DESIGN OF FAULT TOLERANT CONTROL SYSTEMS

by

Benjamin Erik Walker

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Electrical Engineering

Charlotte

2012

Approved by:

_____
Dr. Bharat Joshi

_____
Dr. Yogendra Kakad

_____
Dr. James Conrad

_____
Dr. Mehdi Miri

_____
Dr. Edgar Munday

ABSTRACT

BENJAMIN ERIK WALKER. Design of fault tolerant control systems.
(Under the direction of DR. BHARAT JOSHI)

This research designs a Fault Tolerant Control (FTC) approach that compensates for both actuator and sensor faults by using multiple observers. This method is shown to work for both linear time-variant and linear time-invariant systems. This work takes advantage of sensor redundancy to compensate for sensor faults. A method to calculate the rank of available sensor redundancy is developed to determine how many independent sensors can fail without losing observability. This rank is the upper bound on the number of simultaneous sensor failures that the system can tolerate. Based on this rank, a series of reduced order Kalman observers are created to remove sensors presumed faulty from the internal feedback of the estimators.

Actuator redundancy is examined as a potential way to compensate for actuator faults. A method to calculate the available actuator redundancy is designed. This redundancy would allow for the correction of partial and full actuator failures, but few systems exhibit sufficient actuator redundancy. Actuator faults are instead tolerated by replacing the Kalman estimators with Augmented State Observers (ASO). The ASO adds estimates of the actuator faults as additional states of the system in order to isolate and estimate the actuator faults. Then a supervisor is designed to select the observer that correctly identifies the sensor fault set. From that observer, the supervisor collects state estimates and calculates estimates of the sensors and faults. These estimates are then used in feedback with a controller that performs pole placement on the original system.

ACKNOWLEDGEMENTS

I wish to express my gratitude to Dr. Kakad who taught me the concepts of control theory and advised me throughout my MSEE and Ph.D.

I also want to thank my dissertation committee members, Drs. Joshi, Kakad, Miri, Conrad, and Munday for their comments and suggestions with respect to this work.

Lastly, I owe my greatest gratitude to my parents whose continued support made the completion of this dissertation possible.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

FTC	Fault Tolerant Control
FDI	Fault Detection and Isolation
ASO	Augmented State Observer
r-ASO	Reduced order Augmented State Observer
Φ	State transition matrix
$y\binom{k}{p}$	Permutation of y, choosing up to k elements from the total, p
$\hat{x}$	Estimate of x
$\dot{x}$	Derivative of x
$f(t)$	Sensor fault vector
$v(t)$	Actuator fault vector
$e_x(t)$	Error in the state estimate
$e_y(t)$	Error in the output estimate
R_O, RR_O	Vectors that measure the rank of available sensor redundancy in the single fault and multiple fault cases respectively
R_C, RR_C	Vectors that measure the rank of available actuator redundancy in the single fault and multiple fault cases respectively
r	Rank of sensor/actuator redundancy being tested
k	Rank of available sensor redundancy
$\|v(t)\|$	Norm of vector $v(t)$
$a .* b$	Dot product of vectors a and b
R^n	An n dimensional space of real numbers

i	A subset of the sensors that are presumed faulty
$\bar{\imath}$	A subset of the sensors that are presumed fault free
A_a	The subscript a signifies a matrix or vector that has been altered to correspond to an augmented state observer
C_i	The subscript i signifies a matrix or vector that only contains elements that correspond to the set i, other elements are set to zero or removed
$C_{\bar{\imath}}$	The subscript $\bar{\imath}$ signifies a matrix or vector that only contains elements that correspond to the set $\bar{\imath}$, other elements are set to zero or removed
$\bigcup(e_x(t))$	Union of the state error estimates across all of the observers

CHAPTER 1: INTRODUCTION

1.1: Fault Tolerant Control Theory

The control system design process involves the use of feedback to modify the plant response so that the system closely tracks the reference input, minimizes the sensitivity of the system response to system parameter variations, and renders system response insensitive to any disturbance to the system. In order to implement feedback, the system response is conditioned in the feedback loop and compared with the desired signal. The system response is commonly referred to as output and is measured by sensors. The error signal generated from the comparison of the reference signal and the output signal is utilized in some cases as the actuating signal. However, further design process involves adding additional hardware to modify the error signal to generate the actuating signal for the actuator. This additional hardware is commonly called compensators or controllers. The layout of these elements is shown in Figure 1.1.1.

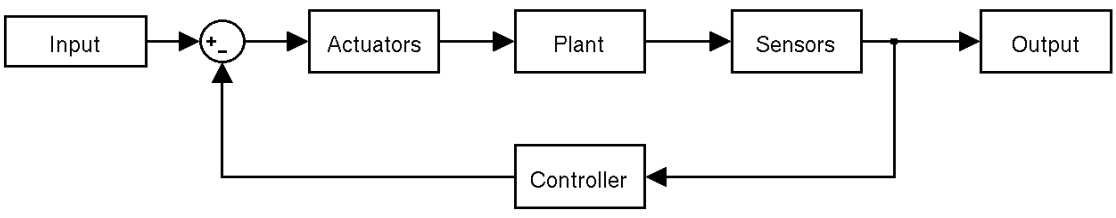

Figure 1.1.1: Diagram of a system with a controller.

The primary function of a controller is to ensure stability of the closed loop system and meet performance needs. With the increase in the complexity of systems and the ever increasing need for more stringent performance requirements, the reliability of the actuators and sensors is of utmost importance. Thus, detecting faults in the sensors and actuators is an area of current research interest in the study of control systems.

Faults occur in many components of the system. Some of the common causes of faults in a system are plant model errors, sensor noise, and actuator wear. Some faults result in minor errors that are tolerable, but eventually faults will prevent a system from performing acceptably or cause it to undergo unstable behavior. To overcome this problem a fault tolerant control (FTC) system is designed.

Fault tolerant controllers must be able to handle multiple categories of faults and errors. There are fundamentally two ways to achieve this. Fault tolerant systems can be designed to be robust enough to perform correctly in the presence of faults. Robustness is a measure of a system's ability to meet performance needs in the presence of errors. This type of method is known as passive FTC. Passive FTC requires less information about the system and the faults than other methods. This makes the design process easier, but it isn't able to handle strong faults. The systems designed by this method are rarely designed to estimate the faults; being only designed to tolerate them.

The simplest passive fault tolerant system is a unity feedback controller. Unity control uses comparative feedback to create a single closed loop. When the system's response fails to meet the desired level, feedback forces it to adjust proportional to the difference. This type of controller is one of the easiest to design. Unfortunately, it is limited in the type of faults it can tolerate and it can't handle faults in the controller.

A more robust method is to design multiple controllers and multiple loops, so that if one of the controllers fails, the system can be corrected via the other controllers. This requires the controllers to be built to fail into the open state, rather than an unbounded failure. This type of failure is common in high stress environments.

However, smarter systems can detect the presence of a fault, identify it, and dynamically compensate for it. Thus they remove the faulty information. Fault detection and isolation (FDI) based methods are effective at achieving fault tolerance. These methods are known as active fault tolerant systems, as they estimate the fault signals or reconfigure themselves to compensate for the faults. While both active and passive methods are able to compensate for faults, active systems are known for performing the full trio of fault detection, fault identification, and fault isolation.

Active systems usually consist of multiple layers. One layer handles the normal operations of the system. Additional layers are added to handle fault detection and isolation. Some active methods rely on the fault information being fed into the controller which modifies its response accordingly. Other methods reconstruct fault-free information that they pass to the controller instead of the plant's faulty outputs. Most active methods produce fault estimates that are passed to other systems or to the user. Having a measure of the faults is useful in many critical systems.

For these reasons, a passive FTC is considered quicker and easier to design, but is less powerful. An active FTC is stronger, but is more difficult to design. Most FDI methods rely heavily upon an accurate model of the plant. Without a complete model of the plant, FDI is complicated by the system improperly classifying un-modeled plant behavior as faults. This problem of modeling error can be solved by designing higher

order models with greater accuracy. Model imperfection is inevitable in any real world plant. Inaccuracies in measurements and imperfections in the fabrication process cause the actual dynamics to deviate from the model of the plant. FDI methods must accept this potential error and avoid improperly classifying it as a fault. To avoid this, some FDI methods do not directly construct a plant model.

FTC is a powerful field of work, bringing increased stability and performance as well as robustness and fault identification. A review of definitions of faults and errors is discussed in detail in section 1.2. Section 1.3 includes a review of various methods that perform FTC and FDI, both actively and passively.

1.2: Fault Classification and Modeling

Modern plants are complicated and modeled as high order systems. Fault tolerant controllers are needed to maintain performance and stability in the presence of faults. To assist in this process, plant faults are classified into categories. Some faults are actual errors, while others are considered failures or disturbances. Each type is described to enable correction and tolerance. However, most methods of FTC are only able to handle certain subsets of fault types.

A type of fault present in nearly all systems is known as modeling error. Modeling error refers to flaws in the design of the model, such as using a linear model to estimate a nonlinear plant. This type of error is regularly caused by using reduced order plants to deal with a high order system. Many systems are too complicated to model both efficiently and correctly. As such, many variables are often omitted to make for a simpler model. The difficulty in compensating for these errors is that the analytical model to represent the error is usually of very high order. This type of error causes a lot of difficulty with passive FTC systems. The errors caused by model estimation lead to faults that can adversely affect the stability of the system if left unchecked.

An alternative to model error that interferes with the effectiveness of the controller is drift error which is internal to the instrumentation. Drift errors are typically caused by the components of a system being subjugated to wear and subsequently not performing up to specification. These errors are often multiplicative in nature, although they are usually modeled as additive faults. The impact of this type of fault can be reduced with regular testing and maintenance of the system's equipment. It is also the second easiest type of error to fix with FTC.

There are also faults caused by external sources. One type of external fault that is present in all real world systems is noise. Noise describes stochastic errors in the system. Noise is typically associated with measurements, but every component suffers noise in some aspect. It is a high frequency error and not predictable by analytical representation. Modern designs often classify modeling errors as noise to simplify the mathematics [21], [38]. While noise is impossible to predict deterministically, its high frequency and low power makes it easy to compensate for with FTC methods.

Another external source of faults is disturbance. Disturbance is a low to mid frequency signal that is an unexpected input to the system. Disturbance signals are deterministic in nature, so unlike noise they can be modeled. An example of a disturbance for an airplane could be crosswind. Disturbances are typically an additive unknown fault. Many systems will lump all faults, from disturbances to model errors, into some form of unknown fault input. As systems can be designed to predict these faults, they can be designed to estimate and correct for them. There are a wide variety of methods to handle these additive and deterministic disturbances.

Most faults are handled as a form of disturbance. Many controllers are designed to predict them and correct for their presence. There are various kinds of disturbances to examine, each with different properties. For example a locked fault, or full fault, is when a component takes on a fixed value instead of its normal dynamics. A subtype of locked fault is when this value is zero, which is known as an open fault. An open fault occurs when portions of the system are no longer connected to each other due to failure. Full faults can be detected and isolated by nearly all fault estimators, given enough time.

Partial faults refer to the class of faults where the original signal has been modified by an external signal. This is often modeled as an additive fault signal. There are some techniques that model partial faults as multiplicative fault signals. Most disturbances, faults, and errors are partial faults. FDI methods rely on additive properties to isolate the fault from the signal. Once this isolation has been done, the fault and corrected signal can be independently passed to other system components.

It is difficult to compensate for intermittent faults. They are deterministically defined faults, with a stochastically defined presence. The fault's presence is not guaranteed after its emergence. This means that fault identification methods must converge upon the fault signal quickly. They also must quickly return to nominal behavior when the fault disappears. If the FDI system is too slow to correct for the presence of the fault, it can lead to oscillatory behavior. FTC methods that focus on robustness or speed are needed to handle intermittent faults.

Linear faults are straightforward and the effect of the fault is easy to quantify. When dealing with nonlinear errors, complications abound. Nonlinear systems do not readily abide by the conveniences of linear time-invariant systems. Nonlinear coupling of the internal states magnifies errors, making the speed of fault detection and isolation highly critical. In addition, most FDI designs are not mathematically complete in the presence of nonlinearities [35].

1.3: Fault Tolerant Methods

Fault tolerant control is a broad field that has seen a lot of growth in the past few decades. Multiple techniques are described in literature to achieve performance and stability in the presence of faults. No single technique has been shown to be optimal in all cases. Each one has its own strengths and weaknesses that must be considered. Some require large amounts of information about the plant or potential faults. Others do not. This section will review many of the popular techniques.

Nearly any FTC method can handle static faults, if given enough time. In general, FDI's are defined by the speed of their fault detection. In a system with low noise, Linear Matrix Inequality (LMI) based methods can be used to make a very fast fault detector. The problem with this method is that the system's quick response leads to improper classification of noise as a fault. Results of LMI based designs should be monitored to handle the problem of chatter caused by over-correction. Chatter is caused when a system over corrects for small disturbances from the target, leading to oscillations around the target output. This speed of fault correction is very important to a robust system, especially in the presence of intermittent faults. LMIs are often combined with another system to reduce the impact of noise and chatter [36].

Another technique to reduce the difficulty of fault identification and increase the speed of an LMI design is to use Linear Fractional Transformation (LFT). LFT takes the plant and controller models and subdivides them to better facilitate system analysis. Once the plant has been disassembled by LFT, it is easier to isolate the faulty component. Once a fault has been isolated, the system must correct for the effect of the fault. Based on the fault's interaction with the LFT, the controller can be dynamically adjusted. This

method is commonly used in gain scheduling, by taking advantage of the controller's dynamic design. This improves fault isolation and ensures that only necessary portions of the system are modified when in the presence of faults [15].

LFTs can also be used to isolate fault dynamics. When specific faults are known to occur in a system, an LFT can be designed to model the fault. This form of adaptive modeling is well suited to sensor drift faults. Normally, these faults are treated as additive errors. With LFT, they can be estimated independently of the rest of the system. This gives a detailed study of how the sensor and actuator faults interact with the stability and the performance of a system results. This study results in a better tuned FDI scheme that can take advantage of the knowledge of the system dynamics. However, the plant's model must be accurate in order to perform fault modeling effectively. Another strength of using LFT in this way is that it can be applied to specific kinds of nonlinear errors, by taking advantage of a dual layered FDI system [27].

A plant model is not always available. A simple method to deal with this situation is to use input/output (I/O) plant modeling and matching. First, an I/O based model is created by observing the plant while it is operating under normal fault free conditions. Once designed, the I/O generated model is compared to the real world responses over time. Isolation of which actuators or sensors are non-functioning is performed by examining the differences in the results. A strength of this type of FTC is that the system can be constructed over an unknown plant. It is limited in the form of faults that it can tolerate and is slower to converge than most FTC systems [4], [22], [23]. These techniques can be adapted to handle nonlinear systems.

Active Disturbance Plant Control (ADPC) is such a design field. Instead of attempting to fully model a complicated nonlinear time-varying system such as an aircraft, the order of the system is used to determine how it will generally respond, and everything beyond that is treated as a disturbance. By designing ADPC to minimize the effect of this disturbance in addition to the plant error, productive results are achieved with limited knowledge of the plant. While these designs aren't able to identify faults, they are very effective at removing them from the system, by correcting for them without needing to identify them. This is very handy when plants operate in a wide range of modes or are so complicated that practical models are filled with errors due to simplification [12].

One of the key differences between FTC and FDI is that the later performs fault identification. Isolating faulty behavior from model errors or noise is a difficult challenge. For that reason, many fault tolerant methods treat all forms of error as faulty behavior. These systems do not perform fault identification. Some of them do not perform fault detection. FTC methods of this nature are designed to be very robust. Robustness is a term that describes a system's ability to continue to operate within specified parameters in the presence of errors and faults. A robust controller is one that modifies the plant so that it resists disturbances and noise while maintaining stability and performance.

Work has gone into finding a design method that produces an optimal level of robustness. Determining a measure that optimizes robustness is complicated, but two have emerged. The H_2 and $H\infty$ methods are considered optimal by many designers. These two methods are designed to minimize the H_2 and $H\infty$ norms respectively. The H_2

norm examines the root-mean-squared result of the impulse response of the system. Effectively, H_2 seeks to minimize the power response of the system. By contrast, the H∞ norm examines the peak response of the system for all frequencies. H∞ optimal control seeks to minimize that peak. These two control problems provide slightly different ways to design optimal controllers for a plant. Controllers designed by these methods are known for their robustness and ability to withstand modeling errors, disturbances, and faults [6]. Their weakness as an FTC scheme is that they do not detect or identify faults. They are designed to keep the plant operating in the presence of any number of errors.

As has been shown, some systems are robust enough to not need direct fault observation. The tradeoff is that robustness usually hides faults from the supervisor. An alternative to this problem is fault observing. In many functionally complete FTC systems, a secondary fault observer can be designed. This is very handy in the case of nonlinear systems. In nonlinear plants, fault correction is often solved by piecewise linearization and robust controllers that are adjusted dynamically. This greatly increases modeling errors, especially when the system's operational point is far from the point at which it was linearized. Certain forms of nonlinearity do not respond well to this form of linearization, such as hysteresis and saturation. Fault detection is complicated by mathematical artifacts from the linearization process. Observers that instead return to the nonlinear model are very effective at avoiding the complications of linearization. Observers can rely on their open-loop definition to relax design conditions, such as the Lipschitz condition. This may increase the observer's complexity, but it produces better results on nonlinear plants [14].

The controllers, observers, and estimators are not immune to the effects of faults. Advanced systems must take into account that the control system that is built to correct for faults is also vulnerable to them. One way to compensate for this is by introducing redundancy into the controller. Dual Loop Fault Tolerant Control (DL-FTC) schemes are built around the idea of redundancy in the fault detection and correction mechanics. Multiple redundant fault detectors watch for errors in the fault compensator. Supervisor systems signal the user when the faults being corrected by the compensator are not detected by other subsystems. The advantage of dual loop systems is that fault detectors are regularly cheaper and more robust than fault compensators. Another advantage is that the additional detector layers can be added to any system, without impacting the design or operation of the system in normal operations modes [5].

The logical conclusion of DL-FTC is to create independent FTC components to monitor each input and/or output. This makes fault isolation simple. If a fault is detected in a single component, then that is also where the fault is located. These FDI systems require a supervisor subsystem that is capable of reconfiguring the controller based on which actuators and sensors are still functioning. A system like this works best when the supervisor is able to disconnect individual faulty systems from the plant's dynamics. This FTC system requires a high degree of component redundancy in both in the sensors and actuators [2].

The highest levels of redundancy can be found in Wireless Signal Networks (WSN). In a WSN, each individual node makes its own opinion about the local data it observes and then transmits to a master node. This master node aggregates all the data and compiles results. This type of system has an immense level of redundancy which

makes many FDI schemes highly effective. When an individual node is found to repeatedly rank poorly on the aggregate data accuracy metrics, the master node can flag its information as faulty or even send it a request to perform a self diagnostics and test [32].

In many applications, there are multiple versions of very similar plants being operated independently of each other. The cost for an engineer to design a high quality FTC system for a single plant often makes it impractical to have them repeat the process on a series of similar plants in a factory. Instead, the engineer produces one full design that is applied to all plants with a simple controller appended to each system, such as a Proportional-Integral-Differential (PID) controller. Then the design can be tuned slightly based on the specifics of the individual plant. Work has been done to simplify the PID even further; reducing it to a bandwidth based tuning system. Based on the specifications of the similar plants and their needs for noise rejection and command following, a simple PID tuning parameter is laid out. This allows a high quality design to be transferred to similar plants. Modifying an existing similar design rather than creating a new one does impact the plant's performance, but it leads to a significant reduction of costs spent on design [11].

The design of FTC is tailored to deal with the types of faults that are meant to be corrected. Many FTC methods are designed to isolate and remove faults from the system before feedback. However, in the case of actuator and plant faults, this is often not possible. Internal feedback complicates fault isolation in most plants. To correct for this, adaptive controllers are designed that take additional inputs from the fault identification subsystems. With the location and strength of the faults identified, the controller can

more efficiently modify the control vector to compensate for them. Modifying the controller in this way is very effective in dealing with actuator faults [3].

Additionally, changes to the plant's dynamics can be fed to the controller to improve its ability to correct failures. There are various methods to maintain the performance of a failing system, once a fault has been detected. A solid approach is the Pseudo Inverse Method (PIM). In this method, the alterations are fed to a controller which is designed to reconstruct the original closed-loop definition of the system. This is not guaranteed to be an unique solution. Therefore, the PIM's reconstruction table must be built beforehand and stored with the adaptive controller. Unfortunately, an adaptive controller built by PIM does not guarantee stability after modification. A method to determine if a Single Input/Single Output (SISO) modification will maintain system stability has been found. However, a Multiple Input/Multiple Output (MIMO) extension has not been completed [10].

Tracking changes to the plant's closed loop performance is one way to keep a system functioning. But even with methods like PIM, each error makes further errors harder to find. Once the system's dynamics are changed in a closed loop, errors will compound themselves. The speed of the FDI system becomes vital. The longer faulty data is in the feedback loop, the greater the performance and stability loss. It is far simpler to maintain good FDI in sensors due to the ease of isolating the faulty signals. In presence of a fault, the faulty sensor signal is disconnected from the feedback and controller. This process is difficult in the case of actuator failures. Actuator failures are insulated from observation by the plant, which compounds the difficulty of correction

[20]. A quality FDI system coupled with regular maintenance greatly reduces actuator and sensor faults.

No matter how well they are maintained, as plants operate they change their behavior. Some changes are caused by faults, modeling errors, or nonlinearities. Sliding Mode Observers (SMO) are a way to deal with faults while allowing for changes in the plant dynamics. The advantage of an SMO is that there is no need for a plant model. The SMO takes a slice of data from the recent history of the plant's inputs and outputs and generates a model based solely on that information. Unusual data patterns and behaviors are classified as faulty and reported as such. As time advances, the data is updated and the dynamic model changes. The strength of the SMO is also its weakness. It does not use any information about the plant other than the recent output. It can't be designed to take advantage of known properties of the plant.

Multiple SMO systems can be designed with various tunable parameters by incorporating a series of SMOs working in concert. The system of SMOs can take advantage of some knowledge about the plant, by tuning the parameters of each layer differently. For example, the front end observer can be designed to reject the chatter and noise problems common to SMOs, while the later systems can be tuned for a higher set of accuracy. This produces better results than a single SMO [19].

SMOs examine a temporal slice of the plant's behavior. This reduction of state space reduces the difficulty of working with nonlinearities. This property has lead to work in the field of Nonlinear Sliding Mode Observers (N-SMO.) If faults are treated as unknown inputs that consist of additive signals, the N-SMO can be designed to separate the system data from the faulty signals. Work in this field has been limited to nonlinear

systems that satisfy the Lipshitz condition. Multiple N-SMOs can be cascaded in series to bring about finer tuned information, while at the same time reducing sensitivity to chatter [26].

The SMO technique allows nonlinear systems to be estimated with linear schemes. These same properties mean that SMOs are capable of handling slowly time-varying signals. This is because of the short term linearization inherent to an SMO. Knowing the speed of the changes in the plant and the speed of the faults becomes very important when dealing with a time-varying nonlinear signal. When dealing with a time-varying system, tuning information needs to be gathered from the plant. The SMO must be designed so that it can tell the difference between natural changes to the properties of the plant and faults. A SMO operating on such a loosely defined system is effective at isolating full failures despite nonlinearities and time-varying details. Intermittent failures and slower slew failures are difficult to isolate, instead being misinterpreted as changes to the plant dynamics [26].

Due to their focus on short term information, SMOs are vulnerable to noise. Alternatively, Kalman filters are designed to optimally tolerate Gaussian noise. Systems utilize Kalman techniques to build observers that either isolate faults and report them, or bring about sufficient robustness to ensure that they can be ignored. Many plants are not subject to Gaussian noise and do not respond as favorably to Kalman based FTC methods. By using the Probability Distribution Function (PDF) of the output sensors, LMI methods can be combined with Kalman techniques. Once the faults have been detected, they can be isolated and the fault-free outputs can be extracted [1].

Time-varying systems that have periods of minimal change followed by periods of high change are not well handled by a Classic Kalman Filter (CKF). Many FTC systems designed with CKFs have to be built so that they consider the broadest changes to the plant as non-faulty behavior. This means that in times of minimal changes, faults can be overlooked. An Adaptive Fading Kalman Filter (AFKF) is created by adding a fading parameter to the filter that is changed dynamically. When the fade time is set high, the filter adapts to changes in the system parameters quickly. As such, large changes in the model and output are not classified as faults. After the system has settled into a new operations mode, the fader can be dropped down. When the fader is low, the AFKF system becomes more sensitive to changes and regains precision with respect to errors. An AFKF is effective when a system can undergo large step changes in the behavior it exhibits. An example of such a system is a high performance car, where changing gears changes the behavior of the system [25], [33].

An alternative to Kalman filters is Principal Component Analysis (PCA). PCA identifies faults by looking at both the cross correlation between the sensors and the autocorrelation within each sensor individually. When using multiple PCA designs simultaneously, Multi-Scale Principal Component Analysis (MSPCA) allows the system to both identify noise and faults by examining different scales of PCA at the same time in different subsystems. When a fault is detected within the same threshold as noise, it can be rejected as misclassified noise. This is effective when dealing with noisy systems. By examining the autocorrelation within a sensor, even systems that lack sufficient redundancy for other methods can be examined with PCA methods [24], [31].

Another way to use PCA in fault detection is to focus on the Partial Least Squares (PLS) and the Squared Prediction Error (SPE). The SPE focuses on the residual space from the PCA calculations. A large residual in the SPE suggests that the output is not matching the expected results. This indicates that the component that is being observed with PCA is exhibiting faulty behavior. Part of the difficulty in this method lies in coming up with an appropriate threshold of fault detection for the SPE. If it is too low, it will mistake changes in plant behavior as faults. But it must be sensitive enough to identify when a real fault occurs. One of the advantages of the SPE/PLS numeric is that it can give a quantitative value for the reasonableness of a fault. If the SPE/PLS is near unity, the system is unlikely to be experiencing a fault, instead experiencing merely high noise. If the SPE/PLS threshold is significantly greater than one, then it is likely that a fault is occurring [7].

Another way to use PCA based techniques is to choose wavelets that isolate the fault dynamics. By choosing a high frequency wavelet filter, the noise of the system can be removed. Then by taking a low frequency wavelet filter, the plant's operations can be removed from an observed dynamic. Applying PCA to the resulting data produces a cleaner view for looking at faults. However, careful design procedure must be observed when designing the high and low wavelet filters. A balance must be struck between rejecting false positives and the speed of fault identification [16].

Both SMO and PCA techniques are designed to produce good results with limited knowledge of the model of the system or its faults. In most systems, a reasonable model of the plant can be obtained. Many techniques focus on directly modifying the state-space equations of the model of the plant. By knowing the internal dynamics of a plant,

it becomes easier to design controllers to correct faults and plant behavior. Modern control design often incorporates a state estimator or state observer to calculate the internal states of a plant, when they are not available as outputs.

Augmented State Observers (ASO) are a form of dynamic fault estimation that builds upon state observers. Faults are classified as inputs and internal states that are unknown, but have a known model. An ASO increases the size of the state estimator to include fault states in the model of the system. By using plant and fault models, the ASO generates the estimated states of the plant and its faults. The state estimates are sent to components such as the controller. The fault estimates are sent to the operator or a fault management system. The standard model for such research focuses on actuator faults and plant faults. Not much work has been done in the field of ASOs to tolerant faulty sensors or controllers. ASOs work best when models of the possible fault types are available [17].

ASO design can be modified to operate when fault models aren't available. The Extended State Observer (ESO) takes the fundamentals of an ASO but reduces all the modeling to a single value. This value determines how sensitive the system is to errors, and how quickly it classifies them as faults. The ESO estimates all the states of the plant as per a normal state observer. Any discrepancy between the estimate and the output is classified as an error. When the aggregate error is over some threshold, it is treated as a fault. ESOs cannot isolate a fault unless the design is limited to only tolerating sensor faults. The advantage of an ESO is that it requires minimal knowledge of how faults will appear in the system, but it is limited in its ability to compensate for them. If the tuning parameter is set too high, model errors and noise will be improperly classified as faults

and give the ESO a chatter problem. If the tuning parameter is set too low, the system's response to a fault may be too slow to prevent a loss of performance and stability [34].

Another classification of ASO is the Unknown Input Observer (UIO). UIOs are designed around classifying faults as uncontrollable input signals to the plant. UIO designs handle additive faults effectively. UIOs can both reject noise and detect disturbances indicative of faults. This is due to their high number of tunable parameters. These tuning parameters require models of the plant and potential faults. Like ASOs, they output the localized fault signals and the estimates of the plant. Unfortunately, the increased number of tuning parameters makes designing an UIO more difficult [37].

The design of FTCs encompasses a large set of methods to satisfy a fault tolerance need. For better results, fault observation and state observation must be performed simultaneously by the same system. Combining this with the desire for noise rejection produces a series of design requirements that are difficult to guarantee in a single observer. Thorough testing of the observer needs to be done before implementation to show that the design is able to handle all the performance requirements. H_2 and $H\infty$ performance indexes can be used to introduce additional robustness, without sacrificing fault localization [40].

All of these methods are effective in some aspects of FTC. Modern performance needs increase the availability of system and fault models. With these models, computing techniques can be used to examine systems. Computer power and capability has made many FTC designs viable. When combined with a skilled designer, performance needs can be met in the modern age despite the presence of faults, disturbances, errors, and noise.

This research uses a series of reduced order observers that take advantage of sensor redundancy to remove sensor faults. The estimators are first designed as Kalman estimators to give them resistance to noise. Each Kalman estimator is designed to anticipate a specific set of sensor faults. A supervisor is designed to select the optimal estimator from the multiple ones available. This supervisor also calculates estimates of the sensor faults. Then the Kalman estimators are upgraded to ASOs. Changing the observers to ASOs adds the ability to estimate and tolerate actuator faults, while retaining the Kalman estimator's resistance to noise. With a few modifications, the controller is able to adapt to the changes in the plant dynamics caused by the actuator faults. One of the strengths of this research is that it incorporates multiple FTC design techniques so that it can tolerate a number of full and partial sensor faults, in addition to actuator errors and noise.

1.4: Research Outline

Chapter 2 provides details of this innovative method for using reduced order observers to create a fault tolerant system. Section 2.1 develops the technique to measure the available sensor redundancy. This is achieved by examining how sensor faults can damage observability. The rank of redundancy is proven sufficient such that sensors can be reconfigured in the presence of faults. Section 2.2 goes into the design process of a bank of reduced order observers. Design begins with a full order Kalman observer. Once the model and the estimator have been defined, a series of reduced order observers are designed. They are laid out to correspond to sets of sensor failures that are tolerable according to the technique developed in section 2.1. Each observer estimates the outputs of the system. If a model of the sensor faults is available, section 2.3 details how to perform fault estimation. Details for how to handle both additive and multiplicative faults are shown. In section 2.4, the supervisor system is designed. The supervisor determines which of the observers is the most accurate one and which set of sensors are currently faulty. In chapter 2, the fault tolerant system does not use a controller and is operating as an observer in the open-loop configuration.

In chapter 3, two different techniques to tolerate actuator faults are explored. Section 3.1 develops a technique to measure the available actuator redundancy. This is done by examining how actuator faults can impair controllability. The redundancy can be used to determine when system reconfiguration can be used to bypass faulty actuator components without losing controllability. As actuator redundancy is uncommon, an alternative method to tolerate actuator faults is explored. An Augmented State Observer (ASO) is used to tolerate actuator faults and adapt the system to tolerate the changes in

the plant dynamics. The ASO technique uses an adaptive controller to modify the system's response to the actuator faults.

In chapter 4, the reduced order observer design of chapter 2 is extended to tolerate actuator and sensor faults. A controller is added to enable feedback control of the system. The observer is changed to an ASO, and the changes that must be made so that the reduced order observers can adapt to actuator faults as well as sensor faults are examined in section 4.1. The supervisor is detailed in section 4.2. Proofs are included to show that fault tolerance is maintained in the presence of simultaneous sensor and actuator faults. The components of the fault tolerant system are assembled as shown in Figure 1.4.1.

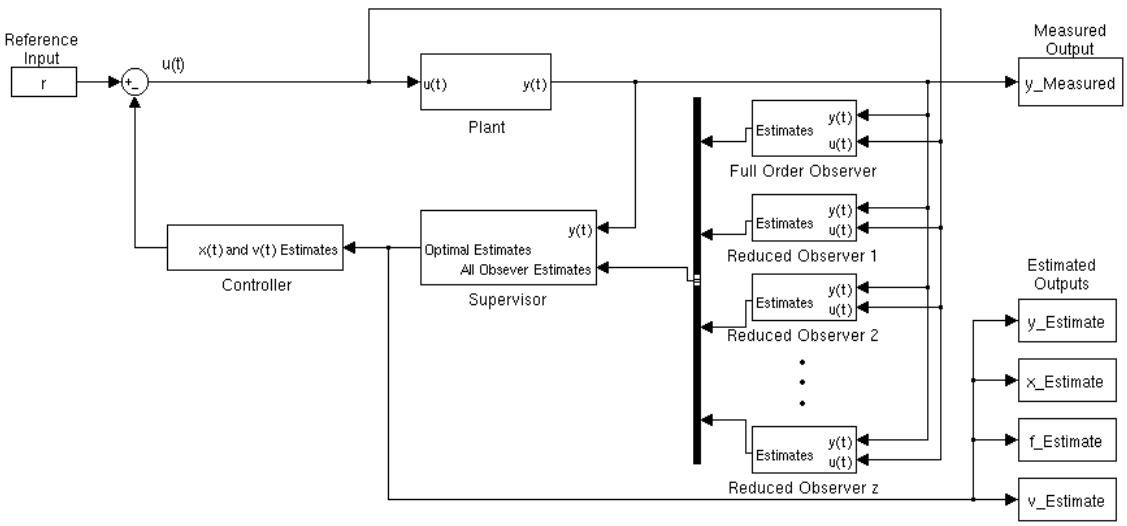

Figure 1.4.1: Overview of the system, observers, supervisor, and controller.

Chapter 5 examines a third order system that illustrates the application of this research with a Kalman estimator as developed in chapter 2. In this chapter, only sensor faults are examined. The plant used in the example is a turbofan engine. The operation

of the engine and the fault tolerant system are simulated with Matlab. Section 5.1 verifies that the turbofan engine exhibits sufficient redundancy for the purposes of this work. The Kalman observers are designed in section 5.2. The supervisor is determined in section 5.3. Fault estimation is detailed, and the whole system is assembled to view the results.

In section 6.1, the actuator redundancy of the turbofan engine is calculated. In sections 6.2-6.4, the plant under examination is changed from a turbofan engine to a Boeing 747. In section 6.2, the airplane's dynamics are examined and an adaptive controller is designed. Section 6.3 details the design of all the ASOs from section 4.1 as well as the supervisor from section 4.2. Then various simulations of the fully assembled fault tolerant system are reviewed in section 6.4. Both actuator and sensor faults are simulated and explored.

Chapter 7 analyzes the results obtained in the previous chapters. Conclusions regarding efficiency, adaptability, and ease of use are listed in section 7.1. Details regarding the strengths and weaknesses of this technique are also included. Design applicability, potential limitations, and short comings of the method are outlined. Potential avenues for research in the future involving these methods are explored in section 7.2.

.

1.5: Contribution of this Research

The research presented in this dissertation is intended to aid in the design of Fault Tolerant Control systems. This work contributes a method for tolerating sensor failures of multiple types. It focuses on taking advantage of redundancy in the sensor information to reconstruct fault-free state and output information and potentially estimate the sensor faults themselves. This research also shows how to apply these novel techniques with an Augmented State Observer in order to incorporate fault tolerance for actuator failures.

A method to measure available sensor redundancy is shown and proven. This gives designers a new tool in the examination of FTC systems. This measure can be examined for additional details that provide insight into the areas of the plant that are more vulnerable to faults. A technique to measure the available actuator redundancy is also shown. This measure can also provide insight into the vulnerable actuators in a system. This measure's application is limited as actuator redundancy is less common in most systems.

A system of designing a series of reduced order observers is presented. This system is proven to eliminate specific sets of sensor faults. The speed and efficiency of the fault tolerance of the individual observers is shown. This system of reduced order observers is first derived from a Kalman Estimator, then an Augmented State Observer. By incorporating Augmented State Observers, the system can tolerate both actuator and sensor faults. One limitation of these reduced order observers is that large fault sets will mandate a large number of observers. Limited computational power can make real time analysis difficult in these types of systems.

Lastly, a supervisor system is designed to determine which sensor faults are occurring and which observers are producing the optimal results. This research shows how to calculate sensor faults if their model is available. All of these components are combined to produce a fault tolerant system that is easier to design than similar methods. This fault tolerant system is able to quickly and efficiently eliminate sensor and actuator faults as well as produce accurate state estimates which can be used in state feedback.

CHAPTER 2: SENSOR FAULT TOLERANCE METHOD

2.1: Measuring Sensor Redundancy

The first part of this work elaborates on a method to calculate sensor redundancy. This initial step outlines sufficient conditions that determine when a sensor can fail and its signal can still be recovered. In this step, sensor failure is taken to the extreme case of a complete loss of information, or a full fault. These worst case assumptions guarantee that the calculations result in a sufficient condition for redundancy. Observability of the system is measured before and after a set of sensors fail. Systems that maintain observability despite the loss of sensor information have sufficient redundancy for the implementation of the subsequent steps of the design process. Additional information about the sensors and redundancy is obtained by analyzing the results of this crucial first step.

This work takes a standard method for calculating observability as discussed in [18] and expounds upon it. The state equations are modified to represent the effects of faulty sensors. The faulty sensors are represented by modifying the corresponding elements of the state matrix to zero. The changes to observability are then computed. A system to test and organize all possible fault sets is outlined in this dissertation. Based on those results, the measure of available redundancy is defined.

Case: Time-variant system with a single sensor failure

Assume a linear time-varying system defined as follows.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \tag{2.1.1}$$

$$y(t) = C(t)x(t) + D(t)u(t) \tag{2.1.2}$$

This pair of equations represents the plant under normal conditions. The matrices and variables are defined as follows: the state vector is $x(t) \in R^n$, $u(t) \in R^m$ is the control vector, and the output vector is $y(t) \in R^p$. The matrices $A(t)$, $B(t)$, $C(t)$, and $D(t)$ are known and of dimensions n x n, n x m, p x n and p x m respectively. Time indexes t_0 and t_f represent the initial and final time respectively. Assume the system is observable on $(A(t), C(t))$ for all $t \in [t_0, t_f]$. $W(t_0, t_f)$ and $M(t_0, t_f)$ are n x n controllability and observability Grammians defined below. $\Phi(t_1, t_2)$ is the state transition matrix between times t_1 and t_2.

$$W(t_0, t_f) = \int_{t_0}^{t_f} \Phi(t_o, t)B(t)B^T(t)\Phi^T(t_o, t)\, dt \tag{2.1.3}$$

$$M(t_0, t_f) = \int_{t_0}^{t_f} \Phi^T(t, t_o)C^T(t)C(t)\Phi(t, t_0)\, dt \tag{2.1.4}$$

In [18] a proof is presented that shows that the state equations (2.1.1) and (2.1.2) are observable for $t \in [t_0, t_f]$ if and only if $M(t_0, t_f)$ is invertible. The proof of this will not be repeated here.

Now that the original plant has been defined, it is modified to represent a failure in one of the sensors. Define $C_z(t)$ as the $C(t)$ matrix, with the z^{th} row set to zero, $z \in [1, p]$ and redefine part of the state equation as follows.

$$y(t) = C_z(t)x(t) \tag{2.1.5}$$

This change to the state equations is mathematically equivalent to the z^{th} element of $y(t)$ no longer being defined by equation (2.1.2). The plant with the sensor failure is defined by (2.1.1) and (2.1.5). Part of the proof for observability will require the definition of an L matrix. Define the p x n matrix-function L by induction, presuming the existence and continuity of the indicated derivatives.

$$L_0(t) = C_z(t)$$

$$L_j(t) = L_{j-1}(t)A(t) + \dot{L}_{j-1}(t), j = 1, 2, \ldots$$

The L matrix has the property that derivatives of the state transition matrix Φ are equivalent to multiplications by L. This property is defined mathematically below. For all t and σ and non-negative j

$$\frac{d^j}{d\sigma^j}\left[C_z(\sigma)\Phi(t,\sigma)\right] = L_j(\sigma)\,\Phi(t,\sigma), j = 0,1,$$

An intuitive proof is laid out with induction. In the $j = 0$ case

$$\frac{d^0}{d\sigma^0}\left[C_z(\sigma)\Phi(t,\sigma)\right] = L_0(\sigma)\Phi(t,\sigma)$$

$$C_z(\sigma)\Phi(t,\sigma) = C_z(\sigma)\Phi(t,\sigma)$$

And in the inductive case

$$\frac{d^{j+1}}{d\sigma^{j+1}}\left[C_z(\sigma)\Phi(t,\sigma)\right] = \frac{d}{d\sigma}\left[L_j(\sigma)\Phi(t,\sigma)\right]$$

$$= L_j(\sigma)A(\sigma)\Phi(t,\sigma) + \left(\frac{d}{d\sigma}\left[L_j(\sigma)\right]\right)\Phi(t,\sigma)$$

$$\frac{d^{j+1}}{d\sigma^{j+1}}\left[C_z(\sigma)\Phi(t,\sigma)\right] = L_{j+1}(\sigma)\Phi(t,\sigma) \qquad \text{Q.E.D.}$$

The application of this L matrix is straightforward and directly useful. Suppose q is a positive integer such that for all $t \in [t_0, t_f]$, $C_z(t)$ is q times continuously differentiable and $A(t)$ is $(q-1)$ times continuously differentiable on $[t_0, t_f]$ for some $t_c \in [t_0, t_f]$. The following test determines if the system is still observable, despite the failure of the z^{th} component of $y(t)$.

$$rank\begin{bmatrix} L_0(t_c) \\ L_1(t_c) \\ \vdots \\ L_q(t_c) \end{bmatrix} = 0 \qquad (2.1.6)$$

This is proven by contradiction. Suppose that $t_c \in [t_0, t_f]$ and satisfies (2.1.6). Setting up a contradiction, presume that $M(t_0, t_f)$ is not invertible. As such, there exists a nonzero n x 1 vector x_a that satisfies the following.

$$0 = x_a^T\, M(t_0, t_f) x_a$$

$$= \int_{t_0}^{t_f} x_a^T \Phi^T(t, t_o) C_z^T(t) C_z(t) \Phi(t, t_0) x_a \, dt$$

$$= \int_{t_0}^{t_f} \| C_z(t) \Phi(t_o, t) x_a \|^2 \, dt$$

$$C_z(t) \Phi(t_o, t) x_a = 0, t \in [t_0, t_f] \tag{2.1.7}$$

Let x_b be the nonzero vector $x_b = \Phi^T(t_o, t_c) x_a$ and substitute it into (2.1.7).

$$C_z(t) \Phi(t_c, t) x_b = 0, t \in [t_0, t_f] \tag{2.1.8}$$

With (2.1.8) defined, all cases must be shown to converge to zero. In the case of $j = 0$, choosing $t = t_c$ gives us that $L_0 \Phi(t_c, t_c) x_b = L_0 x_b = 0$. In the case of $j = 1$, differentiating (2.1.8) with respect to t gives $L_1 \Phi(t_c, t) x_b = 0, t \in [t_0, t_f]$ and specifying $t = t_c$, gives us that $L_1 x_b = 0$. From this we can derive the formula for the general case.

$$\frac{d^j}{dt^j} [L_1 \Phi(t_c, t) x_b]_{t=t_c} = L_j(t_c) x_b = 0, j = 1,2, \dots q$$

All these cases are compiled together for $j = 0,1,2, \dots q$.

$$\begin{bmatrix} L_0(t_c) \\ L_1(t_c) \\ \vdots \\ L_q(t_c) \end{bmatrix} x_b = 0$$

This contradicts the linear independence implied by (2.1.6). Thus (2.1.6) is a sufficient condition for the state equation to be observable on $[t_0, t_f]$. So long as this condition is maintained, the system maintains observability despite the loss of the z^{th} sensor.

Case: Time-invariant system with a single sensor failure

Complications due to (2.1.1) and (2.1.2) being time-variant prevent this from being a necessary condition. By reducing the set of state equations to time-invariant systems, this can be extended to a necessary and sufficient condition. In this case, the state equations are redefined to the definitions given in (2.1.9) and (2.1.10). Based on that, the *L* matrix rank test described in (2.1.6) is simplified to (2.1.11).

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{2.1.9}$$

$$y(t) = C_z x(t) \tag{2.1.10}$$

$$rank\begin{bmatrix} C_z \\ C_z A \\ \vdots \\ C_z A^{n-1} \end{bmatrix} = n \tag{2.1.11}$$

To prove the necessary condition, we presume that the observability Grammian $M(t_0, t_f)$ is not invertible. Therefore, there exists a nonzero n x 1 vector x_a such that $x_a^T M(t_0, t_f) x_a = 0$. In the time-invariant case $\Phi(t_o, t) = e^{A(t_o - t)}$.

$$\int_{t_0}^{t_f} x_a^T e^{A^T(t_o - t)} {C_z}^T(t) C_z(t) e^{A(t_o - t)} x_a \, dt = 0$$

$$\int_{t_0}^{t_f} \left\| C_z(t) e^{A(t_o - t)} x_a \right\|^2 dt = 0$$

$$C_z(t) e^{A(t_o - t)} x_a = 0, t \in [t_0, t_f] \quad (2.1.12)$$

The form of (2.1.12) is similar to (2.1.8) and we perform the same differentiations as before. Due to the time-invariant nature, the case structure is simpler this time. Equation (2.1.12) is differentiated j times, and at each derivative t is set to t_0 so that it cancels out. This expansion gives the general form $C_z A^j x_a = 0, j = 0,1 \ldots n-1$. All of those equations are aligned into a column.

$$\begin{bmatrix} C_z \\ C_z A \\ \vdots \\ C_z A^{n-1} \end{bmatrix} x_a = 0 \quad (2.1.13)$$

This equation contradicts (2.1.11). Q.E.D. This proof is a necessary condition for maintaining observability. The proof for the sufficient condition for the time-variant case also applies to the time-invariant case described in (2.1.9) and (2.1.10). By combining these proofs together, the condition becomes a necessary and sufficient requirement for maintaining observability in the time-invariant case.

Case: Rank 1 sensor redundancy

These proofs only examine a single test of a system undergoing one sensor fault. To be useful, this test must be applied to the entirety of the plant. As such, a vector is constructed with the faulty sensor z as a variable. Define $R_O(z)$ as an n x 1 vector.

$$R_O(z) := rank\begin{bmatrix} L_0(t_c) \\ L_1(t_c) \\ \vdots \\ L_q(t_c) \end{bmatrix}, z \in [1, p] \tag{2.1.14}$$

This vector can be analyzed to gather information regarding how the plant responds to sensor failure. Equation (2.1.14) provides a test that determines if there is sufficient redundancy in the sensors of a plant for any single sensor to fail without impacting observability. If the minimum value of (2.1.14) is n for all $z \in [1, m]$, then the system is observable in the presence of a single fault, regardless of which sensor is faulty. This kind of plant has rank 1 sensor redundancy. This forms the basis in the design of the reduced order observers required in the later steps of this research. It should be noted that if the R_O vector is being applied to a time-invariant system, this condition is both necessary and sufficient.

In a time-variant system, if $R_O(z) < n$, removal of the z^{th} component of $y(t)$ may cause a loss of observability. In a time-invariant system, removal of the z^{th} component will cause a loss of observability. This is because the test is only proven necessary in the time-invariant case. In general, failing this test means there isn't enough redundancy to proceed with this technique. By examining which sensor failure leads to the system failure, additional redundant sensors can be added to the original plant.

Case: Rank r sensor redundancy

Equation (2.1.14) deals with a single sensor fault of undetermined characteristics. Modern systems fault tolerance needs often require that they are able to handle multiple sensor faults. From this need, (2.1.14) is extended to handle multiple sensor failures. A series of new definitions need to be laid out to achieve this.

Define y_f as the set of all sensor elements $\{1, 2, \ldots, p\}$. Define r as an integer from$[1, p)$. This r represents the number of simultaneous sensor faults that are to be tested for redundancy. Define $zz(i)$ as the i^{th} combination of set y_f, for $i \in [1, \frac{p!}{r!(p-r)!}]$. The full set of combinations is defined as

$$zz(i) = y_f \binom{r}{p}, r \in [1, p)$$

Further define $C_{zz}(t)$ as the $C(t)$ matrix with all columns corresponding to the i^{th} set of sensors defined by $zz(i)$ redefined as zero and adjust the state matrix accordingly.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \tag{2.1.15}$$

$$y(t) = C_{zz}(t)x(t) \tag{2.1.16}$$

Again, we will presume an L matrix as follows, subject to the existence and continuity of all indicated derivatives. A complete proof of the definition of this version of L is not derived here as the proof is functionally equivalent to the ones derived earlier in this section.

$$L(t) = C(t)_{zz}(t)$$

$$L_j(t) = A(t)L_{j-1}(t) + \dot{L}_{j-1}(t), j = 1, 2, \ldots$$

The L matrix is redefined for each different $C_{zz}(t)$, in the same way as before. In this case, each L corresponds to each set of $zz(i)$ for all possible i for a given r. Define $RR_O(i)$ as the i x 1 vector below.

$$RR_O(i) := rank\begin{bmatrix} L_0(t_c) \\ L_1(t_c) \\ \vdots \\ L_q(t_c) \end{bmatrix}, i \in [1, \frac{n!}{r!(n-r)!}] \tag{2.1.17}$$

The measure $RR_O(i)$ can be analyzed in the same way as the $R_O(z)$ vector. If (2.1.17) passes the rank test for all i, the plant is sufficient to have sensor redundancy of rank r. To be more explicit, if the minimum of $RR_O(i)$ is n for all i, the system is observable in the presence of up to r sensor faults. This property does not require any knowledge about the nature of the faults. As before, if (2.1.15) and (2.1.16) define a time-invariant system, the test in (2.1.17) becomes both necessary and sufficient.

If the minimum of $RR_O(i)$ is not n, there is insufficient redundancy to guarantee sensor reconstruction is possible. In a time-invariant system, sensor reconstruction is not possible. For any $RR_O(i)$ that does not equal n, $zz(i)$ is the set of faults that can cause the system to lose observability. As before, designers can consider adding additional sensors to the plant to improve the redundancy. One option is to reduce the value of r and recalculate the redundancy for a smaller set of faults. Another alternative is to design a subsystem for those specific fault cases. Usually this kind of subsystem is built to perform a safe shutdown.

As has been seen, the preceding work defines a series of vectors and matrices that measure sensor redundancy. By calculating the R_O or RR_O measures, the level of

available redundancy is determined. Once this has been completed and sufficient redundancy confirmed, the design process of this research proceeds to the next step. The next step is to design a bank of reduced order observers that correspond to each possible fault set that is tolerable.

2.2: Reduced Order Observer Design

The second step in the design process demonstrates the design of a set of observers that can reconstruct all possible sensor failures. This bank of observers consists of a single full order observer and a reduced order observer for each tolerable fault set. First, a full order output estimator for the system is constructed. This estimator will be used by the system when the plant is not undergoing faults. In this chapter, a modified Kalman observer is chosen as the full order observer. This method is applicable to other estimation techniques with simple adjustments. One of the strengths of this method is that a model for sensor failure is not needed at this step in the design process.

Once the observer design template has been selected, a reduced order observer is designed for each set of sensor failures. For example, in the case where only one sensor can fail at a time, each reduced order observer is built to compensate for a single sensor fault. This step creates an observer that estimates the system's sensors without using those sensors it presumes are faulty. This means that each reduced observer performs its estimation based only on what it presumes is fault free sensors. The previous section's rank of redundancy guarantees that there is sufficient sensor redundancy in the plant to perform this reduction and still reconstruct all the sensors. This process is repeated for all tolerable sets of sensor failures that the system is designed to withstand. This step is outlined in such a way as to speed up the design process. All of the observers use the same basic design structure and mathematics.

This method can be applied to both time-invariant and time-variant systems. It is possible that certain classifications of nonlinearities can work with this method, but that is beyond the scope of this research. There are a few requirements that must be met for

this step of the design to work. The plant must be observable. Additionally, the sensor redundancy calculated in section 2.1 must be sufficient to handle the number of simultaneous faults that the system is meant to tolerate. With that level of redundancy, any appropriate set of sensor failures will not affect the observability of the reduced order observers. In this section, faults are limited to sensor faults. Chapter 3 shows how to enhance this work in order to tolerate actuator faults. This method can handle a wide range of sensor fault types. This method can compensate for drift errors, intermittent faults, noise, and full failures. Preliminary work suggests that errors of a nonlinear nature are also corrected, but that is not proven in this research.

Because this method does not correct for model errors, an accurate model of the system must be available. A model of the sensor failures is not needed in this section, but a general model will be needed in section 2.3 if fault estimation is to be done. Even without a fault model, this method requires that it must have the knowledge of the maximum simultaneous system faults it must tolerate. Work shows that so long as this maximum is within the sensor redundancy calculated in section 2.1, fault free data can always be reconstructed by an appropriate observer.

Another strength of this method is that its design is quite simple compared to more complicated methods. Optimizing the design is quick, because there is only one tuning matrix that is consistent over all the observers. Lastly, the exact type of estimator being used can vary with design needs. This gives the designer a great degree of freedom in the application of this process, while being extremely effective at isolating sensor failures.

To show the validity of this method, a series of proofs is given below. Assume a linear time-variant system as follows defines the plant. In this plant, faults are presumed to be an unknown additive function.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \quad (2.2.1)$$

$$y(t) = C(t)x(t) + D(t)u(t) + F(t)f(t) \quad (2.2.2)$$

Define the variables as follows: $x(t) \in R^n$ is the state vector, the control vector is $u(t) \in R^m$, $y(t) \in R^p$ is the output vector, and $f(t) \in R^p$ is the fault vector. Time t_e represents the time index when the fault begins. Before time t_e, $f(t) = 0$. After time t_e, $f(t)$ becomes an unknown, but bounded vector. The matrices $A(t)$, $B(t)$, $C(t)$, $D(t)$, and $F(t)$ are known and have dimensions n x n, n x m, p x n, p x m, and p x p respectively. Time indexes t_i and t_f represent the initial and final time respectively. Assume the system is observable on $(A(t), C(t))$ for all $t \in [t_i, t_f]$. The $F(t)$ matrix represents how the additive faults interact with the various sensors. If a model is not available, the $F(t)$ matrix can be defined as the identity matrix without a loss of generality.

Theorem 2.2.1

A reduced order observer can be designed to remove the faulty information from a specific sensor, assuming all other sensors are operating without fault and the system has rank 1 sensor redundancy.

Proof 2.2.1

The subscript i will be used to reference the i^{th} row in a matrix, or element in a vector. Presume that a fault has occurred in the i^{th} sensor, $y_i(t)$. No other faults are occurring. This means that $f(t) = 0$ for all elements other than $f_i(t)$.

The subscript $\bar{\imath}$ will be used to reference a matrix that has the i^{th} row removed, or a vector that has the i^{th} element removed. Define the reduced order output vector $y_{\bar{\imath}}(t)$ to be the reduced set of $y(t)$ outputs, that does not include the faulty i^{th} sensor as follows

$$y_{\bar{\imath}(t)} = C_{\bar{\imath}}(t)x(t) + D_{\bar{\imath}}(t)u(t) + F_{\bar{\imath}}(t)f_i(t) \tag{2.2.3}$$

$F_{\bar{\imath}}(t)$ has no elements on the i^{th} row and $f(t)$ is zero in all rows other than the i^{th} row.

$$y_{\bar{\imath}(t)} = C_{\bar{\imath}}(t)x(t) + D_{\bar{\imath}}(t)u(t) \tag{2.2.4}$$

The faulty signal has been removed completely from the reduced system. As the rank of redundancy ensures that the system defined by (2.2.1) and (2.2.4) is still observable on $(A, C_{\bar{\imath}})$, a reduced observer can be designed to estimate $x(t)$ with $\hat{x}(t)$. From this estimate, the full set of sensor outputs can be estimated.

$$\hat{y}(t) = C(t)\hat{x}(t) + D(t)u(t) \tag{2.2.5}$$

The specific estimate of interest is $\hat{y}_i(t)$. This is a fault free estimate of $y_i(t)$, as the faulty sensor information was not used in the fault estimate.

Remark 2.2.1

It can be seen that when there is no fault in the i^{th} component, (2.2.3) will converge to the fault free output estimates in (2.2.5). This property will hold true for all of the reduced order observers. Therefore, when all reduced order observers report that there is no fault, the full order observer produces the most accurate estimates because it has all possible fault free information. It will also converge faster than the other observers due to the same property.

The next section will use the output estimates produced by the observers and the measured outputs from the plant to produce sensor fault estimates. These fault estimates rely on understanding how the faults interact with the system. If this information is unavailable, assumptions about the fault model can be made so that fault estimation is possible.

2.3: Sensor Fault Estimation

It is imperative that the reduced order observer is designed with sufficient speed such that $\hat{x}(t)$ converges to $x(t)$ faster than the system changes. When properly designed, the fault free $\hat{y}_i(t)$ estimate can be used to estimate the fault signal $f(t)$. Equation (2.2.5) is substituted into (2.2.2) and it is assumed that the estimate $\hat{x}(t)$ converges to $x(t)$with sufficient speed.

$$y(t) = \hat{y}(t) + F(t)f(t)$$

The i^{th} row of this equation is analyzed, as each estimator assumes faults only occur on the i^{th} row.

$$y_i(t) = \hat{y}_i(t) + F_i(t)f_i(t)$$
$$y_i(t) = \hat{y}_i(t) + F_i(t)f_i(t)$$
$$\hat{f}_i(t) = F_i^{-1}(t)(y_i(t) - \hat{y}_i(t)) \qquad (2.3.1)$$

If the inverse of the fault matrix $F^{-1}(t)$ exists, an estimate of $f(t)$ can be calculated based on the error between the sensor estimate and the faulty sensor. On rows other than i, $f(t)$ and $\hat{f}(t) = 0$.

Remark 2.3.1

When a fault occurs, the full order observer can detect by comparison that it is not able to track the output of the system. To identify which component is faulty, each reduced order observer is checked by (2.3.1) to see if $\hat{y}_{\bar{i}}(t) - y_{\bar{i}}(t)$ is less than some

error threshold f_t which is determined experimentally. Only the reduced order observer that does not include the faulty sensor i will converge to zero error on the non-faulty signals. Once the optimal reduced order observer is selected, its sensor estimates and expected faults are used in the fault estimation process.

Multiplicative Faults

Up to now, the model has presumed that the faults are additive in nature. This is not always the way the fault dynamics operate. While additive models can be used to examine slew errors, they are often multiplicative in nature. This is because many types of fatigue can cause a sensor to uniformly degrade in performance. In that case, a multiplicative model of the sensor faults is more useful. To accommodate this, the state space output equation is changed.

$$y(t) = F(t)C(t)x(t) + D(t)u(t) \tag{2.3.2}$$

In this case, $F(t)$ is redefined so that the faults are defined as a diagonal matrix consisting of the elements of $f(t)$. The fault vector $f(t)$ is 1 for non faulty components and a number generally between 0 and 1 for faulty components.

$$F(t) = \begin{bmatrix} f_1(t) & 0 & \cdots & 0 \\ 0 & f_2(t) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & f_p(t) \end{bmatrix}$$

It was assumed that no more than k faults occur. Or put another way, $f(t) \neq 1$ for no more than k elements. The change to the fault model does not alter the

convergence of the reduced order observers, or the number of faults they can tolerate. So the sensor faults can be estimated by comparing the estimation of the system outputs with the measured sensor outputs. Equation (2.2.5) is subtracted from (2.3.2) and it is assumed that the estimate $\hat{x}(t)$ converges to $x(t)$ with sufficient speed.

$$y(t) - \hat{y}(t) = F(t)C(t)x(t) - C(t)\hat{x}(t)$$
$$y(t) - \hat{y}(t) = (F(t) - I)C(t)\hat{x}(t) \tag{2.3.3}$$

As both $F(t)$ and I only exist on the diagonal, they can be replaced with a vector representing the diagonal, and the multiplication replaced with a dot product. The vector $\underline{1}$ is a p x 1 vector of appropriate length where all the elements are 1. The diagonal of $F(t)$ is $f(t)$. As such, (2.3.3) is further derived as follows.

$$y(t) - \hat{y}(t) = \big(f(t) - \underline{1}\big).* C(t)\hat{x}(t) \tag{2.3.4}$$

This series of vectors can be subdivided into p equations, one for each of the p sensors. The subscript j represents the j^{th} element of a vector, or the j^{th} row of C, for $j \in [1, p]$.

$$y_j(t) - \hat{y}_j(t) = \big(f_j(t) - 1\big)C_j(t)\hat{x}(t)$$
$$\frac{y_j(t) - \hat{y}_j(t)}{C_j(t)\hat{x}(t)} = \big(f_j(t) - 1\big)$$
$$\hat{f}_j(t) = \frac{y_j(t) - \hat{y}_j(t)}{C_j(t)\hat{x}(t)} + 1, j \in [1, p] \tag{2.3.5}$$

The $\hat{f}(t)$ vector can be calculated as shown in (2.3.5) to produce an estimate of $f(t)$. This allows the system to estimate multiplicative sensor faults.

2.4: Supervisor Decision Process

A series of reduced order observers have been designed so that they remove sensor data presumed to be faulty from their outputs. The observers are not able to localize sensor faults by themselves. A supervisor system is built that selects the reduced observer that best estimates the sensor fault set. Presuming that there are no more than k sensor faults, there must exist at least one observer that correctly identifies and isolates the faulty sensors.

Each reduced order observer presumes that a specific set of sensors are faulty and the others are fault free. If one of the sensors that was presumed fault free is not, the observer will be unable to compensate for the sensor fault. This will cause the sensor estimates of that observer to deviate from the actual outputs of the plant. Errors on components that were presumed fault free are called unexpected errors. These unexpected errors are used by the supervisor to detect that the observer has not matched the sensor fault set and is not an optimal estimate. The supervisor selects the observer that has the lowest unexpected sensor error as the optimal estimator for the sensor data. The reduced observers anticipate that specific sensors are faulty and ignore those sensors. The optimal supervisor removes the faulty sensor data from its estimates. Therefore the unexpected error is minimal for the optimal observer. The supervisor propagates the state and output estimates produced from that observer's result. The supervisor then calculates sensor failure estimates as was detailed in section 2.3.

Assume a linear time-variant system of the form below.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \tag{2.4.1}$$

$$y(t) = C(t)x(t) + F(t)f(t) \tag{2.4.2}$$

Where $x(t) \in R^n$ is the state vector, $u(t) \in R^m$ is the control vector, $y(t) \in R^p$ is the output vector, and $v(t) \in R^q$ is the fault signals. Matrices $A(t), B(t)$, and $C(t)$ are known and of dimensions n x n, n x m, and p x n respectively. The matrix $F(t)$ has dimensions p x p. With no loss of generality, q is assumed to be equal to p and $F(t)$ is assumed to be an identity matrix. Time indexes t_i and t_f represent the initial and final time respectively. The sensor fault signal $f(t)$ is non-zero for no more than k elements for any given t. An upper limit for k is given in section 2.1, for a given $A(t)$ and $C(t)$. Presume that a series of reduced order observers have been designed for the system described by (2.4.1) and (2.4.2) by the method detailed in section 2.2. Each observer has been designed to remove one set of up to k output signals from their estimates. As per section 2.2, reduced order observers have been designed for all tolerable fault sets where up to k output signals fail. Within these limits, there is sufficient redundancy for the system to maintain observability on all reduced observers. Based on this, the general form of a reduced observer is described below.

$$\dot{\hat{x}}(t) = A(t)\hat{x}(t) + B(t)u(t) + L_{\bar{\imath}}(t)\left(y_{\bar{\imath}}(t) - C_{\bar{\imath}}(t)\hat{x}(t)\right) \tag{2.4.3}$$

$$\begin{bmatrix}\hat{y}(t)\\ \hat{x}(t)\end{bmatrix} = \begin{bmatrix}C(t)\\ I\end{bmatrix}\hat{x}(t) \tag{2.4.4}$$

$L(t)$ is the matrix defined in 2.2 that determines the convergence speed of the full order observer. The set i represents the set of sensor signals that the reduced order observer presumes are faulty. The subscript $\bar{\imath}$ signifies a matrix or vector that has the

rows or elements that correspond to the fault set i set to zero. Assume that the design of the $L_i(t)$ matrix for the observers is sufficient such that when $f(t) = 0$, as $t \to \infty$, $\hat{x}(t) - x(t) \to 0$ for all tolerable i. Define $e_y(t)$ as the measure of observer error.

$$e_y(t) = |y(t) - \hat{y}(t)| \tag{2.4.5}$$

This error is split into two components. The expected sensor error is called $e_e(t)$. This corresponds to the elements of $y(t)$ that the reduced order observer ignored. The unexpected sensor error is $e_u(t)$, which is the errors in the sensor estimates that were presumed fault free by the reduced observer. The subscript i corresponds to the sensor elements that were presumed faulty. Correspondingly, the subscript $\bar{\imath}$ references to elements that were presumed fault free by the observer.

$$\begin{bmatrix} e_u(t) \\ e_e(t) \end{bmatrix} = \begin{bmatrix} |y_{\bar{\imath}}(t) - \hat{y}_{\bar{\imath}}(t)| \\ |y_i(t) - \hat{y}_i(t)| \end{bmatrix} \tag{2.4.6}$$

As $y_i(t)$ contains faulty information from the real sensor outputs, the observer that minimizes $e_e(t)$ is not the one that is fault free. In the general case, the observer that maximizes the expected error is the correct one, presuming all other observers converge to the faulty data. That presumption of convergence cannot be guaranteed for all cases. Instead, the supervisor system is designed to choose the observer that has the lowest unexpected error $e_u(t)$. The optimal reduced observer will predict the location of the sensor error correctly. As the optimal observer correctly predicts the location of the errors, all faulty signals will be in $e_e(t)$. When the supervisor selects the correct

observer for fault free sensor estimates, the $e_e(t)$ errors can be used to calculate the sensor fault vector $f(t)$, as was shown in section 2.3.

Theorem 2.4.1

If and only if a reduced order observer's unexpected error converges to zero at the input, that observer reconstructs a fault free estimate of all the outputs. In other words, by choosing the reduced order observer with the lowest $e_u(t)$, the supervisor selects the reduced order observer that has the best estimates of the fault set and sensor data. When multiple observers have the same near zero $e_u(t)$ for a given t, the supervisor selects the observer with the fewest removed components. Three cases are shown.

Proof 2.4.1: Case 1

In this case, when $t \in [t_i, t_f]$, $f(t) = 0$. Equations (2.4.3) and (2.4.2) are combined, assuming that $\hat{x}(0) = x(0)$ and $L(t)$ is properly designed so that $\hat{x}(t) \to x(t)$ as $t \to \infty$.

$$\dot{\hat{x}}(t) = A(t)\hat{x}(t) + B(t)u(t) + L_{\bar{i}}(t)\big(y(t) - C(t)\hat{x}(t)\big)$$

$$\dot{\hat{x}}(t) = A(t)\hat{x}(t) + B(t)u(t) + L_{\bar{i}}(t)\big(C(t)x(t) - C(t)\hat{x}(t)\big)$$

$$\dot{\hat{x}}(t) = A(t)\hat{x}(t) + B(t)u(t) \tag{2.4.7}$$

$$\begin{bmatrix}\hat{y}(t)\\ \hat{x}(t)\end{bmatrix} = \begin{bmatrix}C(t)\\ I\end{bmatrix}\hat{x}(t) \tag{2.4.8}$$

Equations (2.4.7) and (2.4.8) have the same form as (2.4.1) and (2.4.2) and the error $e_y(t)$ will be zero for all observers. Thus all observers produce fault free estimates. In this case, the supervisor chooses the full observer because it has the most redundant

information. As $\hat{y}(t) = y(t)$, this is both the sufficient and the necessary case for fault free estimation, albeit the trivial case.

Proof 2.4.1: Case 2

In this case, when $t \in [t_i, t_f]$, $f(t) \neq 0$ for a number of elements equal to k. As k is the maximum possible number of simultaneous faults the system of observers can tolerate, there will only be one observer that will collapse down to the equations defined by (2.4.7) and (2.4.8). The following shows that no other observers will converge without error, and thus are faulty. For the initial conditions, assume $\hat{x}(0) = x(0)$.

$$\dot{\hat{x}}(t) = A(t)\hat{x}(t) + B(t)u(t) + L_{\bar{\imath}}(t)\big(y(t) - C(t)\hat{x}(t)\big)$$

$$\dot{\hat{x}}(t) = A(t)\hat{x}(t) + B(t)u(t) + L_{\bar{\imath}}(t)\big(C(t)x(t) + F_i(t)f_i(t) - C(t)\hat{x}(t)\big)$$

$$\dot{\hat{x}}(t) = A(t)\hat{x}(t) + B(t)u(t) + L_{\bar{\imath}}(t)F_i(t)f_i(t) \tag{2.4.9}$$

$$\begin{bmatrix}\hat{y}(t)\\ \hat{x}(t)\end{bmatrix} = \begin{bmatrix}C(t)\\ I\end{bmatrix}\hat{x}(t) \tag{2.4.10}$$

Equations (2.4.4) and (2.4.10) are substituted into (2.4.5).

$$e_y(t) = |C(t)x(t) + F(t)f(t) - C(t)\hat{x}(t)| \tag{2.4.11}$$

Equations (2.4.3) and (2.4.9) are substituted into (2.4.11).

$$e_y(t) = \begin{vmatrix} C(t)\int_{t_i}^{t_f} A(t)x(t) + B(t)u(t)dt + F(t)f(t) \\ -C(t)\int_{t_i}^{t_f} A(t)\hat{x}(t) + B(t)u(t) + L_{\bar{\iota}}(t)F_i(t)f_i(t)dt \end{vmatrix}$$

$$e_y(t) = \begin{vmatrix} C(t)\int_{t_i}^{t_f} A(t)(x(t) - \hat{x}(t))dt + F(t)f(t) \\ -C(t)\int_{t_i}^{t_f} L_{\bar{\iota}}(t)F_i(t)f_i(t)dt \end{vmatrix}$$

It has been assumed that $\hat{x}(t) \to x(t)$, as $t \to \infty$.

$$e_y(t) = \left|F(t)f(t) - C(t)\int_{t_i}^{t_f} L_{\bar{\iota}}(t)F_i(t)f_i(t)dt\right| \tag{2.4.12}$$

$$e_e(t) = \left|F_i(t)f_i(t) - C_i(t)\int_{t_i}^{t_f} L_{\bar{\iota}}(t)F_i(t)f_i(t)dt\right|$$

$$e_u(t) = \left|F_{\bar{\iota}}(t)f_{\bar{\iota}}(t) - C_{\bar{\iota}}(t)\int_{t_i}^{t_f} L_{\bar{\iota}}(t)F_i(t)f_i(t)dt\right|$$

For all observers but one, $f_i(t) \neq 0$, and thus $e_u(t) \neq 0$. In the reduced observer where the $\bar{\iota}$ elements set to zero match the k faults located on $f_i(t)$, $L_{\bar{\iota}}(t)F_i(t)f_i(t)$ and $F_{\bar{\iota}}(t)f_{\bar{\iota}}(t)$ will become zero and thus $e_u(t)$ will go to zero. As $e_u(t)$ can be directly measured and only one observer will converge to zero, the supervisor selects that observer's fault free estimates of the outputs and states. In this case, it is shown to be both the necessary and sufficient that only one observer will produce $\hat{y}(t) = y(t)$, thus producing fault free estimates.

Proof 2.4.1: Case 3

In this case, when $t \in [t_i, t_f]$, $f(t)$ is non-zero for a number of elements between zero and k noninclusive. This means that $L_{\bar{i}}(t)F_i(t)f_i(t)$ and $F_{\bar{i}}(t)f_{\bar{i}}(t)$ will be zero for more than one set of predicted faults. This means that multiple observers will have an error $e_u(t)$ of zero. The proof of convergence is the same as in case 2. The supervisor relies on a priority structure to handle this case. Specifically, the supervisor selects the observer that has the least sensors presumed faulty of the reduced order observers that compensate for the sensor faults. This observer is chosen to maximize robustness by using as many of the fault free sensors as possible. Case 3 uses the same proof as case 2, and therefore case 3 is also proven sufficient and necessary for fault free convergence.

Remark 2.4.1

The three cases of Theorem 2.4.1 have been shown sufficient and necessary for zero to k sensor failures inclusively, which completes the proof. Thus, Theorem 2.4.1 is sufficient and necessary for fault convergence of the system of observers. It is worth pointing out that so long as the number of sensor faults is no more than the k tolerable faults, the magnitude and type of fault does not prevent the FTC method from removing the fault, including nonlinear additive faults on linear systems.

This chapter has focused on systems only undergoing sensor faults. Chapter 3 examines techniques to tolerate systems only undergoing actuator faults. First a way to measure actuator redundancy will be defined. Then Augmented State Observers will be explored.

CHAPTER 3: ACTUATOR FAULT TOLERANCE METHODS

3.1: Measuring Actuator Redundancy

Sensor redundancy is commonly available in modern systems. However, some systems exhibit actuator redundancy. The following method provides the details of how to calculate the rank of available actuator redundancy. This rank measures the number of actuators that can fully fail without the system losing controllability.

Assume a linear time-varying system defined as follows.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \tag{3.1.1}$$

$$y(t) = C(t)x(t) + D(t)u(t) \tag{3.1.2}$$

Where $x(t) \in R^n$ is the state vector, $u(t) \in R^m$ is the control vector, and $y(t) \in R^p$ is the output vector. $A(t)$, $B(t)$, $C(t)$, and $D(t)$ are known matrices of dimensions n x n, n x m, p x n and p x m respectively. Time indexes $t_0$and t_f represent the initial and final time respectively. Assume that the system is controllable on $(A(t), B(t))$ and observable on $(A(t), C(t))$ for $t \in [t_0, t_f]$. $W(t_0, t_f)$ and $M(t_0, t_f)$ are n x n Grammian matrices.

$$W(t_0, t_f) = \int_{t_0}^{t_f} \Phi(t_o, t)B(t)B^T(t)\Phi^T(t_o, t)\, dt \tag{3.1.3}$$

$$M(t_0, t_f) = \int_{t_0}^{t_f} \Phi^T(t, t_o) C^T(t) C(t) \Phi(t, t_0)\, dt \qquad (3.1.4)$$

In [39] a proof is presented that states that the state equation is controllable for $t \in [t_0, t_f]$ if and only if $W(t_0, t_f)$ is invertible, and it is observable for $t \in [t_0, t_f]$ if and only if $M(t_0, t_f)$ is invertible. The proof of this will not be repeated here.

Define $B_z(t)$ as the $B(t)$ matrix, with the z^{th} column set to zero, $z \in [1, m]$ and redefine the state space equation as follows.

$$\dot{x}(t) = A(t)x(t) + B_z(t)u(t)$$

This change to the state equations is mathematically equivalent to the z^{th} element of $u(t)$ no longer being defined by equation (3.1.1). This represents a total fault in the z^{th} element of $u(t)$. Define the following n x m matrix K, presuming the existence and continuity of the indicated derivatives.

$$K_0(t) = B_z(t)$$

$$K_j(t) = -A(t)K_{j-1}(t) + \dot{K}_{j-1}(t), j = 1, 2, \ldots$$

Theorem 3.1.1

For all t and σ and non-negative j,

$$\frac{d^j}{d\sigma^j} [\Phi(t, \sigma) B_z(\sigma)] = \Phi(t, \sigma) K_j(\sigma), j = 0,1, \ldots$$

Proof 3.1.1

This proof is shown by induction. First the $j = 0$ case is shown.

$$\frac{d^0}{d\sigma^0}\left[\Phi(t,\sigma)B_z(\sigma)\right] = \Phi(t,\sigma)K_0(\sigma)$$

$$\Phi(t,\sigma)B_z(\sigma) = \Phi(t,\sigma)B_z(\sigma)$$

Then the inductive case is shown.

$$\frac{d^{j+1}}{d\sigma^{j+1}}\left[\Phi(t,\sigma)B_z(\sigma)\right] = \frac{d}{d\sigma}\left[\Phi(t,\sigma)K_j(\sigma)\right]$$

$$= -\Phi(t,\sigma)A(\sigma)K_j(\sigma) + \Phi(t,\sigma)\frac{d}{d\sigma}\left[K_j(\sigma)\right]$$

$$\frac{d^{j+1}}{d\sigma^{j+1}}\left[\Phi(t,\sigma)B_z(\sigma)\right] = \Phi(t,\sigma)K_{j+1}(\sigma) \qquad \text{Q.E.D.}$$

Theorem 3.1.2

Suppose q is a positive integer such that for all $t \in [t_0, t_f]$, $B_z(t)$ is q times continuously differentiable and $A(t)$ is $(q-1)$ times continuously differentiable on $[t_0, t_f]$ for some $t_c \in [t_0, t_f]$. The following test will determine if the system is still controllable, despite the failure of the z^{th} component of $u(t)$.

$$rank\left[\, K_0(t_c) \;\; K_1(t_c) \ldots \;\; K_q(t_c) \,\right] = n \qquad (3.1.5)$$

Proof 3.1.2: Case: Time-varying system

Suppose that $t_c \in [t_0, t_f]$ and satisfies (3.1.5). Setting up a contradiction, it is presumed that $W(t_0, t_f)$ is not invertible. As such, there exists a nonzero n x 1 vector x_a that satisfies the following equation.

$$0 = x_a^T \, W(t_0, t_f) x_a$$

$$0 = \int_{t_0}^{t_f} x_a^T \Phi(t_o, t) B_z(t) B_z^T(t) \Phi^T(t_o, t) x_a \, dt$$

$$0 = \int_{t_0}^{t_f} \| x_a^T \Phi(t_o, t) B_z(t) \|^2 \, dt$$

$$x_a^T \Phi(t_o, t) B_z(t) = 0, t \in [t_0, t_f] \tag{3.1.6}$$

Let x_b be the nonzero vector $x_b = \Phi^T(t_o, t_c) x_a$, and substitute it into (3.1.6).

$$x_b^T \Phi(t_c, t) B_z(t) = 0, t \in [t_0, t_f]. \tag{3.1.7}$$

Choosing $t = t_c$, $x_b^T \Phi(t_c, t_c) K_0 = x_b^T K_0 = 0$. Differentiating (3.1.7) with respect to t gives $x_b^T \Phi(t_c, t) K_1(t) = 0, t \in [t_0, t_f]$ And again, specifying $t = t_c$, shows that $x_b^T K_1 = 0$. In the general case this formula derives into the following.

$$\frac{d^j}{dt^j} [x_b^T \Phi(t_c, t) B_z(t)]_{t=t_c} = x_b^T K_j(t_c) = 0, j = 1,2, \ldots q$$

Therefore $x_b^T [K_0(t_c) \;\; K_1(t_c) \ldots \;\; K_q(t_c)] \;=\; 0$.

This contradicts the linear independence implied by (3.1.5). Q.E.D. Thus (3.1.5) is a sufficient condition for the state equation to be controllable on $[t_0, t_f]$ given the conditions above.

Proof 3.1.2: Case: Time-invariant system

In the case of time-invariant systems, this can be extended to a necessary condition as well as a sufficient one. In the case of a time independent system, the *K* matrix rank test described in (3.1.5) is simplified into the following.

$$rank[B\ AB\ \dots A^{n-1}B] = n \tag{3.1.8}$$

To prove the necessary condition, we presume that the controllability Grammian W is not invertible. Therefore, there exists a nonzero n x 1 vector x_a that satisfies the following equation.

$$x_a^T\, W(t_0, t_f) x_a = 0$$

$$x_a^T \int_{t_0}^{t_f} e^{A(t_o - t)} B_Z B_Z^T e^{A^T(t_o - t)}\, dt = 0$$

$$x_a^T \int_{t_0}^{t_f} \left\| e^{A(t_o - t)} B_z \right\|^2 dt = 0$$

$$x_a^T\, e^{A(t_o - t)} B_Z = 0\ , t\ \in [t_0, t_f] \tag{3.1.9}$$

Differentiating (3.1.9) k times, and setting $t = t_0$ gives $(-1)^k x_a^T A^k B = 0, k = 0,1 \dots n-1$. As such, (3.1.9) derives into the following.

$$x_a^T[B\ AB\ \dots A^{n-1}B] = 0$$

Which proves that (3.1.8) fails, and thus the condition is both necessary and sufficient in the time-invariant case.

Remark 3.1.1

Define $R_C(z)$ as an n x 1 vector.

$$R_C(z) := rank\big[\ K_0(t_c)\ \ K_1(t_c) \dots\ \ K_q(t_c)\ \big], z \in [1, m] \tag{3.1.10}$$

This vector can be analyzed for information regarding controllability of the system. If $minimum(R_C(z)) = n$ for all $z \in [1, m]$, the system is controllable in the presence of a single actuator fault, regardless of which actuator is faulty. The system has rank 1 actuator redundancy. In a time-variant system, if $R_C(z) < n$, a fault in the z^{th} component of $u(t)$ may cause a loss of controllability. In a time-invariant system, if $R_C(z) < n$, a fault in the z^{th} component of $u(t)$ will cause a loss of controllability. This is because the R_C test is necessary and sufficient in the time-invariant case.

Remark 3.1.2

Define u_f as the set of all control elements $\{1, 2, \dots, m\}$. Choose r as an integer from $[1, m)$, representing the total number of simultaneous faults that can occur. Define $zz(i)$ as the i^{th} combination of set u_f, for $i \in [1, \frac{m!}{r!(m-r)!}]$. The full set of combinations is defined as

$$zz = u_f \binom{r}{m}, \mathrm{r} \in [1, m)$$

Further define $B_{zz}(t)$ as the $B(t)$ matrix with all columns in the set zz set to zero. Presume a K matrix as follows, subject to the existence and continuity of all indicated derivatives.

$$K_0(t) = B_{zz}(t)$$

$$K_j(t) = -A(t)K_{j-1}(t) + \dot{K}_{j-1}(t), j = 1, 2, \ldots$$

Define $RR_c\big(zz(i)\big)$ as an i x 1 vector.

$$RR_C\big(zz(i)\big) = rank\big[\, K_0(t_c) \;\; K_1(t_c) \ldots \;\; K_q(t_c) \,\big] \qquad (3.1.11)$$

The measure $RR_c\big(zz(i)\big)$ can be analyzed in the same way as the $R_C(z)$ vector. Instead of a single fault, the measure determines rank r actuator redundancy. That is, if $minimum(RR_C(zz(i))) = n$ for all i, the system is controllable in the presence of up to r actuator faults. As before, in the time-invariant case, for any $RR_C\big(zz(i)\big) < n$, $zz(i)$ is the set of faults that will cause a loss of controllability.

Remark 3.1.3

This section produces a pair of measures that can be examined to determine the rank of available actuator redundancy. These measures are useful and can provide designers with information about how the system responds to actuator faults.

Unfortunately, most systems do not have sufficient actuator redundancy for any single actuator to fail due to the cost of actuator components. As such, this measure is less applicable than the measure for sensor redundancy. Because of this limitation, the next section examines using Augmented State Observers to produce actuator fault tolerance.

3.2: Augmented State Observer

This portion of the research designs a full order Augmented State Observer to tolerate actuator faults. The ASO design uses eigen value assignment feedback to control the response of the original plant. This is done by incorporating an adaptive controller that uses the state and actuator fault estimates from the observer. The ASO adds additional states to the observer's state space model that correspond to estimates of the actuator faults. These additional states allow the ASO to estimate actuator faults. The controller uses the actuator fault estimates to compensate for the faults through the controller. This technique is shown to be sufficient for time-varying and time-invariant systems, presuming additive independent faults on the actuators. In the next chapter, this ASO is incorporated into the fault tolerant method designed in sections 2.1-2.4 so that the system can tolerate both sensor and actuator failures.

Assume a linear time-variant system as follows.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + E(t)v(t) \tag{3.2.1}$$

$$y(t) = C(t)x(t) + F(t)f(t) \tag{3.2.2}$$

Where $x(t) \in R^n$ is the state vector, $u(t) \in R^m$ is the control vector, $y(t) \in R^p$ is the output vector, $f(t) \in R^q$ is the fault vector for sensors, and $v(t) \in R^r$ is the actuator error vector. The fault vectors $f(t)$ and $v(t)$ are unknown but bounded vectors. The matrices $A(t)$, $B(t)$, $C(t)$, $E(t)$, and $F(t)$ are known and have dimensions n x n, n x m, p x n, n x r, and p x q respectively. Assume the linear system is observable on $(A(t), C(t))$ and that it is controllable on $(A(t), B(t))$. The $F(t)$ matrix represents how

the additive faults interact with the various sensors. Without loss of generality, the $F(t)$ matrix can be defined as having ones on the main diagonal and zeros everywhere else. When $p = q$, the $F(t)$ matrix is defined as an identity matrix. The $E(t)$ matrix represents how additive actuator faults interact with the states. Without loss of generality, the $E(t)$ matrix can be defined as having ones on the main diagonal and zeros everywhere else. When $n = r$, the $E(t)$ matrix is defined as an identity matrix.

Augmented State Observer Formulation

The design of an Augmented State Observer (ASO) allows the system to adapt to actuator faults. The full order ASO is not designed to tolerate sensor failures. As such, the ASO's estimation assumes that $f(t) = 0$, for all t. The state space of the ASO is augmented by adding additional states that estimate the actuator faults. The vectors $\hat{x}(t)$ and $\hat{v}(t)$ are estimates of the states and actuator faults respectively. The subscript a is used to denote a vector or matrix that has been augmented to correspond to these additional states.

$$\hat{x}_a = \begin{bmatrix} \hat{x} \\ \hat{v} \end{bmatrix}$$

These estimates are used to create an adaptive state feedback controller.

$$u(t) = r(t) - K_x\hat{x}(t) - K_v\hat{v}(t) \tag{3.2.3}$$

The original input to the system is defined as the reference signal $r(t)$. The $K_x(t)$ term is a gain matrix that performs state feedback for the original system. $K_x(t)$ is

determined by using pole placement in order to improve performance and stability of the original system. $K_v(t)$ is the feedback gain matrix that adapts the controller to the presence of actuator faults.

Based on the system defined by (3.2.1) and (3.2.2) and the adaptive controller from (3.2.3), a state and actuator fault observer is designed as follows. The time references are omitted for legibility.

$$\dot{\hat{x}} = A\hat{x} + Bu + L_x(y - C\hat{x})$$

$$\dot{\hat{x}} = A\hat{x} + Br - BK_x\hat{x} - BK_v\hat{v} + L_x(y - C\hat{x}) \tag{3.2.4}$$

$$\hat{v} = L_v(y - C\hat{x}) \tag{3.2.5}$$

The state estimation equation and actuator fault estimation equations of (3.2.4) and (3.2.5) are combined into a single equation that describes the ASO. The time references are omitted for legibility.

$$\underbrace{\begin{bmatrix}\hat{x}\\ \hat{v}\end{bmatrix}}_{\dot{\hat{x}}_a} = \left(\underbrace{\begin{bmatrix}A - BK_x & -BK_v\\ 0 & 0\end{bmatrix}}_{A_a} - \underbrace{\begin{bmatrix}L_x\\ L_v\end{bmatrix}}_{L_a}\underbrace{\begin{bmatrix}C & 0\end{bmatrix}}_{C_a}\right)\underbrace{\begin{bmatrix}\hat{x}\\ \hat{v}\end{bmatrix}}_{\hat{x}_a} + \underbrace{\begin{bmatrix}B\\ 0\end{bmatrix}}_{B_a} r + \underbrace{\begin{bmatrix}L_x\\ L_v\end{bmatrix}}_{L_a} y \tag{3.2.6}$$

$$\dot{\hat{x}}_a = (A_a - L_aC_a)\hat{x}_a + B_a r + L_a y \tag{3.2.7}$$

The observer gains that have to be designed are L_x and L_v. L_x is the observer gain matrix for state estimation, while L_v is the observer gain matrix for actuator fault estimation. The state estimation error is given in (3.2.8).

$$e_x(t) = x(t) - \hat{x}(t) \tag{3.2.8}$$

Equations (3.2.1), (3.2.2), (3.2.3), and (3.2.6) are combined with the derivative of (3.2.8) to find the state space error equation. The time references are omitted for legibility.

$$\dot{e}_x = (Ax + Bu + Ev) - (A\hat{x} + Bu + L_x(y - C\hat{x}))$$

$$\dot{e}_x = (Ax + B(r - K_x x) + Ev) - (A\hat{x} + B(r - K_x\hat{x} - K_v\hat{v}) + L_x(Cx - C\hat{x}))$$

$$\dot{e}_x = A(x - \hat{x}) + B(r - r) - BK_x(x - \hat{x}) + BK_v\hat{v} - L_xC(x - \hat{x}) + Ev$$

$$\dot{e}_x = Ae_x - BK_xe_x + BK_v\hat{v} - L_xCe_x + Ev$$

$$\dot{e}_x = (A - BK_x - L_xC)e_x + BK_v\hat{v} + Ev$$

$$\begin{bmatrix} \dot{e}_x \\ \dot{\hat{v}} \end{bmatrix} = \left(\underbrace{\begin{bmatrix} A - BK_x & BK_v \\ 0 & 0 \end{bmatrix}}_{A_e} - \underbrace{\begin{bmatrix} L_x \\ L_v \end{bmatrix}}_{L_e} \underbrace{\begin{bmatrix} C & 0 \end{bmatrix}}_{C_e} \right) \begin{bmatrix} e_x \\ \hat{v} \end{bmatrix} + \begin{bmatrix} E \\ 0 \end{bmatrix} v \tag{3.2.9}$$

In [17] it was shown that if L_a is chosen properly, there exists a symmetric positive definite matrix $P \in R^{(n+q)}$ that will satisfy (3.2.10) such that $\alpha > 0$ and $\gamma \geq \|v(t)\|$.

$$P(A_e - L_eC_e) + (A_e - L_eC_e)^T P = -\alpha\gamma I \tag{3.2.10}$$

Equation (3.2.10) ensures that $e_x(t)$ and $\hat{v}(t)$ will be bounded independently of the initial conditions $x(0)$ or $v(0)$. Also, (3.2.10) ensures that the observer's state estimation error and actuator fault estimation are both convergent and bounded. In

general, as $t \to \infty$, $\hat{x}(t) \to x(t)$ and $\hat{v}(t) \to v(t)$. The ASO defined in (3.2.6) will converge to a region around the fault free states and the actuator faults when there are no sensor faults.

The next chapter uses this ASO design that tolerates actuator faults and combines it with the reduced order observer design of chapter 2 that tolerates sensor faults. By combining these two methods, both sensor and actuator faults can be tolerated simultaneously by the FTC system.

CHAPTER 4: ACTUATOR AND SENSOR FAULT TOLERANCE

4.1: Reduced Observer Formulation

In this section, a set of reduced order Augmented State Observers (r-ASO) are built in addition to the initial Augmented State Observer (ASO). The r-ASOs are designed to tolerate sensor faults in addition to actuator faults. As such, the sensor fault vector $f(t)$ can be nonzero after time t_e. These reduced observers require sensor redundancy. The available redundancy in the system is calculated by the method developed in section 2.1. The rank of sensor redundancy in the plant is defined as k. The total number of r-ASOs that must be designed is found by performing the permutation of the p sensors with k or less potentially faulty sensors. One r-ASO is built for each permutation.

Each reduced order augmented state observer is designed according to (3.2.6) and starts with the same K_x, K_v, and L_a as was designed for the ASO. Each reduced order observer is designed to remove a specific sensor set from its estimates. Let i represent the set of sensors that a specific r-ASO presumes are faulty. Each reduced observer ignores the $y_i(t)$ outputs that correspond to the set of i sensors, removing them from the observer's estimation feedback process. Each r-ASO also assumes all other sensors, $y_{\bar{i}}(t)$, are operating without fault.

For each of these r-ASOs, the presumed faulty y_i sensors are removed from the estimation process. This is done by setting the corresponding i columns of the L_a matrix in (3.2.6) to zero. This removes the presumed faulty sensor data from the ASO's estimation process. As an r-ASO has been designed for each of the possible fault sets, at least one r-ASO will correctly remove the faulty sensor data.

The r-ASOs are not able to localize sensor faults by themselves so they require a supervisor system similar to the one developed in section 2.4. Presuming that there are no more than k sensor faults, there must exist at least one r-ASO that correctly identifies and isolates the faulty sensors. This means that the convergence determined by (3.2.10) is still valid when $f(t) \neq 0$. As such, the r-ASOs produce bounded and convergent estimates of the states and actuator faults.

Each r-ASO assumes that a specific set of sensors are faulty and assumes that the others are fault free. If an r-ASO does not match the sensor faults correctly, faulty data will be used in the observer's feedback. The r-ASO's mismatch will initially be detected by examining the unexpected sensor error $e_{yu}(t)$.

$$e_{yu}(t) = |y_{\bar{\imath}}(t) - \hat{y}_{\bar{\imath}}(t)| \tag{4.1.1}$$

$y_{\bar{\imath}}(t)$ is the subset of the outputs that were presumed fault free by a specific r-ASO. If this set of outputs is not fault free, the ASO will be operating with faulty sensor data. These unmitigated sensor errors will be propagated to the feedback of the observer. The observer will then treat the unmitigated sensor fault as an unknown actuator fault, causing errors with its $\hat{v}(t)$ estimation. The presence of these two types of errors is used

by the supervisor to detect that an observer does not match the sensor fault set and is not an optimal estimate. The supervisor selects the observer that has the lowest unexpected error $e_u(t)$ as the optimal estimator for the sensor faults.

$$e_u(t) = \frac{e_{yu}(t)}{\|\cup e_{yu}(t)\|} + \frac{\hat{v}(t)}{\|\cup \hat{v}(t)\|} \tag{4.1.2}$$

The norms of the unexpected sensor failures and actuator fault estimates are calculated on the set of all of the unexpected errors for all of the observers. This is done to normalize the errors so that they can be compared across multiple observers. The optimal observer will converge to the lowest $e_u(t)$, but is not guaranteed to be the lowest for any time t. When the system is operating in the presence of an actuator fault the lowest $e_u(t)$ will not be zero. Once the supervisor selects the optimal ASO, it propagates that observer's state and actuator fault estimates. The supervisor then calculates sensor failure estimates $\hat{f}(t)$ by comparing the $y_i(t)$ outputs with the $\hat{y}_i(t)$ estimates as was shown in section 2.3.

Theorem 4.1.1

Consider a plant defined by (3.2.1) and (3.2.2), with at least rank k sensor redundancy. An augmented state observer is built using the model given in (3.2.6). The set of all tolerable sensor fault sets is found by permuting the p sensors with the k rank of redundancy. One reduced order augmented state observer is built by the model given in (3.2.6) for each tolerable sensor fault set. Each reduced order observer sets the columns of L_a that correspond to their sensor fault set to zero.

A supervisor that chooses the reduced order observer with the lowest $e_u(t)$ at a given t selects the best estimate of the states, outputs, and faults. As a special case, when multiple observers have a similar $e_u(t)$ at a given t, the supervisor selects the observer with the smallest sensor fault set size. This maximizes the FTC's robustness against noise.

Proof 4.1.1: Case: Sensor faults only

This section shows the convergence of at least one r-ASO to zero estimation error in the presence of no more than k sensor faults and zero actuator faults. In this case, $f(t)$ is nonzero for no more than k elements for any given t and $v(t)$ is zero. The notation i is used to reference a set of no more than k sensor elements. The subscript i references the subset of columns of a matrix or elements of a vector that match the elements in i. The subscript $\bar{\imath}$ will be used to reference a matrix that has the i columns set to zero, or a vector that has the i elements set to zero. An r-ASO has been designed for each permutation of up to k sensor faults, and each has been designed to remove a specific set of $y_i(t)$ outputs from the estimation process. Assume that the set i represents the faulty sensors in the system. Faulty outputs are observed on the i sensors, $y_i(t)$. The sensor fault vector $f(t)$ is zero for all elements other than $f_i(t)$. Therefore, one of the r-ASOs does not use the faulty $y_i(t)$ sensor data in its feedback, and only references $y_{\bar{\imath}}(t)$. For that specific r-ASO, it follows that equation (3.2.6) is reduced to the form below. The time indexes are omitted for legibility.

$$\begin{bmatrix}\hat{x}\\ \hat{v}\end{bmatrix} = \left(\begin{bmatrix}A - BK_x & -BK_v\\ 0 & 0\end{bmatrix} - \begin{bmatrix}L_{x_{\bar{\imath}}}\\ L_{v_{\bar{\imath}}}\end{bmatrix}\begin{bmatrix}C & 0\end{bmatrix}\right)\begin{bmatrix}\hat{x}\\ \hat{v}\end{bmatrix} + \begin{bmatrix}B\\ 0\end{bmatrix}r + \begin{bmatrix}L_{x_{\bar{\imath}}}\\ L_{v_{\bar{\imath}}}\end{bmatrix}y_{\bar{\imath}} \qquad (4.1.3)$$

The r-ASO's output estimate is determined by (4.1.4).

$$\hat{y}(t) = C(t)\hat{x}(t) \tag{4.1.4}$$

As the order of i is less than or equal to k, the system defined by (4.1.3) and (4.1.4) is still observable, as was shown in section 2.1. Equation (3.2.2) is substituted into (4.1.4) and it is assumed that L_a is properly designed so that as $t \to \infty$, $x(t) - \hat{x}(t) \to 0$.

$$\hat{y}(t) = y(t) - F(t)f(t)$$

The sensor fault vector $f(t)$ is zero for all elements other than the i^{th} elements. This output estimate is divided into two subsets: $\hat{y}_{\bar{\imath}}(t)$ and $\hat{y}_i(t)$.

$$\hat{y}_{\bar{\imath}}(t) = y_{\bar{\imath}}(t) \tag{4.1.5}$$

$$\hat{y}_i(t) = y_i(t) - F_i(t)f_i(t) \tag{4.1.6}$$

Equation (4.1.5) shows that the r-ASO does not introduce new errors into the fault free outputs, $y_{\bar{\imath}}(t)$. It can be observed from (4.1.6) that the error of the r-ASO's estimation of $y_i(t)$ is the fault effect, $F_i(t)f_i(t)$. Therefore, the r-ASO's estimate $\hat{y}(t)$ is a fault-free estimate of the outputs when there are no actuator faults.

This shows that there is at least one r-ASO that converges to the fault free outputs. This is sufficient to guarantee that up to k sensor faults can be compensated for by the

FTC system. Furthermore, as both the estimate of the fault free sensors and the measured faulty sensors are available to the supervisor, it is able to calculate an estimate of the fault signal $f(t)$ using the method described in section 2.3.

Proof 4.1.1: Case: Actuator faults in the presence of sensor faults

It has been shown that within the bank of r-ASOs, there exists at least one observer that removes the effects of $f(t)$ from the feedback. This means that sensor faults have no effect on the state and actuator fault estimates on the optimal r-ASO. Therefore, previous proofs of ASO convergence in the presence of actuator faults do not need to be modified in the case when there are sensor faults. The full proof that an ASO that satisfies (3.2.10) has bounded and convergent state errors and actuator fault estimates when in the presence of actuator faults is found in [17] and not repeated here. This design ensures that the estimate $\hat{x}(t)$ converges to a region around $x(t)$.

Proof 4.1.1: Case: Sensor faults in the presence of actuator faults

In the presence of actuator faults, $\hat{x}(t)$ is no longer guaranteed to converge exactly to $x(t)$. Instead, the difference is bounded within a region around the equilibrium. This complicates the derivations that produced the fault free output estimates from (4.1.5) and (4.1.6). The error between the states and their estimates is $e_x(t)$. The following is derived to calculate the error between the estimate of the outputs and the fault free outputs, when both sensor and actuator faults are occurring. This is obtained by first rewriting (3.2.8) into the following.

$$e_x(t) = x(t) - \hat{x}(t)$$

$$\hat{x}(t) = x(t) - e_x(t) \tag{4.1.7}$$

Equation (4.1.7) is then substituted into (4.1.4).

$$\hat{y}(t) = C(t)\hat{x}(t)$$

$$\hat{y}(t) = C(t)\big(x(t) - e_x(t)\big)$$

$$\hat{y}(t) = C(t)x(t) - C(t)e_x(t) \tag{4.1.8}$$

Equation (3.2.2) is modified as follows.

$$y(t) = C(t)x(t) + F(t)f(t)$$

$$C(t)x(t) = y(t) - F(t)f(t) \tag{4.1.9}$$

Equation (4.1.9) is substituted into (4.1.8).

$$\hat{y}(t) = y(t) - F(t)f(t) - C(t)e_x(t) \tag{4.1.10}$$

The error between the theoretical fault free output and the estimated output is defined as $e_y(t)$.

$$e_y(t) = C(t)x(t) - \hat{y}(t) \tag{4.1.11}$$

Equations (4.1.9) and (4.1.10) are substituted into (4.1.11).

$$e_y(t) = (y(t) - F(t)f(t)) - \big(y(t) - F(t)f(t) - C(t)e_x(t)\big)$$

$$e_y(t) = C(t)e_x(t) \tag{4.1.12}$$

Equation (4.1.12) shows the relationship between the state error $e_x(t)$ and the sensor error $e_y(t)$. It has been shown that $e_x(t)$ is bounded and convergent, so $e_y(t)$ is also bounded and convergent. The error in (4.1.12) is present on both faulty sensors and fault free sensors. This means that if the state estimate errors do not converge to zero, the r-ASO will propagate state estimate errors to the output estimates, causing sensor faults to appear on non-faulty sensors. Because of this, it is important to design the L_a matrix in such a way that the state estimate error converges quickly and to zero. This will ensure that additional sensor errors are not introduced by state estimation errors. The supervisor that selects as to which of the observers produces the optimal estimates is defined in the next section.

4.2: Supervisor Formulation

A supervisor is designed to select the full or reduced ASO that has the optimal estimate of the fault free states. The supervisor chooses the estimator that has the lowest unexpected error, $e_u(t)$. That specific ASO's estimates of the actuator faults, states, and sensors are sent to other components of the system and used in feedback. As all of the ASOs are built using the same fundamental L_a structure in the observer gain matrix, their errors will be bounded in a similar region around the fault-free estimates of the system states, actuator faults, and sensor faults. Due to variations in the system specifics caused by the variations in the L_a gain matrix, their rates of convergence may vary. The r-ASO form is repeated from (4.1.3) and (4.1.4) below. The time indexes of (4.1.3) are omitted for legibility.

$$\begin{bmatrix}\dot{\hat{x}}\\ \dot{\hat{v}}\end{bmatrix} = \left(\underbrace{\begin{bmatrix} A - BK_x & -BK_v \\ 0 & 0 \end{bmatrix}}_{A_a} - \underbrace{\begin{bmatrix} L_{x_{\bar{\imath}}} \\ L_{v_{\bar{\imath}}} \end{bmatrix}}_{L_{a\bar{\imath}}} \underbrace{\begin{bmatrix} C & 0 \end{bmatrix}}_{C_a} \right) \underbrace{\begin{bmatrix}\hat{x}\\ \hat{v}\end{bmatrix}}_{\hat{x}_a} + \underbrace{\begin{bmatrix}B\\ 0\end{bmatrix}}_{B_a} r + \underbrace{\begin{bmatrix} L_{x_{\bar{\imath}}} \\ L_{v_{\bar{\imath}}} \end{bmatrix}}_{L_{a\bar{\imath}}} y_{\bar{\imath}}$$

$$\hat{y}(t) = C(t)\hat{x}(t)$$

The set of all potential sensor faults is limited to the permutations of up to k sensor faults on the p total sensors. This k is no larger than the available redundancy measured in section 2.1. The observer matrix L_a is designed such that (3.2.10) is valid for all variations of $L_{a\bar{\imath}}$ so that as $t \to \infty$, $\hat{x}(t)$ converges to an area around $x(t)$, for all r-ASOs. The general form of the r-ASO with all of its outputs to the supervisor is given in the pair of equations below. The time indexes are omitted for legibility.

$$\dot{\hat{x}}_a = (A_a - L_{a\bar{\imath}}C_a)\hat{x}_a + B_a r + L_{a\bar{\imath}}y \tag{4.2.1}$$

$$\begin{bmatrix} \hat{y} \\ \hat{x} \\ \hat{v} \end{bmatrix} = \begin{bmatrix} C_a \\ I \end{bmatrix} \hat{x}_a \tag{4.2.2}$$

Equations (4.2.1), (3.2.1), and (3.2.2) are combined into the derivative of (3.2.8), assuming $\hat{x}(0) = x(0)$, $t \in [t_i, t_f]$, N is an n x n identity matrix, and 0 is an n x r null matrix. The time indexes are omitted for legibility.

$$\dot{e}_x = Ax + Bu + Ev - [N \quad 0]((A_a - L_{a\bar{\imath}}C_a)\hat{x}_a + B_a u - L_{a\bar{\imath}}y)$$

$$\dot{e}_x = A(x - \hat{x}) + B(u - u) + Ev + L_{x\bar{\imath}}C\hat{x} - L_{x\bar{\imath}}(Cx + Ff) \tag{4.2.3}$$

The sensor fault vector $f(t)$ is only nonzero for a number of rows less than or equal to k. An r-ASO has been designed for each tolerable permutation of the sensor fault set. As such, there exists at least one r-ASO where $f(t)$ is nonzero on the i rows that are set to zero on $L_{a\bar{\imath}}(t)$, and zero on the other rows. This means that the sensor faults are removed from the system. On that r-ASO, the error estimate (4.2.3) is further derived. The time indexes are omitted for legibility.

$$\dot{e}_x = A(x - \hat{x}) + Ev + L_{x\bar{\imath}}C\hat{x} - L_{x\bar{\imath}}Cx$$

$$\dot{e}_x = A(x - \hat{x}) + Ev - L_{x\bar{\imath}}C(x - \hat{x}) \tag{4.2.4}$$

Equation (3.2.8) describes the error between the states and their estimates and is substituted into (4.2.4).

$$\dot{e}_x(t) = A(t)e_x(t) - L_{x\bar{\imath}(t)}(t)C(t)e_x(t) + E(t)v(t)$$

$$\dot{e}_x(t) = (A(t) - L_{x\bar{\imath}}(t)C(t))e_x(t) + E(t)v(t) \quad (4.2.5)$$

The form of equation (4.2.5) is the same as (3.2.10) and the same derivations hold true. As such, the error is convergent and bounded. In ASOs that do not remove the $f(t)$ component properly, there will be an additional error term added in the coefficient of $\dot{e}_x(t)$. The errors in $x(t)$ are not directly measurable by the supervisor, so the supervisor examines the unexpected errors $e_u(t)$. As a properly designed L_a forces a sufficiently fast convergence of $e_x(t)$ to zero, the observer with the most accurate estimates of the system converges to the lowest unexpected error.

Supervisor Selection Special Case

It is possible for the set of active sensor faults to be a subset of multiple tolerable sensor fault sets. This leads to a situation where multiple r-ASOs will converge to the same unexpected error. In the case where multiple r-ASOs converge to the same unexpected error, the one with the least sensors presumed faulty is used by the supervisor to preserve as many of the original outputs as possible and produce the most robust estimates. In the trivial case where there are no faults, all r-ASOs will converge and the full ASO is selected by the supervisor to maximize robustness.

CHAPTER 5: APPLICATION OF SENSOR FAULT TOLERANCE

5.1: Sensor Redundancy Calculation

A three input, three output, linear time-invariant model for a turbofan engine is provided by in reference [9] and is considered here. Specifically, the model for the fan operating under a Power Code of 30 is given. The three inputs $u(t)$ correspond to actuators controlling the fuel flow rate, the nozzle area, and the bypass duct area. The three outputs $y(t)$ correspond to sensors measuring the fan speed, core engine pressure ratio, and the overall engine pressure ratio. The system follows the state transition definition from (5.1.1) and (5.1.2).

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{5.1.1}$$

$$y(t) = Cx(t) + Du(t) + Ff(t) \tag{5.1.2}$$

$$A = \begin{bmatrix} -2.8451 & 0.8116 & 0.0581 \\ 0.2584 & -2.6778 & 0.0584 \\ -0.0133 & -0.1305 & -0.1202 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.2686 & 0.0834 & -0.0087 \\ 0.2676 & 0.0334 & 0.0048 \\ 0.0660 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 8.7379 & 0 & 0 \\ -2.3615 & 3.2833 & 0.0579 \\ 1.7798 & -2.3684 & -0.0349 \end{bmatrix}$$

The matrix D is a zero matrix of appropriate size. The method to calculate the available sensor redundancy is shown here by example.

R_O example

The test for sensor redundancy is formulated. Again, there are three $R_O(z)$ tests to set up. The first case is $R_O(1)$.

$$L_0 = C_z = \begin{bmatrix} 0 & 0 & 0 \\ -2.3615 & 3.2833 & 0.0579 \\ 1.7798 & -2.3684 & -0.0349 \end{bmatrix}$$

$$L_1 = L_0 A + \dot{L}_0 = \begin{bmatrix} 0 & 0 & 0 \\ 7.5663 & -10.716 & 0.0457 \\ -5.6752 & 7.7911 & -0.0307 \end{bmatrix}$$

$$L_2 = L_1 A + \dot{L}_1 = \begin{bmatrix} 0 & 0 & 0 \\ -24.297 & 34.830 & -0.1917 \\ 18.160 & -25.465 & 0.1290 \end{bmatrix}$$

$$R_O(1) = rank(L) = 3$$

The second case is $R_O(2)$.

$$L_0 = C_z = \begin{bmatrix} 8.7379 & 0 & 0 \\ 0 & 0 & 0 \\ 1.7798 & -2.3684 & -0.0349 \end{bmatrix}$$

$$L_1 = \begin{bmatrix} -24.860 & 7.0917 & 0.5077 \\ 0 & 0 & 0 \\ -5.6752 & 7.7911 & -0.0307 \end{bmatrix}$$

$$L_2 = L_1 A + \dot{L}_1 = \begin{bmatrix} 72.5555 & -39.2329 & -1.0912 \\ 0 & 0 & 0 \\ 18.1603 & -25.4651 & 0.1290 \end{bmatrix}$$

$$R_O(2) = rank(L) = 3$$

The third case is $R_O(3)$.

$$L_0 = C_z = \begin{bmatrix} 8.7379 & 0 & 0 \\ -2.3615 & 3.2833 & 0.0579 \\ 0 & 0 & 0 \end{bmatrix}$$

$$L_1 = L_0 A + \dot{L}_0 = \begin{bmatrix} -24.860 & 7.0917 & 0.5077 \\ 7.5663 & -10.716 & 0.0475 \\ 0 & 0 & 0 \end{bmatrix}$$

$$L_2 = L_1 A + \dot{L}_1 = \begin{bmatrix} 72.5555 & -39.2329 & -1.0912 \\ -24.2967 & 34.8304 & -0.1919 \\ 0 & 0 & 0 \end{bmatrix}$$

$$R_O(3) = rank(L) = 3$$

The complete set of sensors are tested by setting up the R_o vector to find that it is equal to [3,3,3]. This shows that the sensors are arranged in such a way that there is some redundancy. The loss of any single sensor does not prevent the system from being observable. This system has sensor redundancy of rank 1.

RR_o example

By choosing $k = 2$, the RR_o vector can be calculated for all permutations of two sensor failures. This leads to three cases again. The total set of y_i fault permutations is found to be $zz = \{(1,2), (1,3), (2,3)\}$. The first case is $RR_O(1)$.

$$L_0 = C_{zz} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1.7798 & -2.3684 & -0.0349 \end{bmatrix}$$

$$L_1 = L_0 A + \dot{L}_0 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ -5.6752 & 7.7911 & -0.0307 \end{bmatrix}$$

$$L_2 = L_1A + \dot{L}_1 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 18.160 & -25.4651 & 0.1290 \end{bmatrix}$$

$$RR_O(1) = rank(L) = 3$$

The second case is $RR_O(2)$.

$$C_{zz} = L_0 = C_{zz} = \begin{bmatrix} 0 & 0 & 0 \\ -2.3615 & 3.2833 & 0.0579 \\ 0 & 0 & 0 \end{bmatrix}$$

$$L_1 = L_0A + \dot{L}_0 = \begin{bmatrix} 0 & 0 & 0 \\ 7.5663 & -10.7162 & 0.0476 \\ 0 & 0 & 0 \end{bmatrix}$$

$$L_2 = L_1A + \dot{L}_1 = \begin{bmatrix} 0 & 0 & 0 \\ -24.2967 & 34.8304 & -0.1919 \\ 0 & 0 & 0 \end{bmatrix}$$

$$RR_O(2) = rank(L) = 3$$

The last case is $RR_O(3)$.

$$L_0 = C_{zz} = \begin{bmatrix} 8.7379 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$L_1 = L_0A + \dot{L}_0 = \begin{bmatrix} -24.8602 & 7.0917 & 0.5077 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$L_2 = L_1A + \dot{L}_1 = \begin{bmatrix} 72.5555 & -39.2329 & -1.0912 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$R_O(1) = rank(L) = 3$$

$$RR_o = [3,3,3]$$

The measure shows that in the situation where any two sensors have failed, the resulting system is still observable. As such, the system has rank 2 sensor redundancy. A system like this is highly desirable, with well designed redundancy. In the next section, this redundancy will be used to design a series of reduced order Kalman observers.

5.2: Layout of Reduced Order Observers

A three input, three output, linear time-invariant model for a turbofan engine is provided by [9]. Specifically, the model for the fan operating under a Power Code of 30 is used. The three inputs $u(t)$ correspond to actuators controlling the fuel flow rate, the nozzle area, and the bypass duct area. The three outputs $y(t)$ correspond to sensors measuring the fan speed, core engine pressure ratio, and the overall engine pressure ratio. The state space model is defined by (5.2.1) and (5.2.2).

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{5.2.1}$$

$$y(t) = Cx(t) + Du(t) + Ff(t) \tag{5.2.2}$$

$$A = \begin{bmatrix} -2.8451 & 0.8116 & 0.0581 \\ 0.2584 & -2.6778 & 0.0584 \\ -0.0133 & -0.1305 & -0.1202 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.2686 & 0.0834 & -0.0087 \\ 0.2676 & 0.0334 & 0.0048 \\ 0.0660 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 8.7379 & 0 & 0 \\ -2.3615 & 3.2833 & 0.0579 \\ 1.7798 & -2.3684 & -0.0349 \end{bmatrix}$$

The zero matrix D is of appropriate size and F is an identity matrix of appropriate size. The fault vector $f(t)$ is an unknown signal. Section 5.1 showed that this system has rank 2 redundancy in all three sensors. In this case, only one sensor is presumed to fail at a time. As such, $k = 1$.

A full order output observer is designed based on (5.2.3) and (5.2.4). The vectors $\hat{x}(t)$ and $\hat{y}(t)$ estimate the state and output vectors respectively. The 3 x 3 matrix L is

chosen to ensure error convergence by pole placement of the estimator. In this case, LC is designed to be a diagonal matrix with poles that converge over one order of magnitude faster than A.

$$\hat{\dot{x}}(t) = (A - LC)\hat{x}(t) + Ly(t) \tag{5.2.3}$$

$$\hat{y}(t) = C\hat{x}(t) \tag{5.2.4}$$

$$L = \begin{bmatrix} 3.5 & 0.1 & 0 \\ 2.8 & -31 & -33.2 \\ -13.2 & 2115 & 1881.9 \end{bmatrix}$$

The full order observer will converge quickest when there are no faulty sensors, providing an accurate estimate of the outputs. A series of reduced order observers are constructed, each presuming one sensor is faulty. The first of which is detailed here. By (5.2.3) and section 2.2, the first matrix is set up with $i = 1$.

$$L_{\bar{1}} = \begin{bmatrix} 0 & 0.1 & 0 \\ 0 & -31 & -33.2 \\ 0 & 2115 & 1881.9 \end{bmatrix}$$

The first column is set to zero as the system is designed to ignore the measurement from the first sensor. The second and third reduced observers are built similarly, with the second and third columns set to zero respectively.

$$L_{\bar{2}} = \begin{bmatrix} 3.5 & 0 & 0 \\ 2.8 & 0 & -33.2 \\ -13.2 & 0 & 1881.9 \end{bmatrix}$$

$$L_{\bar{3}} = \begin{bmatrix} 3.5 & 0.1 & 0 \\ 2.8 & -31 & 0 \\ -13.2 & 2115 & 0 \end{bmatrix}$$

The same state space observer definitions from (5.2.3) and (5.2.4) are used in all three reduced observers and the full observer. This speeds up the design process and ensures that the rates of convergence between all observers are similar. The fluctuations in the rate of convergence between the observers are shown in Table 5.2.1, which lists the eigenvalues of each observer and the original system.

Table 5.2.1: Eigenvalues of the turbofan engine and its observers

Observer	Eigenvalues in order from most dominant to least dominant		
Turbofan Engine	-0.124	-2.292	-3.226
Full Observer	-18.433±12.208i	-32.135	
Reduced Observer 1	-2.595	-18.125±12.344i	
Reduced Observer 2	-10.324	-26.223	-32.779
Reduced Observer 3	-11.978±18.785i	-32.165	

Each of the three reduced observers estimate the internal states $\hat{x}(t)$. Once the $\hat{x}(t)$ estimates are obtained from the reduced order observer, the $\hat{y}(t)$ can be calculated by (5.2.5), which is the same for all of the observers.

$$\hat{y}(t) = C(t)\hat{x}(t)$$

$$\hat{y}(t) = \begin{bmatrix} 8.7379 & 0 & 0 \\ -2.3615 & 3.2833 & 0.0579 \\ 1.7798 & -2.3684 & -0.0349 \end{bmatrix} \hat{x}(t) \qquad (5.2.5)$$

With this, the three reduced order observers and the full order observer are designed. The full order observer is not able to tolerate additive fault signals and will fail to converge in the presence of sensor faults, as can be seen in Figure 5.2.1. Figure 5.2.1 also shows that the first reduced observer does correctly match the fault free output when there is a fault on the first sensor. However, the Kalman based estimator does give the observers some degree of resistance to noise. The supervisor that selects the optimal estimator is detailed in section 5.3.

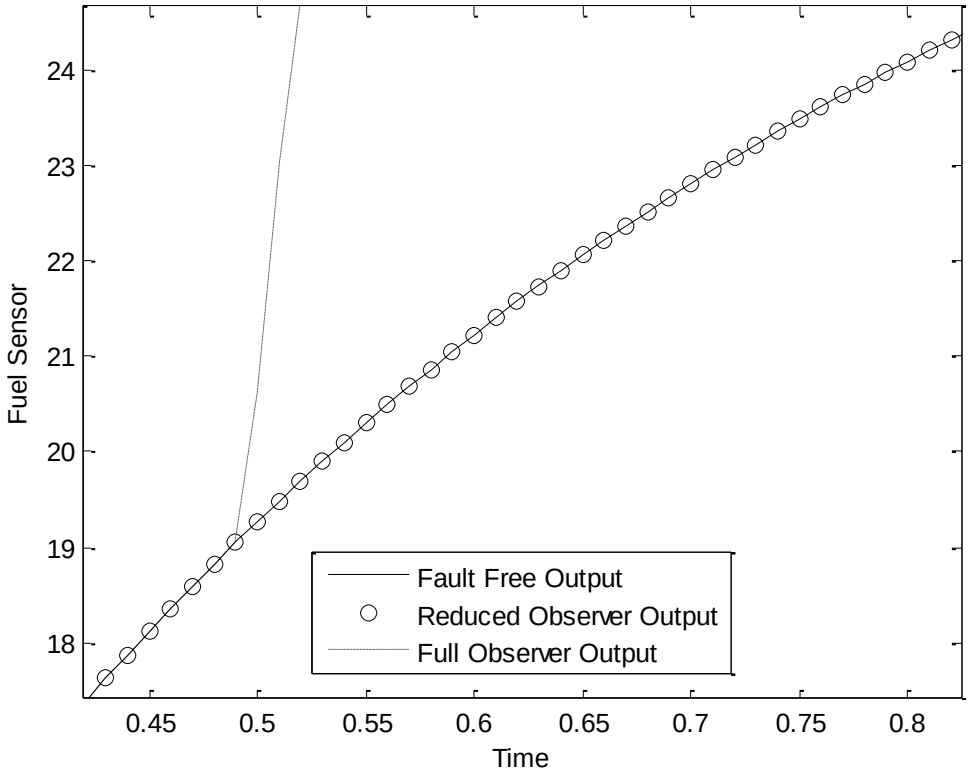

Figure 5.2.1: Comparison of the first reduced observer and the full observer when the first sensor is faulty.

5.3: Design of Supervisor, Fault Estimation, and Assembled System Results

A three input, three output, linear time-invariant model for a turbofan engine is provided in reference [9]. Specifically, the model for the fan operating under a Power Code of 30 is given. The three inputs $u(t)$ correspond to actuators controlling the fuel flow rate, the nozzle area, and the bypass duct area. The three outputs $y(t)$ correspond to sensors measuring the fan speed, core engine pressure ratio, and the overall engine pressure ratio. The plant is defined by (5.3.1) and (5.3.2) and we use the following definitions for the matrices.

$$\dot{x}(t) = Ax(t) + Bu(t) \tag{5.3.1}$$

$$y(t) = Cx(t) + Du(t) + Ff(t) \tag{5.3.2}$$

$$A = \begin{bmatrix} -2.8451 & 0.8116 & 0.0581 \\ 0.2584 & -2.6778 & 0.0584 \\ -0.0133 & -0.1305 & -0.1202 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.2686 & 0.0834 & -0.0087 \\ 0.2676 & 0.0334 & 0.0048 \\ 0.0660 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 8.7379 & 0 & 0 \\ -2.3615 & 3.2833 & 0.0579 \\ 1.7798 & -2.3684 & -0.0349 \end{bmatrix}$$

The zero matrix D is of appropriate size and F is an identity matrix of appropriate size. The fault vector $f(t)$ is varied over the different cases that are examined in this section. The system's sensor redundancy has already been tested in section 5.1 and the system is found to have rank 2 redundancy. In this case, only one sensor is allowed to fail at a time, so $k = 1$. All of the estimators are designed based on (5.3.3) and (5.3.4).

The vectors $\hat{x}(t)$ and $\hat{y}(t)$ estimate the state and output vectors respectively. The L matrix has been defined in section 5.2 to ensure error convergence of the full observer. The reduced order observer matrices L_1, L_2, and L_3 have already been defined in section 5.2.

$$\hat{\dot{x}}(t) = (A - LC)\hat{x}(t) + Ly(t) \tag{5.3.3}$$

$$\hat{y}(t) = C\hat{x}(t) \tag{5.3.4}$$

Each of the reduced order observers has been tuned to tolerate one of the three outputs suffering a sensor fault. The supervisor needs to determine which of the four observers is operating optimally. This is done by examining the minimum of the unexpected error vector.

$$e_u(o,t) = \sum_{\bar{\imath}} |\hat{y}_{o\bar{\imath}}(t) - y_{\bar{\imath}}(t)| \tag{5.3.5}$$

In (5.3.5), o represents which observer the unexpected error measurement is referencing. The unexpected error for a specific observer is the sum of the output errors on sensors that were presumed fault free, at a specific time t. As the full order observer presumes no faults occur, the set of faulty sensors i is zero, and so $\hat{y}_{\bar{\imath}}(t)$ is equal to $\hat{y}(t)$ on the full observer. For numerical convenience, the full order observer is chosen as $o = 0$, and the first reduced order observer corresponding to the fault set where the first sensor fails is chosen to be $o = 1$. The other reduced order observers are numbered in the same way. For example, $\hat{y}_{21}(5.5s)$ is the second reduced order observer's estimate of

the first sensor at time 5.5 seconds. The observer that has the lowest value of $e_u(o,t)$ at a given time t is chosen by the supervisor for its $\hat{y}(t)$ estimates. The data from that estimate is selected to represent the fault tolerant system.

Once the optimal observer has been selected, the supervisor can estimate the fault vector $f(t)$ by (2.3.1), reprinted here.

$$\hat{f}_i(t) = F_i^{-1}(y_i(t) - \hat{y}_i(t))$$

The F matrix was presumed to be an identity matrix, so its inverse is trivial to calculate. In the real world application of the system, the requirement that only one sensor fails at a time needs to be relaxed. Instead, only one sensor suffers a deterministic fault at a time, while all of them can suffer noise. The presence of noise on components that were presumed fault free means that non faulty sensors may still exhibit some minor fluctuations around zero. With these changes, the fault estimate is expanded.

$$\hat{f}(t) = y(t) - \hat{y}(t) \tag{5.3.6}$$

Equation (5.3.6) is used by the supervisor to determine the magnitude of sensor faults on all sensors, which are provided to the user in addition to the output estimates.

Now that the system is fully designed, the components are put together and simulated in Matlab. The following situations show the fault tolerant system's operation in various cases.

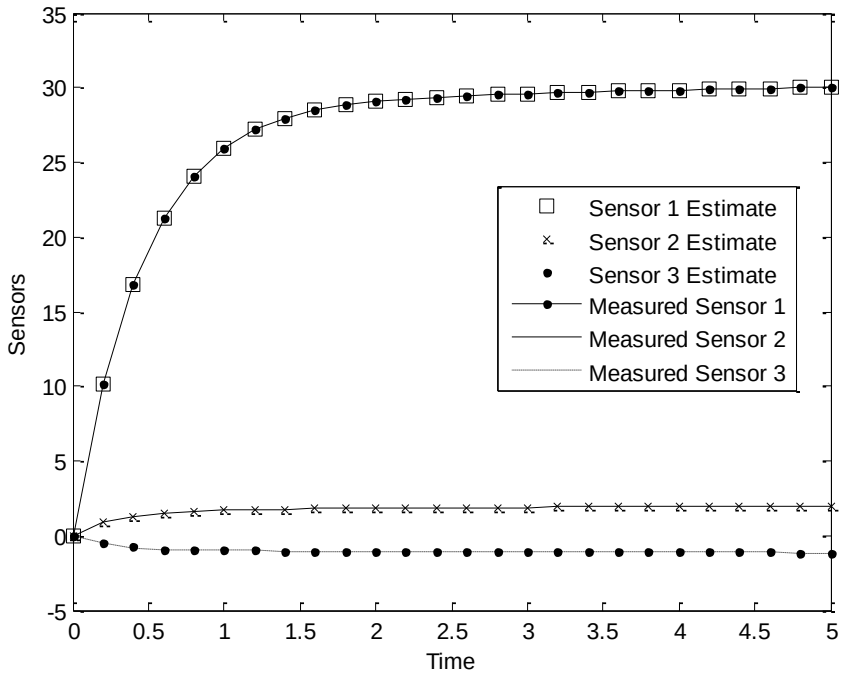

Figure 5.3.1: Sensor estimates vs. measured sensors when no faults are occurring.

Figure 5.3.1 shows that this fault tolerant method converges quickly and without introducing any new faults, when there are no faults in the system. There is no delay on measurements introduced by this technique. This means that this FTC method does not reduce the effectiveness of the system, when operating in the fault free mode.

The system's operation in the presence of a ramp fault on the first sensor is plotted in Figure 5.3.2. This type of fault is used to represent a slew error on the sensor. The first measured sensor diverges and is incorrect after the first fractions of a second. The estimates are able to correctly reconstruct all three outputs despite the failure of one of the sensors. The errors between the estimated outputs and the fault free outputs are shown in Figure 5.3.3.

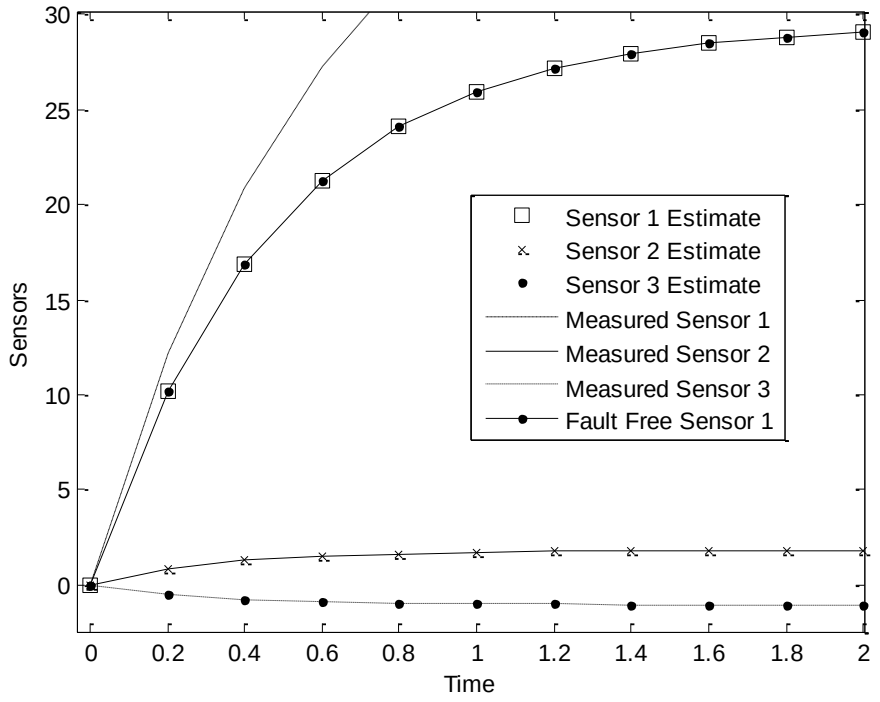

Figure 5.3.2: Sensor estimates, measured sensors, and theoretical fault free first sensor. The first sensor is suffering a ramp offset error at time zero.

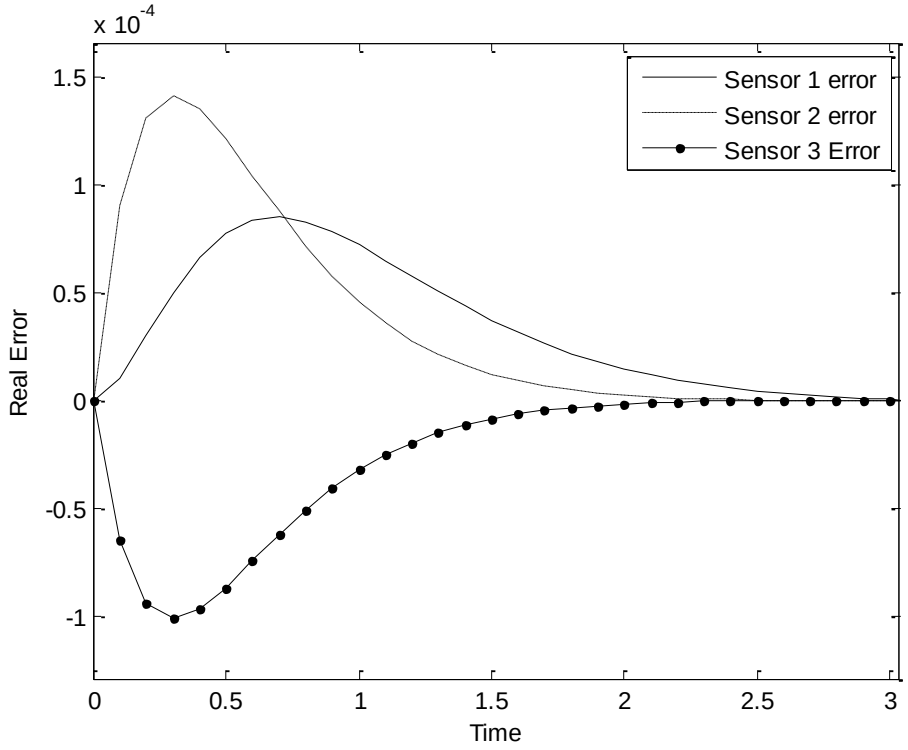

Figure 5.3.3: Error between FTC system sensor estimates and fault free sensors.

The estimator errors produced by the ramp failure are of the order of 10^{-4}, as can be seen in Figure 5.3.3. Transients cause this minor fluctuation. Because of this, the supervisor's programming must include a threshold level for faults so that it doesn't say that a fault is always occurring. In this example, 10^{-1} was used as the threshold for fault detection.

The supervisor's $e_u(o,t)$ fault detection vector is shown in Figure 5.3.4. As can be seen, the first reduced observer is the only observer expressing an $e_u(t)$ less than the 10^{-1} threshold for fault detection. Thus the supervisor chooses the first reduced observer's estimates as the optimal estimates.

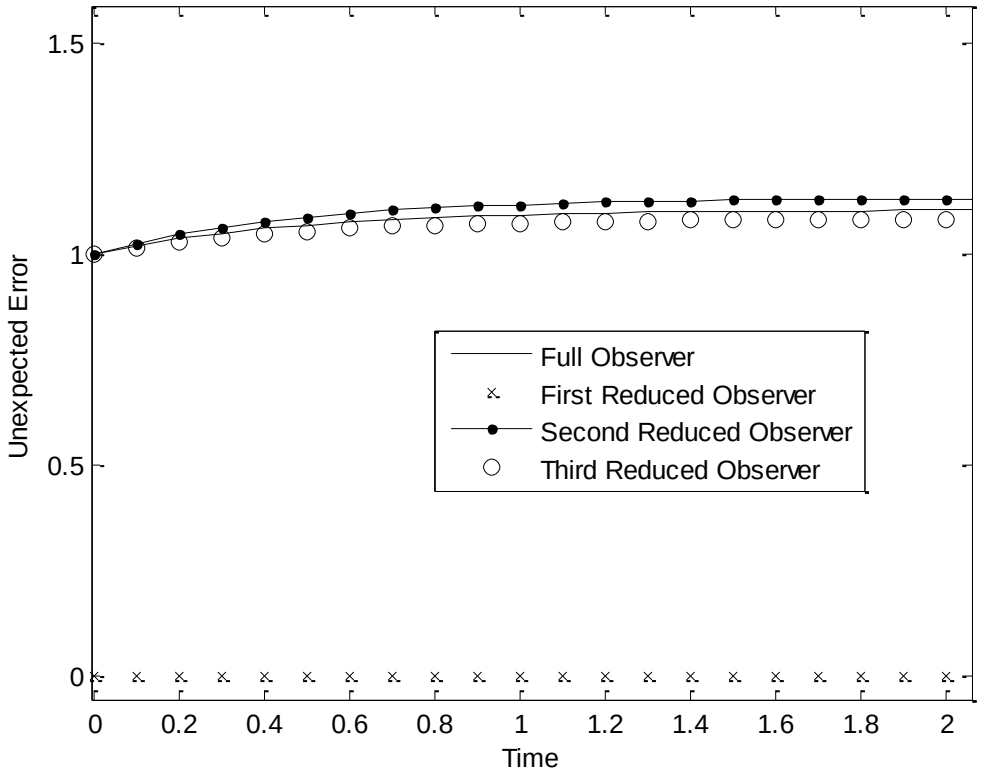

Figure 5.3.4: Unexpected error of each observer when the first sensor is faulty.

The mathematical proof of this FTC technique requires that faults only occur on k sensors or less. This is a reasonable real world expectation with respect to deterministic faults, but fails to take into account that noise will often be present in real world applications. Figure 5.3.5 shows the results of this method in the presence of significant white Gaussian noise. The estimator stays correct even in the presence of a high level of noise. This is because the Kalman observers that are used in the fault tolerant system are naturally resistant to noise. This means that the requirements of this system can be relaxed to allow noise on all sensors simultaneously.

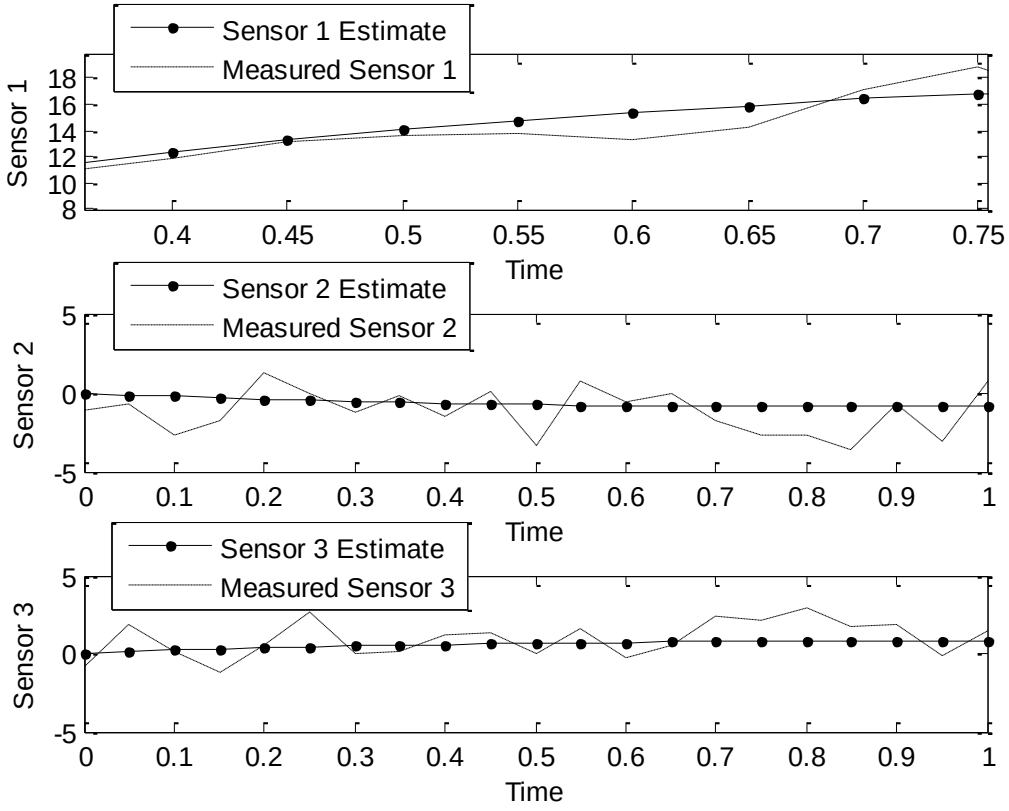

Figure 5.3.5: Sensor estimation with white Gaussian noise on all sensors.

CHAPTER 6: APPLICATION WITH SENSOR AND ACTUATOR FAULTS

6.1: Actuator Redundancy Calculation

An example of the actuator redundancy calculations is provided here. A three input, three output, linear time-invariant model for a turbofan engine is provided by Fredrick [9]. Specifically, the model for the fan operating under a Power Code of 30 is given. The three inputs $u(t)$ correspond to actuators controlling the fuel flow rate, the nozzle area, and the bypass duct area. The three outputs $y(t)$ correspond to sensors measuring the fan speed, core engine pressure ratio, and the overall engine pressure ratio. The state space equations are defined by (6.1.1) and (6.1.2).

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) \tag{6.1.1}$$

$$y(t) = C(t)x(t) + D(t)u(t) \tag{6.1.2}$$

$$A = \begin{bmatrix} -2.8451 & 0.8116 & 0.0581 \\ 0.2584 & -2.6778 & 0.0584 \\ -0.0133 & -0.1305 & -0.1202 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.2686 & 0.0834 & -0.0087 \\ 0.2676 & 0.0334 & 0.0048 \\ 0.0660 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 8.7379 & 0 & 0 \\ -2.3615 & 3.2833 & 0.0579 \\ 1.7798 & -2.3684 & -0.0349 \end{bmatrix}$$

And D is a zero matrix of appropriate size.

R_c example

There are three $R_C(z)$ values to calculate. Starting with $z = 1$, we calculate K.

$$K_0 = B_Z = \begin{bmatrix} 0 & 0.0834 & -0.0087 \\ 0 & 0.0334 & 0.0048 \\ 0 & 0 & 0 \end{bmatrix}$$

$$K_1 = -AK_0 + \dot{K}_0 = \begin{bmatrix} 0 & 0.2101 & -0.0286 \\ 0 & 0.0678 & 0.0151 \\ 0 & 0.0054 & 0.0005 \end{bmatrix}$$

$$K_2 = -AK_1 + \dot{K}_1 = \begin{bmatrix} 0 & 0.5424 & -0.9365 \\ 0 & 0.1269 & 0.0478 \\ 0 & 0.0123 & 0.0017 \end{bmatrix}$$

$$R_C(1) = rank(K) = 2$$

In the next case, $z = 2$.

$$K_0 = B_Z = \begin{bmatrix} 0.2686 & 0 & -0.0087 \\ 0.2676 & 0 & 0.0048 \\ 0.0660 & 0 & 0 \end{bmatrix}$$

$$K_1 = -AK_0 + \dot{K}_0 = \begin{bmatrix} 0.5432 & 0 & -0.0286 \\ 0.6433 & 0 & 0.0151 \\ 0.0464 & 0 & 0.0005 \end{bmatrix}$$

$$K_2 = -AK_1 + \dot{K}_1 = \begin{bmatrix} 1.0207 & 0 & -0.0937 \\ 1.5796 & 0 & 0.0478 \\ 0.0967 & 0 & 0.0017 \end{bmatrix}$$

$$R_C(2) = rank(K) = 3$$

In the third and final case, $z = 3$.

$$K_0 = B_Z = \begin{bmatrix} 0.2686 & 0.0834 & 0 \\ 0.2676 & 0.0334 & 0 \\ 0.0660 & 0 & 0 \end{bmatrix}$$

$$K_1 = -AK_0 + \dot{K}_0 = \begin{bmatrix} 0.5432 & 0.2102 & 0 \\ 0.6433 & 0.0679 & 0 \\ 0.0464 & 0.0055 & 0 \end{bmatrix}$$

$$K_2 = -AK_1 + \dot{K}_1 = \begin{bmatrix} 1.0207 & 0.5426 & 0 \\ 1.5796 & 0.1272 & 0 \\ 0.0968 & 0.0123 & 0 \end{bmatrix}$$

$$R_C(3) = rank(K) = 3$$

From this, the R_c vector is calculated to be [2,3,3]. The measure shows two pieces of information. The system is not guaranteed to controllable in the presence of any single actuator fault, as the minimum of R_c is less than n. Observation of the vector shows that the first control signal $u_1(t)$ is the critical actuator. A full failure of the actuator controlling the fuel flow rate will lead to a loss of controllability.

This means that it is not possible to guarantee reconstruction with this plant, and there will not be a way to use reduced order observers to recreate fault free actuator behavior. Most modern systems will fail this test, as few systems have any actuator redundancy due to the cost of the components. This is why this research uses ASOs to perform actuator fault tolerance, as is shown in the next section.

6.2: Airplane Dynamics and Controller Design

This section goes into an example of designing an Augmented State Observer (ASO) for actuator fault tolerance. In this section the plant is changed from the turbofan engine to a Boeing 747 jet transport airplane. The airplane's model is a fourth order linear time-invariant lateral perturbation equation. This model operates at a horizontal flight of 40,000 ft altitude and with a forward speed of 774 ft/s. The aircraft coordinate system is shown in Figure 6.2.1 [8]. This model uses a system of three hydraulic actuators on the rudder as the control surfaces for the airplane.

$$u(t) = [\delta r_1, \delta r_2, \delta r_3]^T$$

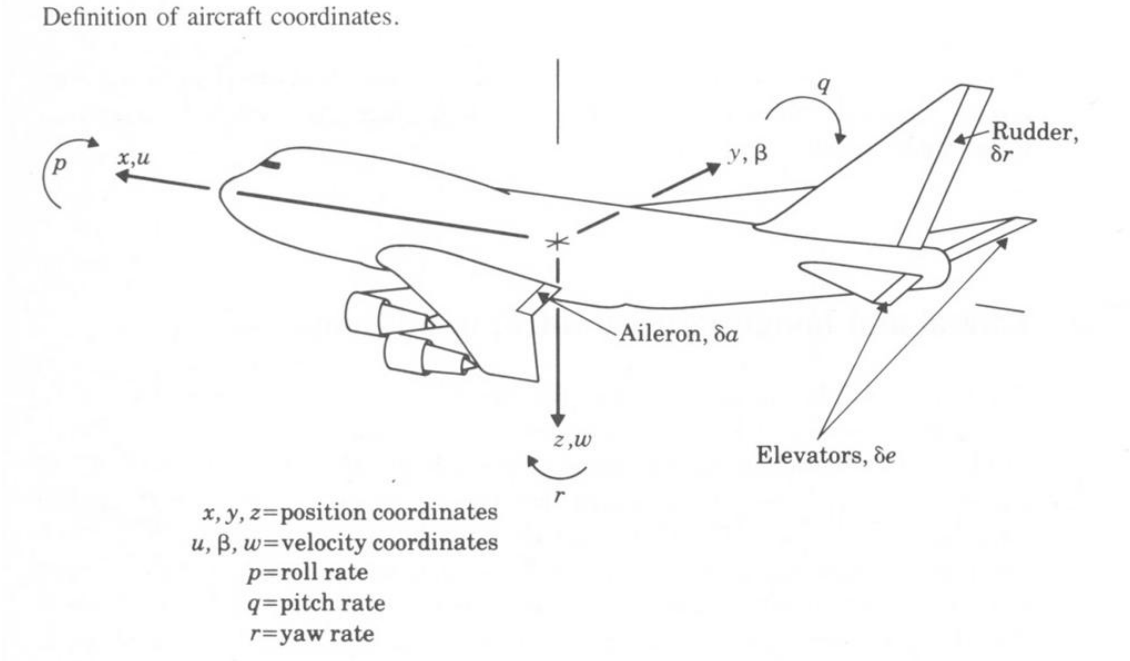

Figure 6.2.1: Boeing 747 coordinate diagram.

The plant has four internal states: the sideslip angle β, the yaw rate r, the roll rate p, and the roll angle φ. They are measured in radians, radians/second, radians/second, and radians respectively.

$$x(t) = [\beta, r, p, \varphi]^T$$

The airplane has sensors that measure each state directly to measure the sideslip angle, yaw rate, roll rate, and roll angle.

$$y(t) = [\beta, r, p, \varphi]^T$$

This model is taken from [8], [13], and [17] and is defined by equations (6.2.1) and (6.2.2) with the following parameters.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + E(t)v(t) \tag{6.2.1}$$

$$y(t) = C(t)x(t) + F(t)f(t) \tag{6.2.2}$$

$$A = \begin{bmatrix} -0.0558 & -0.9968 & 0.0802 & 0.0415 \\ 0.5980 & -0.1150 & -0.0318 & 0 \\ -3.050 & 0.3880 & -0.4650 & 0 \\ 0 & 0.0805 & 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.0073 & -0.0329 & 0.478 \\ -0.4750 & 0.4980 & 0.0245 \\ 0.1530 & -0.0033 & 2.45 \\ 0 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The airplane dynamics do not include a model of the actuator or sensor faults, so they are assumed to be one on the main diagonal and zero elsewhere. There are only three $v(t)$ fault signals because the actuators do not directly modify the roll angle φ. All four sensors can suffer faults so there are four faulty sensor signals $f(t)$.

$$E = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$$

$$F = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

First it needs to be verified that the plant has sufficient sensor redundancy for this method. This is checked with the method outlined in section 2.1. After checking, it is confirmed that the airplane has rank 1 sensor redundancy. As such, any one sensor can fail without the plant losing observability. This redundancy will not be shown here.

The system's base response is measured. The airplane is underdamped and takes around 15 minutes to settle. This performance is unacceptable. The FTC system uses a state feedback controller $u(t) = K_x\hat{x}(t) + K_v\hat{v}(t)$ with negative feedback. Eigen Value Assignment is used to design a K_x that will modify the plant so that the system dynamics are similar to a second order system with a zeta (damping factor) of 0.7 and a new settling time of around 13 seconds.

$$K_x = \begin{bmatrix} -104.466 & 12.0494 & -0.0122 & -0.0770 \\ -98.680 & 22.2793 & -0.8517 & -0.0459 \\ 5.1450 & -0.7206 & 0.1081 & 0.0848 \end{bmatrix}$$

The K_v term is designed by using eigen value assignment to ensure that the eigen values of the adaptive controller are critically damped and converge an order of magnitude faster than the plant. This has an undesired side effect because the norm of $\|BK_v\|$ is ten. This means that the actuator fault estimates produced by the observers are attenuated by a factor of ten. This is corrected by the supervisor which uses a post-amplifier to increase the observer's internal $\hat{v}(t)$ by a factor of ten to get the actual $\hat{v}(t)$.

$$K_v = \begin{bmatrix} -187.1946 & -12.1226 & 36.6433 \\ -179.1122 & 8.4798 & 34.8604 \\ 11.4474 & 0.7685 & 1.8405 \end{bmatrix}$$

6.3: Design of the Augmented State Observers

The controller that uses state and actuator fault feedback to perform eigen value assignment has been designed in section 6.2. The example continues with the Boeing 747 airplane. The airplane's model is a fourth order linear time-invariant lateral perturbation equation. This model operates at a horizontal flight of 40,000 ft altitude and with a forward speed of 774 ft/s. This model of the airplane uses a system of three hydraulic actuators on the rudder as the control surfaces for the airplane. This model is obtained from references [8], [13], and [17] and is defined by (6.3.1) and (6.3.2) with the following parameters.

$$\dot{x}(t) = A(t)x(t) + B(t)u(t) + E(t)v(t) \tag{6.3.1}$$

$$y(t) = C(t)x(t) + F(t)f(t) \tag{6.3.2}$$

$$A = \begin{bmatrix} -0.0558 & -0.9968 & 0.0802 & 0.0415 \\ 0.5980 & -0.1150 & -0.0318 & 0 \\ -3.050 & 0.3880 & -0.4650 & 0 \\ 0 & 0.0805 & 1 & 0 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.0073 & -0.0329 & 0.478 \\ -0.4750 & 0.4980 & 0.0245 \\ 0.1530 & -0.0033 & 2.45 \\ 0 & 0 & 0 \end{bmatrix}$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The FTC system uses a state feedback controller $u(t) = K_x\hat{x}(t) + K_v\hat{v}(t)$ with negative feedback. The state and actuator feedback matrices have been determined in section 6.1.

$$K_x = \begin{bmatrix} -104.466 & 12.0494 & -0.0122 & -0.0770 \\ -98.680 & 22.2793 & -0.8517 & -0.0459 \\ 5.1450 & -0.7206 & 0.1081 & 0.0848 \end{bmatrix}$$

$$K_v = \begin{bmatrix} -187.1946 & -12.1226 & 36.6433 \\ -179.1122 & 8.4798 & 34.8604 \\ 11.4474 & 0.7685 & 1.8405 \end{bmatrix}$$

The eigenvalues of the original plant and the plant with state feedback are in Table 6.3.1. All of the observers are defined by the dynamics outline in equations (4.2.1) and (4.2.2), reprinted here as (6.3.3) and (6.3.4). The L_a matrix is designed to force the ASO to converge faster than the original system, by eigen value assignment. The eigen values are placed at least one order of magnitude to the left of the plant's dominant eigen values. The ASO's eigenvalues are detailed in Table 6.3.1. The ASO definitions are given below.

$$\dot{\hat{x}}_a = (A_a - L_a C_a)\hat{x}_a + B_a r + L_a y \tag{6.3.3}$$

$$\begin{bmatrix} \hat{y} \\ \hat{x} \\ \hat{v} \end{bmatrix} = \begin{bmatrix} C_a \\ I \end{bmatrix} \hat{x}_a \tag{6.3.4}$$

$$\hat{x}_a = [\beta \quad r \quad p \quad \varphi \quad \hat{v}_1 \quad \hat{v}_2 \quad \hat{v}_3]^T$$

$$\hat{y} = [\beta \quad r \quad p \quad \varphi \quad \hat{v}_1 \quad \hat{v}_2 \quad \hat{v}_3]^T$$

$$A_a = \begin{bmatrix} A - BK_x & -BK_v \\ 0 & 0 \end{bmatrix}$$

$$A_a = \begin{bmatrix} -5.0001 & -0.0072 & 0.0006 & 0 & -10 & 0 & 0 \\ -0.0072 & -5.4690 & 0.3839 & -0.0158 & 0 & -10 & 0 \\ 0.0006 & 0.3839 & -0.7309 & -0.1962 & 0 & 0 & -10 \\ 0 & 0.0805 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$B_a = \begin{bmatrix} B \\ 0 \end{bmatrix}$$

$$C_a = [C \quad 0]$$

$$L_a = \begin{bmatrix} L_x \\ L_v \end{bmatrix}$$

$$L_a = \begin{bmatrix} 22.50 & -0.0072 & 0.0006 & 0 \\ -0.0072 & 22.031 & 0.3839 & -0.0158 \\ 0.0006 & 0.3839 & 26.769 & -0.1962 \\ 0 & 0.0805 & 1 & 27.50 \\ 11 & 0 & 0 & 0 \\ 0 & 11 & 0 & 0 \\ 0 & 0 & 11 & 0 \end{bmatrix}$$

Table 6.3.1: Eigenvalues of the plant and observers

State Space Model	Eigenvalues of Matrix A			
Airplane without feedback	-0.0073	-0.0329 ± 0.9467i	-0.5627	
Airplane with feedback	-0.35± 0.274i	-5	-5.5	
Full Augmented Observer	-4.858 (x3)	-22.642 (x3)	-27.5	
1st Reduced Observer	-4.858 (x2)	-5	-22.642 (x2)	-27.5
2nd Reduced Observer	-4.858 (x2)	-5.469	-22.642 (x2)	-27.5
3rd Reduced Observer	-0.731 (x2)	-4.858 (x2)	-22.642	-27.5
4th Reduced Observer	-4.858 (x3)	-22.642 (x3)		

The full order ASO design is complete. Section 6.2 has demonstrated that there is at least rank 1 sensor redundancy in this system. As such, the rank of redundancy k is set to 1, allowing no more than one sensor fault at a time. As one sensor is presumed to fail at any time, four reduced order observers are required. Each reduced order observer is designed to presume a specific sensor is faulty. The r-ASOs use the same matrix

definitions as the ASO, with the exception of the L_a matrix. In the first r-ASO, the L_a matrix is modified to ignore the first sensor $y_1(t)$ so that it is not used by the r-ASO. This is obtained by setting the first column of L_a to zero.

$$L_{a1} = \begin{bmatrix} 0 & -0.0072 & 0.0006 & 0 \\ 0 & 22.031 & 0.3839 & -0.0158 \\ 0 & 0.3839 & 26.769 & -0.1962 \\ 0 & 0.0805 & 1 & 27.50 \\ 0 & 0 & 0 & 0 \\ 0 & 11 & 0 & 0 \\ 0 & 0 & 11 & 0 \end{bmatrix}$$

This process is repeated for the other three r-ASOs. All of them are built by (6.3.3) and have a different column of L_a set to zero. The other three L_a matrices are not printed here.

As can be seen in Table 6.3.1, there is a side effect of reducing the order of the observers. The reduced order observers are not able to place the eigen values of the observers optimally. On the 1st, 2nd, and 4th r-ASO, the eigenvalues of the observers are at least one order of magnitude to the left of the dominant eigenvalues of the airplane with feedback and all are overdamped as was intended. However, the 3rd r-ASO has two eigen values at -0.731 which are close to the plant's dominant pair of eigen values at $-0.35 \pm 0.274i$. The 3rd r-ASO's eigenvalues are to the left of the plant so it will converge, but it will not converge quickly enough that the dynamics of the airplane can be fully ignored. This may cause the third r-ASO to experience some undesired transient behaviors.

The supervisor is designed to measure the unexpected error of each estimator and select the ASO that has the lowest $e_u(t)$. From that ASO, the supervisor collects the

estimate of the states $\hat{x}(t)$, the outputs $\hat{y}(t)$, and the estimate of the actuator errors $\hat{v}(t)$. The supervisor then calculates an estimate of the sensor errors $\hat{f}(t)$ by equation (2.3.1). The next section will assemble this fault tolerant system and examine how it operates in the presence of actuator and sensor faults.

6.4: Assembled Augmented State Observer Fault Tolerant System

Sections 6.2 and 6.3 have produced the ASOs and the supervisor for the Boeing 747 model and can handle both sensor and actuator faults. The system is assembled and simulated in Matlab. In all the simulations, the airplane is excited by a five second step input to observe the model's response. The behavior of the airplane without a controller is shown in Figure 6.4.1. The airplane is stable, but it has unsatisfactory performance. The settling time is around 2 minutes for the faster states, and around 15 minutes for the roll angle φ and the yaw rate r. Figure 6.4.2 shows the improved airplane dynamics with the addition of state feedback. The controller has modified the airplane design so that it settles in 15 seconds and overshoot has been reduced to 4.3%.

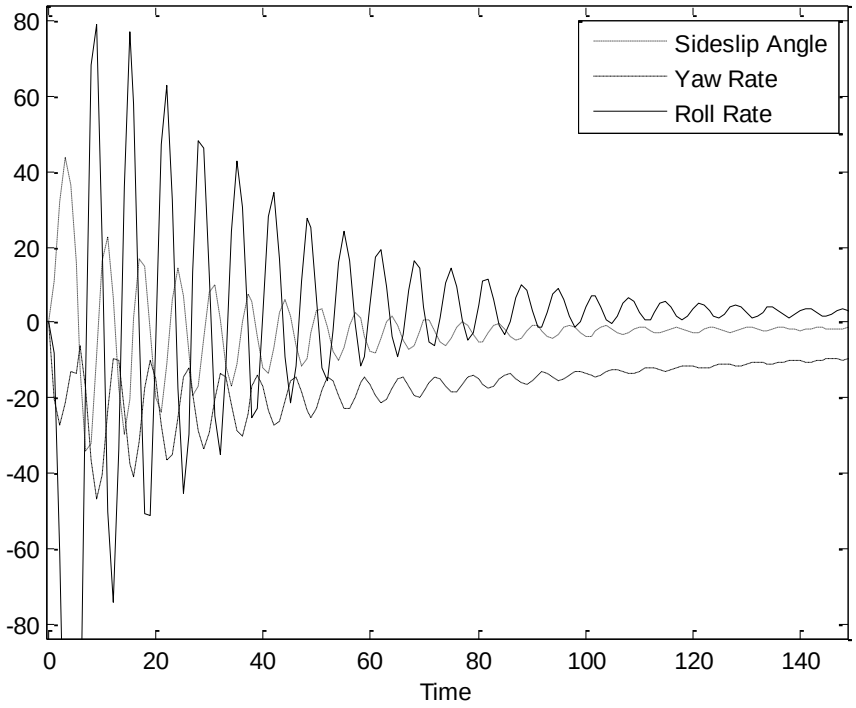

Figure 6.4.1: Uncontrolled airplane response to a five second step input. Roll angle is not shown due to scale.

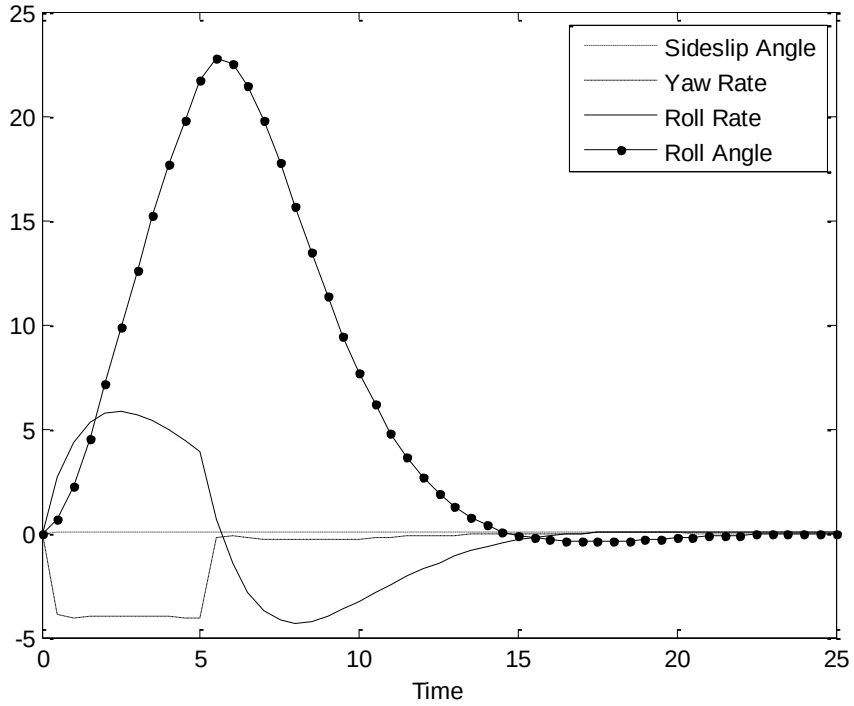

Figure 6.4.2: Controlled airplane response to a five second step input.

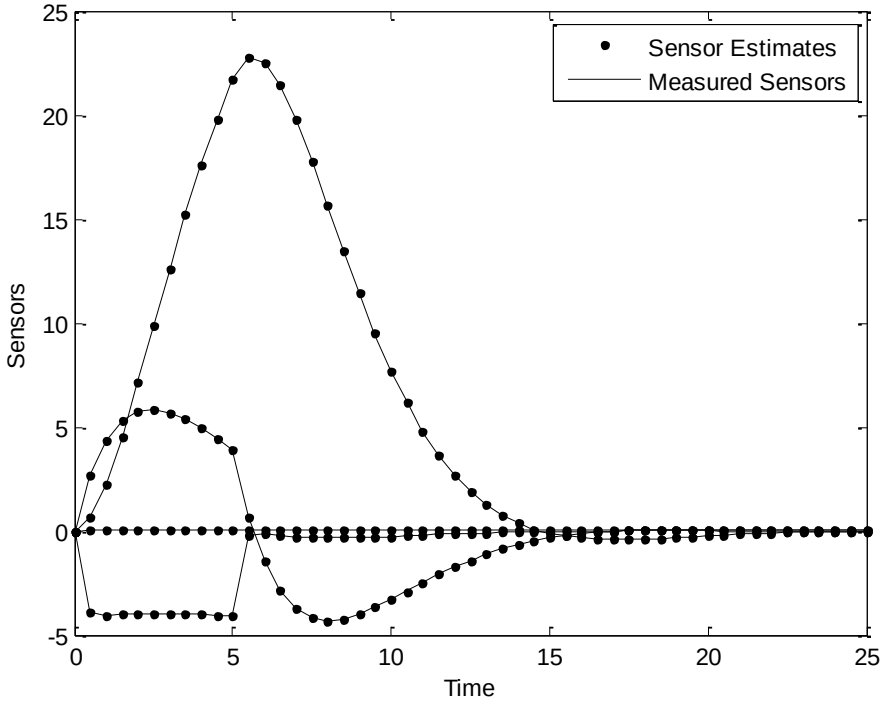

Figure 6.4.3: Fault tolerant system tracking the airplane outputs when there are no faults in the system.

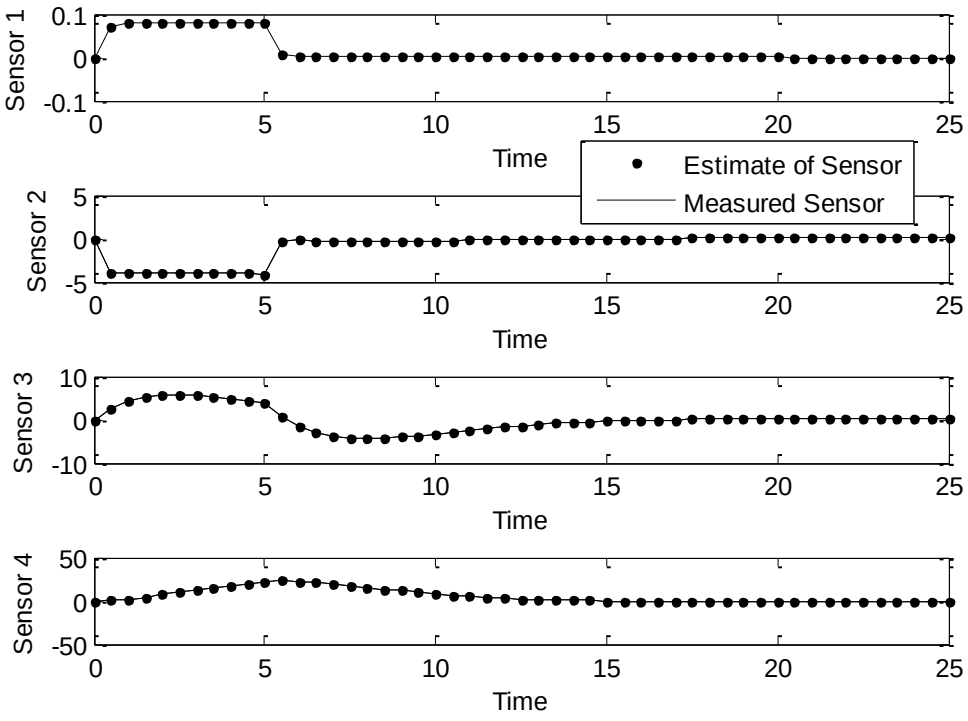

Figure 6.4.4: Fault tolerant system tracking the airplane outputs when there are no faults in the system. Plots split to better see each sensor's dynamics.

The FTC system is shown to converge to the airplane dynamics in Figure 6.4.3. In this plot, there are no faults in the actuators or sensors. This image is split in Figure 6.4.4 to better show the tracking of the system to each individual output. As can be seen, the system properly matches the airplane's dynamics without introducing delay or errors.

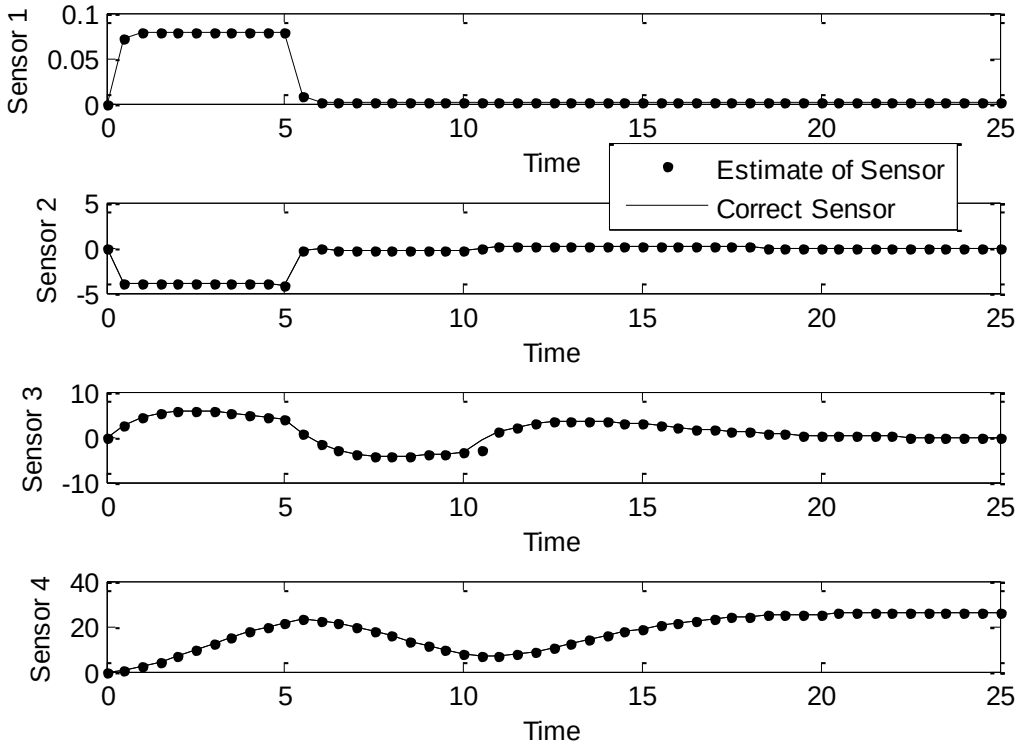

Figure 6.4.5: Fault tolerant system tracking the airplane. An actuator fault occurs at ten seconds.

Next, the actuators suffer an offset fault at 10 seconds. In Figure 6.4.5 it can be seen that the system tracks the airplane's new dynamics. The speed of convergence cannot be easily seen in Figure 6.4.5. In Figure 6.4.6 an actuator fault occurs after 1 second and the image is enhanced to better see the rate of convergence of $\hat{v}(t)$. As can be seen, the actuator fault estimates settle in around two seconds.

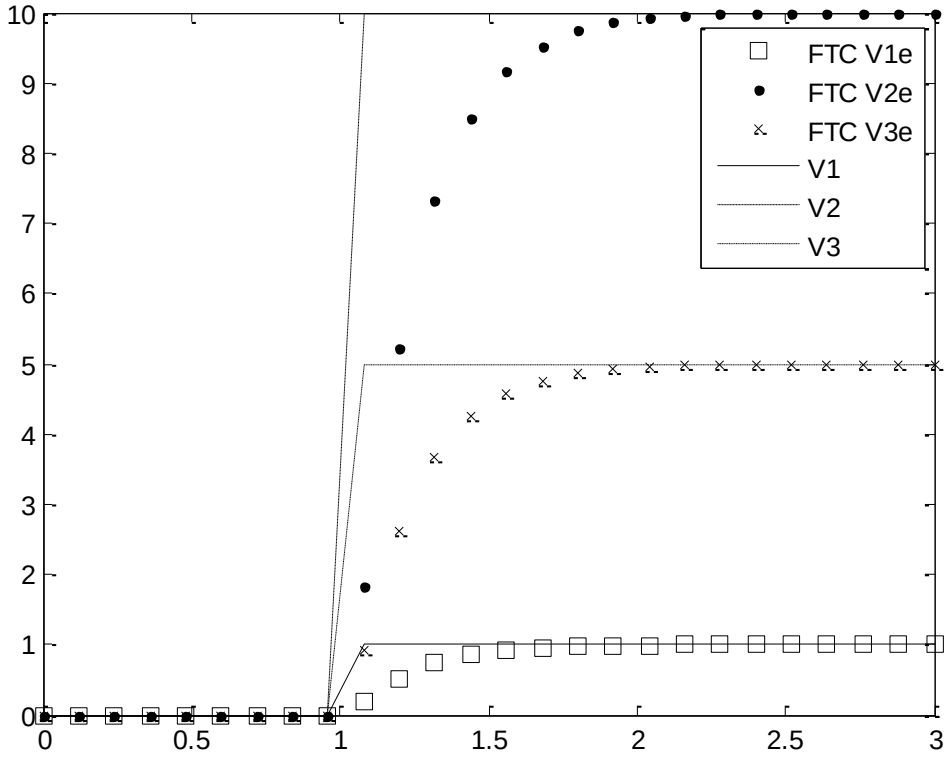

Figure 6.4.6: Comparison of the actuator faults and their estimates. Actuator faults occur after one second.

The system's response to sensor failures is tested by setting a fault on the first sensor. The sensor measuring the sideslip angle β suffers an offset error at 3 seconds. Figure 6.4.7 looks at the effect of the sensor error on the system's estimates. The plot shows that the fault tolerant system fully rejects the effects of the faulty sensor.

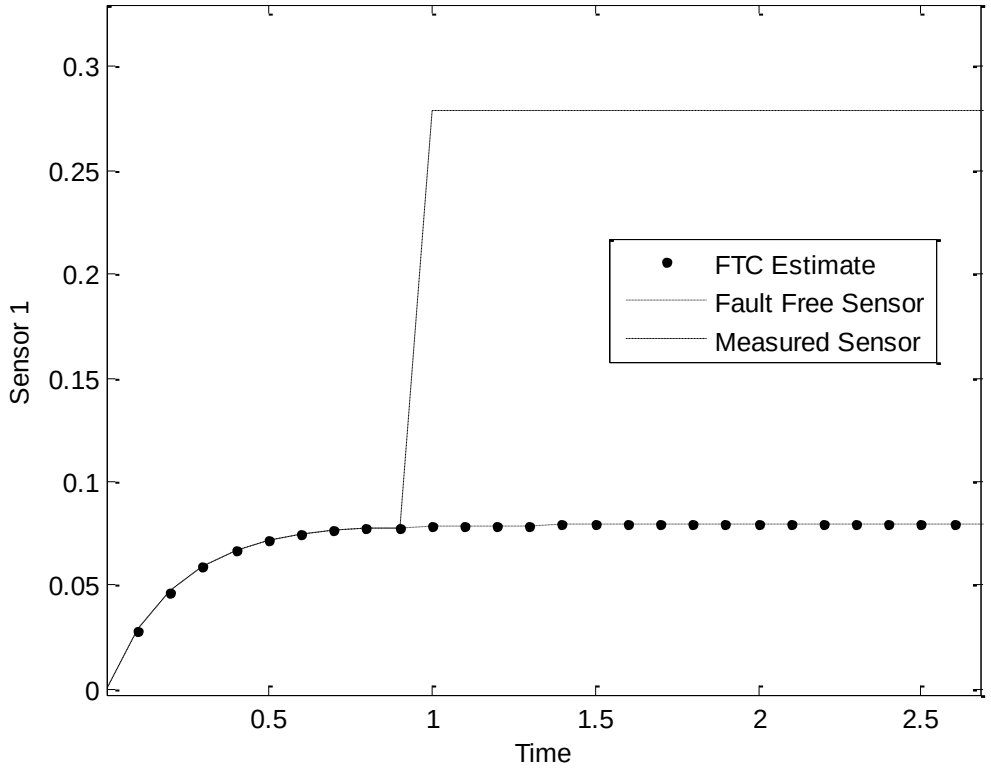

Figure 6.4.7: System's estimate of the airplane's sideslip angle. The first sensor suffers an offset fault after three seconds.

In Figure 6.4.8, the airplane is subjected to both actuator and sensor faults. The second sensor measures the yaw rate r and suffers an offset error at five seconds. At the same time, the actuator dynamics suffer a fault. Despite the presence of both sensor and actuator faults, the fault tolerant system properly tracks the airplane's response and rejects the faulty sensor data.

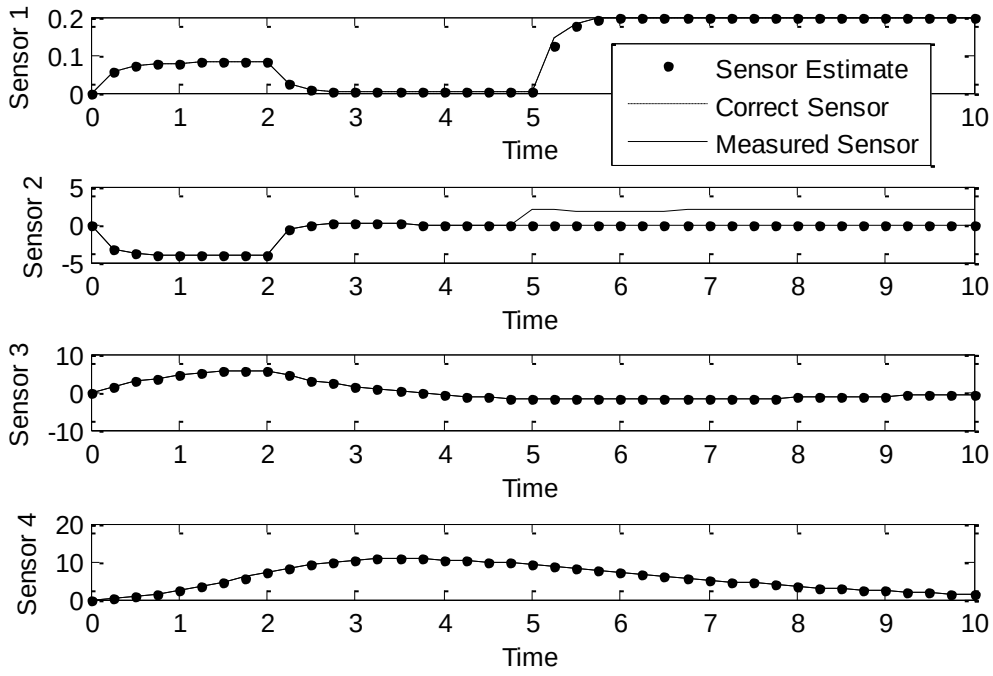

Figure 6.4.8: Estimation of all four sensors. An actuator fault occurs at five seconds. The second sensor suffers an offset fault at five seconds.

CHAPTER 7: CONCLUSIONS AND FUTURE WORK

7.1: Conclusions

In this research, a novel method to calculate the level of available sensor redundancy that aids in fault tolerant design in an innovative way is defined. The measures R_o and RR_o provide designers with detailed information regarding the effect of faults upon the observability of a plant. At the broadest view, they can be used to determine the rank of sensor redundancy available in a system. With a focused view, critical components can be identified as well as fault sets that result in a loss of observability. The focused view can point out which components of the overall system should have additional sensors, to improve redundancy. This provides designers with measures that are useful in the design process of FTC systems that take advantage of sensor redundancy.

All of this information can also be collected about the actuators by using the innovative technique to measure actuator redundancy. The vectors R_C and RR_C can be examined to understand how actuator failures interact with the plant. Critical actuators can be identified, as well as those actuators that can fail without causing the system to lose controllability. As few systems have sufficient actuator redundancy, thus the study helps a designer to add additional redundancy sensors to make it fault tolerant in terms of sensor failure.

This research further shows as to how reduced order observers can be used to isolate sensor faults. Faulty signals can be identified and isolated by combining a full order observer with a bank of reduced order observers. This work also shows that these observers can remove faults without needing to identify a model of the faulty signal. This system of reduced order observers can estimate sensor fault signals if given a model of their behavior. This technique of creating a bank of reduced order observers is shown to be applicable in both the linear time-invariant and linear time-variant cases.

This research also developed an innovating technique of designing a supervisor that selects the best estimate of the fault free outputs from the bank of observers. This completes the design of a fault tolerant control system that is able to compensate for sensor faults by using the sensor redundancy that is available in a system. By starting with a measure of the redundancy, an exact count of the number of sensor faults the system can tolerate is determinable.

This work has developed a unique method to create a system that can tolerate both actuator and sensor failures. This is achieved by replacing the Kalman observer with an Augmented State Observer and incorporating an adaptive controller. This fault tolerant method estimates the states and the actuator faults in such a way that the errors are bounded and convergent. A method to quickly determine controller and observer gains is explained to speed up the design process. The ASO design can be applied to both time-variant and time-invariant systems.

One weakness of the technique is that multiple observers are used in the estimation process which will result in estimation delay. However, observer gain matrices are correlated to each other in order to speed up the design process. Due to the

limitations of hardware space and cost, this method is best implemented with software and thus would be classified as a design of an intelligent control system. Systems that have a large set of sensors and can tolerate multiple sensors failing simultaneously will lead to significant computational demands which can be difficult to satisfy and could result into additional time delays when this system is implemented in real time.

7.2: Future Work

This research develops two complete methods in details of applying the reduced order observer technique to tolerate sensor and/or actuator faults. This fault tolerant method is adaptable to many different plants undergoing different types of sensor and actuator faults. Future work can extend this technique to work with other types of observers. Unknown Input Observers, Extended Kalman Observers, and Sliding Mode Observers can all be incorporated into this technique. The method can be modified to allow for other types of plant component failure, such as model error in the plant itself.

This work focused on the area of linear systems. Research indicates that this method can tolerate a linear system suffering nonlinear sensor faults as well, but it has not been proven. Further research could extend the reduced observer technique to nonlinear systems. Future work could also look into ensuring that systems that perform real-time piecewise linearization of a nonlinear system maintain stability when used with this method.

This research found that there were few applications where there would be enough actuator redundancy to design reduced order actuator based designs. However, there are systems that do have sufficient actuator redundancy for reconstruction. Future work can examine the idea of building a bank of observers that removes non-functioning actuators. This would be useful in systems where each individual actuator could be taken offline independently of the operations of the plant. This could also be extended to partial faults of the actuators, by using the observer to localize the actuator faults and an adaptive controller to tolerate the faults.

It would be advantageous to designers to create a system that limits the loss of performance caused by reducing the observers and proves that stability cannot be lost. This would allow the fault tolerant system to use online observers. It would also open up the possibility for this technique to be coupled with fault tolerant methods that do not require a model of the plant.

In higher order systems with a large level of redundancy, the number of reduced order observers can get too large for a computer to handle in real time. An idea to counteract this is to dynamically change the set of active observers. The system starts with a set of all observers that allow a single fault. When a fault is detected and isolated, the set of active observers is dynamically changed to the set of observers that include the isolated fault and presume one more (or one less) can occur. This way, the identified faulty sensors can be used to determine a smaller range of observers that need to be computed at a given time. This would significantly reduce the set of observers that have to be calculated, and thus the computational requirements would drop.

REFERENCES

[1] Puya Afshar and Hong Wang, Model-based sensor fault diagnosis in general stochastic systems using LMI techniques, Proceedings of the UKACC Control 2008 Conference, Manchester, UK, September 2-4, 2008

[2] Jovan D. Boskovic and Raman K. Mehra, A multiple model-based decentralized system for accommodation of failures in second-order flight control actuators, Proceedings of the 2006 American Control Conference, pp 4436-4441, Minneapolis, Minnesota, USA, June 14-16, 2006

[3] Xuejing Cai and Fen Wu, A robust fault tolerant control approach for LTI systems with actuator and sensor faults, 2009 Chinese Control and Decision Conference, pp 890-895, Raleigh, NC, 2009

[4] Shuhao Chen and Gang Tao, On matching conditions for adaptive state tracking control of systems with actuator failures, Proceedings of the 40th IEEE Conference on Decision and Control, pp1479-1483, Orlando, Florida, December 2001

[5] F.N. Chowdhury and W. Ch, A dual-loop scheme for fault-tolerance and early fault detection, IET Control Theory Appl., 1, (4), pp. 925–932, 2007

[6] John Doyle, Keith Glover, and Pramod Khargonekar, State-space solutions to standard H2 and Hinf control problems, IEEE Transactions on Automatic Control, Vol 34, No. 8, August, 1989

[7] Ricardo Dunias and S. Joe Qin, Joint diagnosis of process and sensor faults using principal component analysis, Control Engineering Practice 6, pp 457-469, 1998

[8] Gene F. Franklin, J.D. Powell, and A. Emami-Naeini. Feedback Control of Dynamic Systems, Addison-Wesley Publishing Company, Reading, Massachusetts, 1991.

[9] Dean K. Frederick, Sanjay Garg, and Shrider Adibhatla, Turbofan engine control design using robust multivariable control technologies, IEEE Transactions on Control Systems Technology, vol. 8, no. 6, pp961-970, November, 2000

[10] Zhiqiang Gao and Panos J. Antsaklis, On the stability of the pseudo-inverse method for reconfigurable control systems, International Journal of Control, pp 333-337, Notre Dame, IN, 1989

[11] Zhiqiang Gao, Scaling and bandwidth-parameterization based controller tuning, Proceedings of the 2003 American Control Conference, pp 4989-4996, Denver, Colorado, June 4-6, 2003

[12] Zhiqiang Gao, Active disturbance rejection control: a paradigm shift in feedback control system design, Proceedings of the 2006 American Control Conference, pp 2399-2405, Minneapolis, Minnesota, USA, June 14-16, 2006

[13] R. K. Heffley and W. F. Jewell. Aircraft Handling Qualities, Tech. Rept. 1004-1, System Technologies Inc., Hawthorne, California, May, 1972

[14] Bin Jiang and Fahmida Chowdhury, Observer-based fault diagnosis for a class of nonlinear systems, Proceeding of the 2004 American Control Conference, pp5671-5675, Boston, Massachusetts, June 30-July 2, 2004

[15] Xiao-Zheng Jin and Guang-Hong Yang, Robust fault-tolerant control via linear fractional transformations, 16th IEEE International Conference on Control Applications part of IEEE Multi-Conference on Systems and Control, pp 640-645, Singapore, October 1-3, 2007

[16] Rongfu Luo, Manish Misra, and David M. Himmelblau, Sensor fault detection via multiscale analysis and dynamic PCA, Ind. Eng. Chem. Res. 1999, 38, pp1489-1495, 1999

[17] R. J. Patton and S. Klinkhieo, Actuator fault estimation and compensation based on an augmented state observer approach, Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, pp 8482-8487, Shanghai, P.R. China, December 16-18, 2009

[18] Wilson Rugh, Linear System Theory, Prentice Hall Inc, Upper Saddle River, NJ, 1996

[19] R. Sharma and M. Aldeen, Design of integral sliding mode observers with application to fault and unknown input reconstruction, Joint 48th IEEE Conference on Decision and Control and 28th Chinese Control Conference, pp 6958-6953, Shanghai, P.R. China, December 16-18, 2009

[20] Silvio Simani, Cesare Fantuzzi, and Sergio Beghelli, Improved observer for sensor fault diagnosis of a power plant, Proceedings of the 7th Mediterranean Conference on Control and Automation (MED99), pp 826-834, Haifa, Israel, June 28-30, 1999

[21] Feng Tao, Stochastic fault tolerant control analysis and synthesis, Thesis for University of Alberta, Edmonton, Alberta, 2007

[22] Gang Tao and Shuhao Che, An adaptive control scheme for systems with unknown actuator failures, Proceedings of the American Control Conference, pp1115-1120, Arlington, VA, June 25-27, 2001

[23] Gang Tao and Shuhao Che, An adaptive actuator failure compensation controller using output feedback, Proceedings of the American Control Conference, pp 3085 - 3090, Arlington, VA, June 25-27, 2001

[24] Xu Tao and Wang Qi, Application of MSPCA to sensor fault diagnosis, ACTA Automatica Sinica, pp 417-421, No. 3, Harbin Institute of Technology, Harbin, May, 2000

[25] Chengwei Tian, Changfu Zong, Lei He, and Xiang Wang, Fault tolerant control method for steer-by-wire system, Proceedings of the 2009 IEEE International Conference on Mechatronics and Automation, pp 291-295, August 9-12, Changchun, China

[26] K. C. Veluvolu and Y. C. Soh, Nonlinear sliding mode observers for state and unknown input estimations, Proceedings of the 46th IEEE Conference on Decision and Control, pp 4347-4352, New Orleans, LA, USA, Dec. 12-14, 2007

[27] Tyrone L. Vincent and Pramod P. Khargonekar, A class of nonlinear filtering problems arising from drifting sensor gains, IEEE Transactions on Automatic Control, pp509-520, vol. 44, no. 3, March, 1999

[28] Benjamin Walker, Yogendra Kakad, and Bharat Joshi, Application of redundancy in observability and controllability towards fault tolerant system design, Proceedings of 44th IEEE Southeastern Symposium on System Theory, 2012

[29] Benjamin Walker, Yogendra Kakad, and Bharat Joshi, Use of reduced order observers in reconstruction of faulty sensors, Proceedings of IEEE Southeast Con 2012, 2012

[30] Hong Wang, F Zhen, J. Juangs, and Steve Daley, On the use of adaptive updating rules for actuator and sensor fault diagnosis, Automatica, Vol. 33, No. 2, pp. 217-225, 1997

[31] Shengwei Wang and Fu Xiao, AHU sensor fault diagnosis using principal component analysis method, Energy and Buildings 36, pp 147-160, 2004

[32] Tsang-Yi Wang, Li-Yuan Chang, Dyi-Rong Duh, and Jeng-Yang Wu, Distributed fault-tolerant detection via sensor fault detection in sensor networks, FUSION - 10th International Conference on Information Fusion, 2007

[33] C.L. Wei, J.S.H. Tsai, S.M. Guo2, and L.S. Shieh, Universal predictive Kalman filter-based fault estimator and tracker for sampled-data nonlinear time-varying systems, IET Control Theory Appl., Vol. 5, Iss. 1, pp. 203–220, 2011

[34] Dapeng Ye and Paul P. Lin, Fault detection with little knowledge of system model, 2008 IEEE International Conference on Systems, Man and Cybernetics, pp1972-1977, 2008

[35] Liu Yutian, Li Changgang, and Hu Junjie, Fault diagnosis and fault tolerant control of nonlinear systems, Proceedings of the 2010 IEEE International Conference on Automation and Logistics, pp 447-450, Hong Kong and Macau, August 16-20, 2010

[36] Ke Zhang, Bin Jiang, and Vincent Cocquempot, Adaptive observer-based fast fault estimation, International Journal of Control, Automation, and Systems, vol. 6, no. 3, pp. 320-326, June, 2008

[37] Wang Zhi, Unknown input observer based fault class isolation and estimation, Proceedings of the 29th Chinese Control Conference, pp 3963-3986, Beijing, China, July 29-31, 2010

[38] D. H. Zhou and P. M. Frank, Fault diagnostics and fault tolerant control, IEEE Transactions on aerospace and electronic systems, vol. 34, no. 2, pp 420-427, April, 1998

[39] Kemin Zhou, John Doyle, and Keith Glover, Robust and Optimal Control, Prentice Hall, Upper Saddle River, New Jersey, 1996

[40] Kemin Zhou, Zhang Ren, and Wei Wang, On the design of unknown input observers and fault detection filters, Proceedings of the 6th World Congress on Intelligent Control and Automation, pp5638-5642, Dalian, China, June 21-23, 2006