, J. L., and Cheng, Y., 2011, "Generalized Multiple-Model Adaptive Estimation using an Autocorrelation Approach," IEEE Transactions on Aerospace and Electronic Systems, **47** (3) pp. 2138-2152.

[63] Martí, E. D., García, J., and Crassidis, J. L., 2012, "Improving multiple-model context-aided tracking through an autocorrelation approach," Information Fusion (FUSION), 2012 15th International Conference on, pp. 1822-1829.

[64] Alsuwaidan,B. N. , 2008, "Generalized Multiple Model Adaptive Estimation," PhD, State University of New York at Buffalo, Buffalo, New York.

[65] Vasquez, J. R., and Maybeck, P. S., 1999, "Enhanced motion and sizing of bank in moving-bank MMAE," American Control Conference, 1999. Proceedings of the 1999, IEEE, **3,** pp. 1555-1562.

[66] Kun, Q., Xiangping, P., Hao, Y., 2010, "Aircraft Health Monitoring System Using Multiple-Model Adaptive Estimation," 2010 Second WRI Global Congress on Intelligent Systems, **1,** pp. 254-257.

[67] White, N. A., Maybeck, P. S., and DeVilbiss, S. L., 1998, "Detection of Interference/Jamming and Spoofing in a DGPS-Aided Inertial System," IEEE Transactions on Aerospace and Electronic Systems, **34** (4) pp. 1208-1217.

[68] Vasquez, J. R., and Maybeck, P. S., 1999, "Density algorithm based moving-bank MMAE," Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No.99CH36304), **4,** pp. 4117-4122 vol.4.

[69] Abuhashim, T. S., Abdel-Hafez, M. F., and Al-Jarrah, M. A., 2008, "Integrity monitoring of lowcost GPS-aided-INS systems," 2008 5th International Symposium on Mechatronics and Its Applications, pp. 1-9.

[70] Bahadorinejad, A., and Braga-Neto, U., 2015, "Optimal Fault Detection and Diagnosis in Transcriptional Circuits using Next-Generation Sequencing," IEEE/ACM Transactions on Computational Biology and Bioinformatics, IEEE, **PP,** pp. 1-1.

[71] Izadian, A., Khayyer, P., and Famouri, P., 2009, "Fault Diagnosis of Time-Varying Parameter Systems with Application in MEMS LCRs," IEEE Transactions on Industrial Electronics, **56** (4) pp. 973-978.

[72] Rahman, M. A., Anwar, S., and Izadian, A., 2015, "Electrochemical model based fault diagnosis of a lithium ion battery using multiple model adaptive estimation approach," 2015 IEEE International Conference on Industrial Technology (ICIT), pp. 210-217.

[73] Roumeliotis, S. I., Sukhatme, G. S., and Bekey, G. A., 1998, "Sensor fault detection and identification in a mobile robot," Intelligent Robots and Systems, 1998. Proceedings., 1998 IEEE/RSJ International Conference on, **3,** pp. 1383-1388 vol.3.

[74] Maybeck, P. S., and Hentz, K. P., 1985, "Investigation of moving-bank multiple model adaptive algorithms," Decision and Control, 1985 24th IEEE Conference on, pp. 1874-1881.

[75] Erickson, J. W., Maybeck, P. S., and Raquet, J. F., 2005, "Multipath-Adaptive GPS/INS Receiver," IEEE Transactions on Aerospace and Electronic Systems, **41** (2) pp. 645-657.

[76] Griffin, G. C., and Maybeck, P. S., 1997, "MMAE/MMAC Control for Bending with Multiple Uncertain Parameters," IEEE Transactions on Aerospace and Electronic Systems, **33** (3) pp. 903-912.

[77] Hanlon, P. D., and Maybeck, P. S., 1997, "Equivalent Kalman filter bank structure for multiple model adaptive estimation (MMAE) and generalized likelihood ratio (GLR) failure detection," Decision and Control, 1997., Proceedings of the 36th IEEE Conference on, **5,** pp. 4312-4317 vol.5.

[78] Maybeck, P. S., 1989, "Moving-Bank Multiple Model Adaptive Estimation and Control Algorithms: An Evaluation," Control and Dynamic Systems: Advances in Aerospace System Dynamics and Control Systems, Edited by CT Leondes, **V31** pp. 1-28.

[79] Maybeck, P. S., and Schore, M. R., 1992, "Reduced-Order Multiple Model Adaptive Controller for Flexible Spacestructure," IEEE Transactions on Aerospace and Electronic Systems, **28** (3) pp. 756-767.

[80] Maybeck, P. S., and Schore, M. R., 1990, "Robustness of a moving-bank multiple model adaptive algorithm for control of a flexible space structure," IEEE Conference on Aerospace and Electronics, pp. 368-374 vol.1.

[81] Lashlee, R. W., and Maybeck, P. S., 1988, "Space structure control using moving bank multiple model adaptive estimation," Proceedings of the 27th IEEE Conference on Decision and Control, pp. 712-717 vol.1.

[82] Griffin, G. C., and Maybeck, P. S., 1995, "MMAE/MMAC techniques applied to large space structure bending with multiple uncertain parameters," Proceedings of 1995 34th IEEE Conference on Decision and Control, **2,** pp. 1153-1158 vol.2.

[83] Schiller, G. J., and Maybeck, P. S., 1997, "Control of a Large Space Structure using MMAE/MMAC Techniques," IEEE Transactions on Aerospace and Electronic Systems, **33** (4) pp. 1122-1131.

[84] Fitch, J. A., and Maybeck, P. S., 1994, "Multiple model adaptive control of a large flexible space structure with purposeful dither for enhanced identifiability," Proceedings of 1994 33rd IEEE Conference on Decision and Control, **3,** pp. 2904-2909.

[85] Gustafson, J. A., and Maybeck, P. S., 1994, "Flexible Spacestructure Control Via Moving-Bank Multiple Model Algorithms," IEEE Transactions on Aerospace and Electronic Systems, **30** (3) pp. 750-757.

[86] Gustafson, J. A., and Maybeck, P. S., 1992, "Control of a large flexible space structure with moving-bank multiple model adaptive algorithms," 1992 Proceedings of the 31st IEEE Conference on Decision and Control, pp. 1273-1278.

[87] S. N. Sheldon, and P. S. Maybeck, 1993, "An Optimizing Design Strategy for Multiple Model Adaptive Estimation and Control," IEEE Transactions on Automatic Control, **38** (4) pp. 651-654.

[88] Greenberg, M.D., 1998, "Advanced Engineering Mathematics," 2nd ed., Prentic Hall, New Jersey.

[89] Woods, R.L., and Lawrence, K.L., 1997, "Modeling and Simulation of Synamic Systems," Prentice-Hall, Upper Saddle River, New Jersey.

[90] Dorf, R.C., and Bishop, R.H., 2001, "Modern Control Systems," 9th Ed., Prentice Hall, New Jersey.

[91] Ogata, K., 2010, "Modern Control Engineering," 5th Ed., Prentice Hall, Boston.

[92] Franklin, G.F., Powell, J.D., and Workman, M.L., 1998, "Digital Control of Dynamic Systems," 3rd Ed., Addison-Wesley, Menlo Park.

[93] Friedland, B., 2005, "Control System Design: an Introduction to State-Space Methods," Dover Publications, Mineola, New York.

[94] Palm, W.J., 2000, "Modeling, Analysis, and Control of Dynamic Systems," 2nd Ed., John Wiley and Sons, New York.

[95] Maybeck, P.S., and Siouris, G.M., 1979, "Stochastic Models, Estimation, and Control, Volume I," Academic Press, New York.

[96] Maybeck, P.S., 1982, "Stochastic Models: Estimation and Control, Volume II," Academic Press, New York.

[97] Vasquez,J. R. , "Moving-Bank Multiple Model Adaptive Estimation and Failure Detection," PhD Research Prospectus, School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH, Unpublished.

[98] Anderson, H. L., 1986, "Metropolis, Monte Carlo and the MANIAC," Los Alamos Science, **14** pp. 96-108.

[99] Metropolis, N., 1987, "The Beginning of the Monte Carlo Method," Los Alamos Science, **15** (584) pp. 125-130.

[100] Ulam, S., Richtmyer, R. D., and von Neumann, J., 1947, "Statistical Methods in Neutron Diffusion," Los Alamos Scientific Laboratory report, LAMS-551, .

[101] Metropolis, N., and Ulam, S., 1949, "The Monte Carlo Method," Journal of the American Statistical Association, **44** (247) pp. 335-341.

[102] Clauset, A., 2011, "A Brief Primer on Probability Distributions," Santa Fe Institute. Http://Tuvalu.Santafe.Edu/~aaronc/Courses/7000/csci7000-001_2011_L0.Pdf, (unpublished) .

[103] Walpole, R.E., Myers, R.H., Myers, S.L., and Ye, K., 2012, "Probability and Statistics for Engineers and Scientists," 9th. Ed., Prentice Hall, Boston.

[104] Botev, Z., 2016, "The Normal Law Under Linear Restrictions: Simulation and Estimation Via Minimax Tilting," Journal of the Royal Statistical Society: Series B (Statistical Methodology), (http://arxiv.org/pdf/1603.04166v1.pdf) .

[105] Chopin, N., 2011, "Fast Simulation of Truncated Gaussian Distributions," Statistics and Computing, **21** (2) pp. 275-288.

[106] Kernler, D., 2014, "A visual representation of the Empirical (68-95-99.7) Rule based on the normal distribution," image file, CC BY-SA 4.0, https://en.wikipedia.org/wiki/Normal_distribution#/media/File:Empirical_Rule.PNG, .

[107] Owen, A.B., 2013, "Monte Carlo theory, methods and examples," (http://statweb.stanford.edu/~owen/mc/), draft textbook.

[108] McKay, M., and Beckman, R., 1979, "A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code," Technometrics, **21** (2) .

[109] Owen, A. B., 1998, "Latin Supercube Sampling for very High-Dimensional Simulations," ACM Transactions on Modeling and Computer Simulation (TOMACS), **8** (1) pp. 71-102.

[110] Stein, M., 1987, "Large Sample Properties of Simulations using Latin Hypercube Sampling," Technometrics, **29** (2) pp. 143-151.

[111] Patterson, H., 1954, "The Errors of Lattice Sampling," Journal of the Royal Statistical Society.Series B (Methodological), pp. 140-149.

[112] Helton, J. C., and Davis, F. J., 2003, "Latin Hypercube Sampling and the Propagation of Uncertainty in Analyses of Complex Systems," Reliability Engineering & System Safety, **81** (1) pp. 23-69.

[113] Nakayama, M. K., 2011, "Asymptotically Valid Confidence Intervals for Quantiles and Values-at-Risk when Applying Latin Hypercube Sampling," International Journal on Advances in Systems and Measurements, **4** .

[114] Nakayama, M. K., 2010, "Confidence Intervals for Quantiles When Applying Latin Hypercube Sampling," 2010 Second International Conference on Advances in System Simulation, .

[115] Nakayama, M. K., 2012, "Confidence intervals for quantiles when applying replicated Latin hypercube sampling and sectioning," Proceedings of the 2012 Autumn Simulation Conference, .

[116] Nakayama, M. K., 2014, "Confidence Intervals for Quantiles using Sectioning when Applying Variance-Reduction Techniques," ACM Transactions on Modeling and Computer Simulation (TOMACS), **24** (4) pp. 19.

[117] Dong, H., and Nakayama, M. K., 2014, "Constructing confidence intervals for a quantile using batching and sectioning when applying Latin hypercube sampling," Proceedings of the Winter Simulation Conference 2014, IEEE, pp. 640-651.

[118] Viana, F. A., 2013, "Things you wanted to know about the Latin hypercube design and were afraid to ask," 10th World Congress on Structural and Multidisciplinary Optimization, ISSMO, Orlando, FL, pp. 1-9.

[119] Allen, T.T., 2011, "Introduction to discrete event simulation and agent-based modeling: voting systems, health care, military, and manufacturing," Springer Science & Business Media, London.

[120] Jones, B., and Johnson, R. T., 2009, "Design and Analysis for the Gaussian Process Model," Quality and Reliability Engineering International, **25** (5) pp. 515-524.

[121] Matala, A., 2008, "Sample Size Requirement for Monte Carlo Simulations using Latin Hypercube Sampling," Helsinki University of Technology, Department of Engineering Physics and Mathematics, Systems Analysis Laboratory, .

[122] Manache, G., and Melching, C., 2007, "Sensitivity of Latin Hypercube Sampling to sample size and distributional assumptions," Proceedings of the 32nd Congress of the International Association of Hydraulic Engineering and Research, IAHR, Venice, Italy, .

[123] McKay, M. D., 1988, "Sensitivity and Uncertainty Analysis using a Statistical Sample of Input Values," Uncertainty Analysis, pp. 145-186.

[124] Hickernell, F.J., Jiang, L., Liu, Y., 2013, "Monte Carlo and Quasi-Monte Carlo Methods 2012,"Springer, pp. 105-128.

[125] Rey-Bellet, L., 2016, "Chapter 6: Variance, the Law of Large Numbers and the Monte-Carlo Method," Lecture Notes for Mathematical Modeling: Math 456-02, (http://people.math.umass.edu/~lr7q/m456-spring2016/m456home.html) .

[126] Chapra, S.C., and Canale, R.P., 2010, "Numerical Methods for Engineers," 6th Ed., McGraw-Hill, New York.

[127] Duffing, G., 1918, "Erzwungene Schwingungen bei veränderlicher Eigenfrequenz und ihre technische Bedeutung," R, Vieweg & Sohn, Braunschweig.

[128] Korsch, H.J., Jodl, H., and Hartmann, T., 1994, "Chaos: A Program Collection for the PC,"Springer-Verlag, Berlin, pp. 157-180, Chap. 8.

[129] Kovacic, I., and Brennan, M.J., 2011, "The Duffing equation: nonlinear oscillators and their behaviour," John Wiley & Sons, West Sussex, UK.

[130] Nørgård, P.M., Ravn, O., Poulsen, N.K., and Hansen, L.K., 2000, "Neural Networks for Modelling and Control of Dynamic Systems-A Practitioner's Handbook," Springer, London.

[131] Vemuri, A. T., and Polycarpou, M. M., 1998, "On the use of on-line approximators for sensor fault diagnosis," Proceedings of the 1998 American Control Conference. ACC (IEEE Cat. No.98CH36207), **5,** pp. 2857-2861 vol.5.

[132] Vemuri, A. T., and Polycarpou, M. M., 2004, "A Methodology for Fault Diagnosis in Robotic Systems using Neural Networks," Robotica, **22** (04) pp. 419-438.

Biographical Statement

Ryan Sifford has worked in the aerospace industry since his days as an engineering coop at Chrysler Technologies starting in 1995. Ryan graduated from Texas Tech University in 1998 with a Bachelor of Science degree in Mechanical Engineering and another degree in Computer Science. After graduation, Ryan worked for 10 years at Rockwell Collins primarily focused on airborne platform integration of communication and navigation systems. In 2006, Ryan completed his Master of Science degree in Mechanical Engineering at the University of Texas at Arlington (UTA) under the advisement of Dr. Robert Woods. Upon graduation, he began his Ph.D. studies at UTA under the advisement of Dr. Kamesh Subbarao in the Aerospace Systems Laboratory. In 2008, Ryan left Rockwell Collins to work at Raytheon in the Airborne Processors business area where he has been employed since. In the fall of 2015, Raytheon sponsored Ryan to focus full time on completing his Ph.D. at UTA. Ryan's research focused on methods to evaluate system health using multi-model adaptive estimation techniques to estimate system parameters and states concurrently. Ryan has also taught an undergraduate section of Numerical Analysis and Programming at UTA regularly since the fall 2015 semester. He concluded his research effort in December of 2016 and received a Doctor of Philosophy degree in Mechanical Engineering with a certificate in unmanned vehicle system technologies and applications. Ryan will return to full-time employment at Raytheon in January of 2017.