# Experimental Control and Model Validation: A Helicopter Case Study

Thesis by

John C. Morris

In Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

California Institute of Technology
Pasadena, California

1996

(Submitted September 21, 1995)

©1996
John C. Morris
All rights reserved

# Acknowledgements

My experience at Caltech has been greatly enhanced through my involvement with many different people. John Doyle and Richard Murray have both helped me tremendously through many valuable discussions as well as providing support for my research program. Both possess great insight and a deep understanding of the relevant issues in control theory; it was through their guidance and willingness to educate me that I completed this dissertation. Drs. Joel Burdick, John Doyle, Rodney Goodman, Richard Murray, and Pietro Perona kindly agreed to review my dissertation and serve on my defense committee; for this I am truly indebted.

I have been fortunate to have met and been able to collaborate with many remarkable people, including Michiel van Nieuwstadt, Matt Newlin, Pete Young, Bobby Bodenheimer, Pascale Bendotti, and Pierre Apkarian. I am thankful for their help with my research. While I was writing this dissertation, Bobby Bodenheimer, Matt Newlin, Pete Young, Michiel van Nieuwstadt, and Jürgen Eich provided invaluable advice, guidance, and helpful comments.

In the course of building the experimental helicopter platform there have been several colleagues who helped tremendously. Michiel van Nieuwstadt provided assistance every step of the way with all aspects of building and maintaining the helicopter experiment. His focus and drive kept the helicopter experiment going through several frustrating stages of development. Brian Clendenin, Jim Ostrowski, Brett Slatkin, and Shantanu Ambastha all helped put together the original helicopter experiment.

I originally became interested in working on helicopters as a result of a summer research position at the NASA Ames Research Center. I am grateful to Victor Cheng and Dallas Denery for providing me with this opportunity. This led to a NASA research fellowship which funded most of my work at Caltech.

While at Caltech I was fortunate to have been able to work at the NASA Jet Propulsion Laboratory on "real" engineering problems in flight computing. Working with Leon, Mike, John, Dan, Jerry, Gary, and Loring was great fun, and I hope to have the opportunity to collaborate with them in the future.

My friendships helped me retain my sanity and happiness; especially the Friday night barbecues and pool with Bobby, Matt, Pete and Pat, Wayne, Gabriel, Carolyn, and Jorge, the fantastic gourmet dinners with Öjvind and Jack, and the volleyball class where I met Josée. Late nighters with Carolyn, Gabriel, and Wayne got me through my first year. Bobby and I have been through a lot together; our friendship will stand the test of time. My friendship with Pete and Pat has been a source of happiness: whether it was shooting pool, guzzling beer, or plodding through Death Valley with Pete. Matt is one of those unique individuals some people never have the good fortune to meet; I count him as a true friend. I had many great days with Diego and Elvira and hope to someday learn how to make good Sangria. I've enjoyed the dinners and evenings spent playing cards with Mike and Donna. I am thankful to have such great friends.

I wish to express my thanks and appreciation for the support provided by Josée, my parents Chris and John, my sister Suzy, Scott, Nikolas, Chantal, Hughes, Lucie, and Denis. Without them I would never have been able to finish.

Most importantly, I want to thank Josée. It was our love and her understanding that brought me through everything.

# Abstract

Robust control has not been used as widely as it could because modelling tools have not advanced as far as analysis and synthesis tools. This becomes readily apparent when applying robust control theory to real problems. With this in mind, an experimental platform was designed and built to study the application of robust control. This platform consists of a real-time computer and a radio-controlled model helicopter mounted on a six degree-of-freedom stand. Experimental systems provide the opportunity not only to verify the applicability of new control theory but also to highlight potential deficiencies.

Traditional system identification and control techniques were used to construct hover controllers for the model helicopter. These techniques are not suitable for the construction of robust models for a system of this complexity. In particular, there was no systematic way to augment nominal identified models with uncertainty suitable for the construction of robust controllers.

To address this issue, frequency-domain model validation algorithms and software were developed. These algorithms provide a methodology for verifying the applicability and consistency between experimental data and robust models. Additionally, they provide a method whereby nominal model parameters can be tuned in a robust setting. This is the first set of software tools which provide this capability for general linear uncertain systems.

Using these new software tools, a systematic design process was developed which incorporated frequency-domain model validation analysis, $\mu$-analysis and $\mu$-synthesis, simulation, and implementation. This design process proved to be a valuable new tool for constructing robust models and designing robust control systems. In particular, by applying this design process to the helicopter, the size of uncertainty in the robust model was substantially reduced without sacrificing the ability of the model to "cover" experimental data and the first controller implemented performed well. This was strikingly different from the results obtained when using standard robust control techniques, where several controllers destabilized the helicopter when implemented, even though they performed well under simulation.

The model validation software and design process provide a consistent methodology and systematic framework which connects system identification, the construction of robust models, and controller synthesis with experimental data. For the first time the control engineer can compute measures on the validity of a robust model, with respect to all observed data on the actual physical system, which are directly related to the robustness measures resulting from $\mu$-analysis and $\mu$-synthesis.

# Table of Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

"Where shall I begin, please your Majesty?" he asked. "Begin at the beginning," the King said, gravely, "and go on till you come to the end: then stop."

—Lewis Carroll, *Alice's Adventures in Wonderland*

The application of robust control theory is fundamentally limited by the fidelity of the nominal model and the applicability of the uncertainty description. Unfortunately, theoretical development in the field of control engineering has diverged in many directions, often leaving no hint as to how techniques might be applied to a real system. In particular, there have been strikingly disparate approaches taken in the two areas of system identification and robust control.

Traditionally, experimental control has been an iterative process: (*i*) develop a nominal model through first principles and identification, (*ii*) verify the nominal model with simulation, (*iii*) develop a robust model, (*iv*) design a robust controller, and finally (*v*) implement and test the robust controller. This process often involves multiple stages of iteration. There has historically been a gap between the methods used for identifying nominal models and the methods used for augmenting these nominal models with uncertainty suitable for robustness analysis and synthesis of robust controllers. This gap results from the lack of a consistent set of tools for identifying nominal models using experimental data, and for constructing robust models and synthesizing robust controllers.

Under these circumstances, the concept of robustness is poorly defined: it only addresses the robustness of the control system with respect to the model set. What we seek as control engineers is a systematic method which produces robustness measures which are relatable to all observed data from the actual system under study.

Following is a discussion of several fundamental areas of control theory which are used in this dissertation. It will be demonstrated that new theoretical results can be systematically exploited to provide a methodology for robustness analysis and synthesis in the context of experimental systems.

## Trends in ID

A trend in system identification research has been to develop models which minimize the predictive error of the model in the time-domain. A classic treatment of this subject is presented by Ljung [44]. The advantage of the predictive error method is that it provides an explicit technique for identifying parameters in state-space multi-input/multi-output (MIMO) systems. More recently, frequency-domain approaches have matured to the point that they can effectively deal with MIMO identification, Bayard [6].

## Trends in Control

The last ten years have witnessed substantial progress in both nonlinear control and in robust multi-variable control. The results in both areas have been revolutionary, offering both mathematical depth and practical applicability. Consider a helicopter: during large maneuvers helicopter dynamics are highly nonlinear with certain well-understood characteristics that can be exploited with the nonlinear theories of Isidori, Krener, Hunt-Su-Meyer, and others [36, 37, 41]. Unfortunately, there is also substantial modelling uncertainty that precludes systematic application of these techniques.

It is exactly the kind of poorly modelled dynamics that arise in the helicopter problem that have been the focus of research in robust control [16, 17, 20, 19, 28]. Unfortunately, even the most powerful techniques such as $\mathcal{H}_\infty$ , $\mu$-analysis, and $\mu$-synthesis work with nominally linear models and treat nonlinearities as uncertainty, even when aspects of those nonlinearities are well known. Thus, to fully address the helicopter control problem a new blend of these methods is required, and recent developments in both areas suggest that the time is right to do this in specific problem domains such as helicopter control.

In the nonlinear area, the work of Krener, Hauser, and others on approximate linearization has extended the range of systems for which their geometric techniques are applicable [33, 32, 31, 30, 40]. Their work can be viewed as providing a way to remove the effects of certain dominant and well-understood nonlinearities. The remaining nonlinearities and uncertainty can then be handled with robust control methods. To be more specific, suppose that an approximate linearization of a nonlinear system neglects modelling uncertainty and disturbances in such a way that the resulting system has acceptable behavior for a class of maneuvers. Then $\mu$-synthesis techniques can be used to design additional feedback that will provide robustness to the unmodelled dynamics and disturbances.

Nonetheless, there are several unresolved issues that must be addressed before such techniques can be carried out. Approximate linearization typically assumes that the desired response is a single linear system across the entire operating envelope. This assumption is clearly undesirable in helicopters, because desired helicopter behavior near hover is vastly different from that during an obstacle avoidance maneuver, so it is natural to consider the ideal response to be a function of operating point. Furthermore, the residual nonlinearities mean that even if the ideal response were linear across the operating conditions, the robust controller is faced with a nonlinear system. While it might be possible to simply design a fixed linear robust controller to handle these nonlinearities, it is almost certain that some scheduling would improve performance.

Fortunately, there are new extensions to the $\mathcal{H}_\infty$ and $\mu$ theories which provide systematic methods for the design of robust gain scheduled controllers [2, 9, 57, 54]. While these results are too new to have been applied to many practical problems, they appear particularly promising.

### Model Validation

The focus of modern control research has been the development of a consistent set of tools which provide a suitably rich modelling structure, including uncertainty descriptions which incorporate unmodelled dynamics and parametric uncertainty, disturbances, and noise, and which is computationally tractable. Currently there is no systematic procedure which solves both identification and control simultaneously in a robust framework.

For the first time, the disparate methods in identification and robust control are being brought together into a consistent, end-to-end methodology through the use of model validation. The "model validation" problem was originally formulated by Smith and Doyle to provide a connection between a robust model and data measured from a physical system [63]. Model validation seeks to answer the question: "Does the robust model account for the measurements from the physical system?"

A new end-to-end iterative design process, that incorporates frequency-domain model validation analysis, $\mu$-analysis and $\mu$-synthesis, simulation, and implementation is proposed in this dissertation which provides a direct connection between experimental data and both nominal and uncertainty modelling. This methodology closes a circle, bringing together for the first time the disparate methods for identifying nominal models and the construction of uncertainty models suitable for robustness analysis and the synthesis of robust controllers.

## 1.1 A helicopter case study

Experimentation is of great benefit to any theoretical work which has been motivated by practical application, not only to demonstrate the applicability of the theory, but to highlight its potential deficiencies. The use of simulation, while an integral aspect of research, is not entirely adequate because it often fails to adequately characterize the properties of complex systems. The helicopter provides an excellent case study for robust control, as the helicopter industry has long been plagued with a very difficult problem: the high degree of nonlinearity, modelling uncertainty, modal coupling, and instability in rotorcraft has made the design of autonomous autopilots capable of tracking and obstacle avoidance nearly impossible.

Motivation for studying the helicopter comes from problems experienced when designing autonomous guidance and control systems for helicopters in Nap-of-the-Earth (NOE) flight. When flying NOE ("hugging the ground"), the guidance and control system must not only track a desired nominal path but also dynamically sense and avoid local obstacles. During these flight maneuvers, the helicopter dynamics are highly nonlinear and there is also a great deal of model uncertainty. Current control design methodologies do not fully address the problems of uncertain nonlinear systems.

Radio Controlled (RC) rotorcraft technology has reached a point where reasonable experiments can now be developed with RC helicopters. RC helicopters share most of the properties of real helicopters. In particular, the presence of strong cross-couplings and nonlinearities makes them difficult to identify and control. Furthermore, research conducted on an experimental platform based on an RC helicopter is directly applicable to most complex systems, because they share the same type of difficulties, such as complicated dynamics, leading to both parametric and dynamic uncertainty, unmeasurable states, sensor and actuator noise, saturation and quantization, bandwidth limitations, friction, and delays.

To assess the new methodologies developed in this dissertation, an experimental RC helicopter platform was developed. The experimental platform consists of a fly-by-wire RC helicopter under direct real-time computer control. The real-time computer is integrated to the helicopter through direct driven actuators on the helicopter, on-board sensors, and a two-way telemetry system. The real-time hardware and software platform was specifically designed to provide rapid and transparent implementation of theoretical control methodologies on experimental systems. The experimental platform provides a unified system for data collection, identification, analysis and synthesis, simulation, and real-time controller implementation.

The experimental helicopter platform is used to assess the effectiveness of traditional methods available to the control engineer, e.g., system identification using experimentally collected input/output data and $\mathcal{H}_\infty$ and $\mu$-synthesis. The inability of these methods to address problems typically found in complex systems like helicopters, provided the motivation to develop new more systematic methodologies which leverage experimental data.

## 1.2 Contribution of the dissertation

The contribution of this dissertation is the development of a consistent methodology and systematic framework which connects system identification with the construction of robust models through the use of experimental data. Throughout this dissertation the helicopter platform is used as a testbed for all new techniques.

The helicopter is a real nonlinear platform which is extremely useful and suitable for evaluating the applicability of linear identification and linear robust control techniques. Of great importance was the development of a general method which bridged the gap between identification and robust modelling through model validation. Overall, the helicopter case study provided an experimental evaluation and constructive critique of both existing and new techniques in control theory.

The model validation methodology, based on the work of Newlin and Smith [52], and further developed and used in this dissertation, has proven to be an extremely useful tool for evaluating and designing robust models and robust controllers for experimental systems.

## 1.3 Organization of the dissertation

This work is organized as follows. Chapter 2 describes the general purpose real-time hardware and software system used for carrying out experiments on the helicopter. Chapter 3 provides motivation for using a helicopter as a robust control case study, develops the necessary notation and theory for modelling the helicopter, and describes the experimental helicopter platform. Chapter 4 discusses techniques used to identify a nominal model of the helicopter in hover. Chapter 5 reviews frequency-domain robust control theory. A discussion of LQG and $\mathcal{H}_\infty$ hover controllers based on the nominal models developed in Chapter 4 is presented in Chapter 6, and underscores that conventional techniques for synthesizing robust controllers are often not suitable for complex systems such as a helicopter. Chapter 7 develops a general methodology based on new techniques in model validation whereby a direct connection is made between experimental data, nominal models, and robust models. Several examples are considered to evaluate the applicability and practicality of the model validation software which was developed. The model validation methodology is then applied to the helicopter to iteratively develop a better robust hover model and robust hover controller in Chapter 8. Finally, a discussion of the implications of this work is contained in Chapter 9.

# Chapter 2
# Real-time Experimental System

The importance of research on experimental systems and the interaction of theory with application is often neglected in the academic control community. This is true mainly because it is (in practice) very difficult to make the transition from mathematical models of systems developed for the purpose of simulation to real-time implementation on an experimental system.

A unified platform of real-time hardware and software was developed by Morris and Van Nieuwstadt to facilitate this transition [48]. This platform allows students and researchers to get hands-on experience with control on real, complicated physical systems without having to invest a great deal of time or money developing custom hardware and software for each experiment. It is in use at Caltech and is accessible by undergraduates, graduates, and faculty doing research in control. This platform was used for all experimental research described in this dissertation.

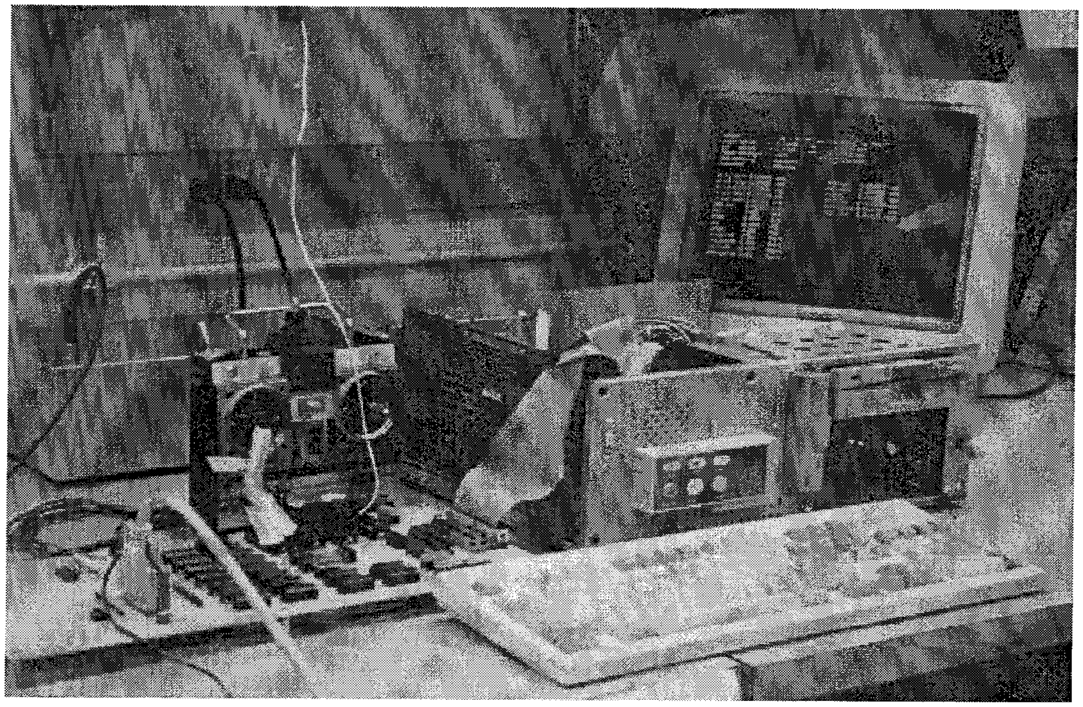

Figure 2.1: Photograph of the real-time experimental system.

Section 2.1 describes the real-time computer and input/output (I/O) device support. The host and real-time software are discussed in Section 2.2.

## 2.1 Real-time hardware

The real-time experimental platform consists of an I/O interface board, a high speed digital signal processing (DSP) board, an IBM PC compatible host computer, and Sun workstations. A photograph of the PC, DSP board, and I/O interface is shown in Figure 2.1. A block diagram representation of the experimental platform is shown in Figure 2.2.

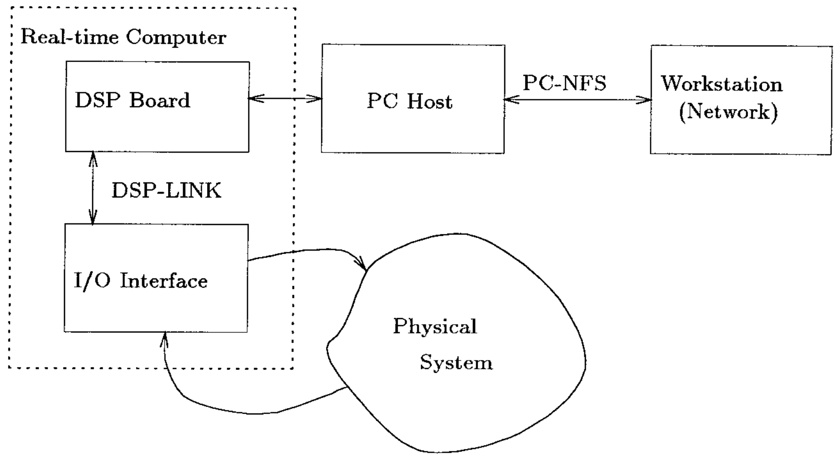

Figure 2.2: General experimental setup.

The DSP board and I/O interface comprise the real-time computer (Figure 2.3). The DSP board provides a high speed numerical engine while the I/O board serves as the interface to physical systems. The DSP board is located inside an IBM PC. The user interacts with the real-time computer during experiments through a user interface on the IBM PC. The IBM PC is connected to a network of Sun workstations through PC-NFS. A transparent link from the workstations to the real-time computer is provided by the host software system. The workstations are typically used for off-line computations, such as analysis of experimental data, system identification, simulation, and controller synthesis. The inclusion of workstations in this setup is merely a convenience. The PC, DSP board, I/O interface, and associated software are all that is necessary to run real-time experiments.

The overall goal while designing the hardware was to develop a system which could be easily applied to a variety of different experiments. Issues taken into consideration were the ability to acquire data at high speeds, implement closed-loop discrete-time systems at high speeds, and interface to a wide variety of physical signals.

### 2.1.1 DSP board

The DSP board is a standard IBM compatible ISA board built by Spectrum Signal Processing and is based on the Texas Instruments TMS320C30 (C30) DSP [66, 67]. The selection of a C30 based DSP board was driven by the need to run real-time at high speeds. The C30 has excellent on-chip facilities for this task: a 32-bit programmable timer capable of registering interrupts and up to 30 MFLOPS throughput [68].

Located on the DSP board is a 200 kHz 16-bit A/D and a 200 kHz 16-bit D/A, each with two multiplexed channels, and a 16-bit external bus, DSP-LINK, which is used to interface with the I/O board [65]. Communication between the PC and the DSP board is accomplished by a dual-access memory buffer, which is discussed in greater detail in Sections 2.2.1 and 2.2.2.

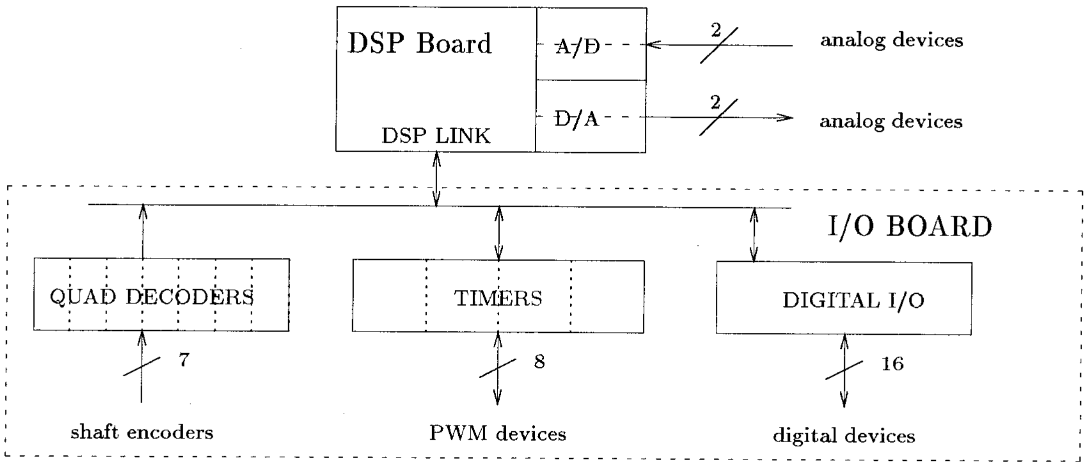

Figure 2.3: Block diagram of the real-time computer.

### 2.1.2 I/O board

The I/O board serves as the physical interface between experimental signals and the real-time computer. Refer to Figure 2.3 for a block diagram of the real-time input and output devices located on the I/O board. Signals supported include pulse-width modulated (PWM) inputs and outputs, digital inputs and outputs, and quadrature encoder inputs.

Table 2.1 summarizes the types of input and output devices which are supported by the real-time system. $\omega_{max}$ is defined below. A photograph of the I/O board is shown in Figure 2.5.

Note that all of the input and output devices shown in Table 2.1 are represented by unscaled binary data in the real-time software. When used in a real-time experiment, as outlined in Section 2.2.2, scaling factors which transform the binary data to the "default unit" must be explicitly taken into account.

| Signal type | In | Out | Default unit | Range | Resolution |
|---|---|---|---|---|---|
| analog | 2 | 2 | volts | $\pm 3$ | 16 bits |
| digital | 16 | 16 | N/A | N/A | N/A |
| PWM | 8 | 8 | seconds | see Table 2.2 | 16 bits |
| Quad. encoder | 7 | N/A | radians | $2\pi$ | 11 bits |
| Quad. encoder rate | 7 | N/A | radians/second | $\omega_{max}$ | N/A |

Table 2.1: Real-time I/O devices. The In and Out column indicate the number of input and output channels, respectively.

In general the scaling used to convert from binary data to the "default unit" can be represented by

$$\text{default unit} = \frac{\text{Range}}{2^{resolution}} \times (\text{binary data}). \tag{2.1}$$

## PWM I/O

The PWM interface is implemented with four AM9513A system timer chips [1]. They are fully programmable and can produce and measure a variety of signals. The use of these chips falls into 2 categories: frequency division to produce precise pulse widths, and count accumulation to measure pulse widths. Each chip contains five 16-bit counters. The counters can be gated by hardware or software, and can be internally concatenated. The main clock frequency is 5 MHz.

| Clock Source | Maximum Pulse Width | Maximum Period |
|---|---|---|
| 200 ns | 13.10 ms | 26.20 ms |
| 2 $\mu$s | 131.00 ms | 262.00 ms |
| 20 $\mu$s | 1.31 sec | 2.62 sec |
| 200 $\mu$s | 13.10 sec | 26.20 sec |
| 2 ms | 131.00 sec | 262.00 sec |

Table 2.2: Programmable PWM modes.

The source clock for each counter is an independently programmed division of the main clock with 16-bit resolution. This allows a maximum pulse width of 13 ms with a resolution of 200 ns and of 131 seconds with a resolution of 2 ms. A PWM signal is defined by its period and maximum pulse width. The software drivers, discussed in Section 2.2.2, automatically select the clock source period to provide maximum pulse width resolution for a given period and maximum pulse width specification. Table 2.2 contains a tabulation of the available modes of PWM signals.

Eight counters are used for PWM inputs and eight for PWM outputs. One counter is used as the PWM period source. The setup time of the AMD9513A is in excess of 1.5 $\mu s$ per counter access per chip. For experiments which use PWM signals the counters limit the maximum sample rate.

The software driver converts a discrete-time series into a PWM signal. Each PWM input and output is sampled at a programmable rate, as discussed previously. The value of the discrete-time series at each sampling interval is converted into a pulse width by the AMD9513A.

Let $\bar{u}$ be the nominal pulse width and $\delta u(k)$ be the differential pulse width for each sample $k$. Then the pulse width for each sample will be $u(k) = \bar{u} + \delta u(k)$. An example of this conversion is shown in Figure 2.4, where the discrete-time series values $\delta u(k)$ are indicated by an "x" in the upper graph, with the corresponding PWM signal in the lower graph.

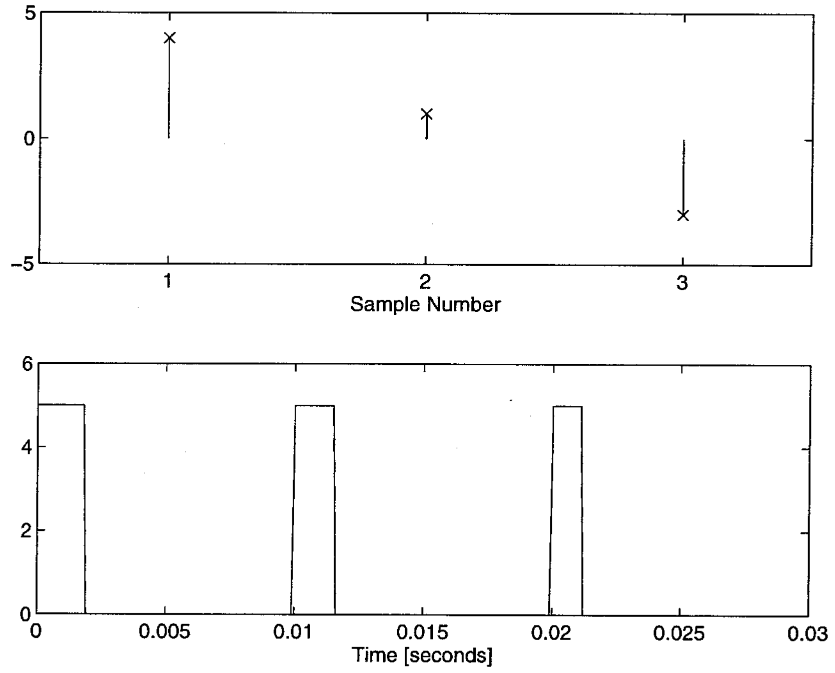

Figure 2.4: PWM encoding example. Top: discrete-time signal. Bottom: PWM signal corresponding to a modulation of the discrete-time signal.

In this example, the nominal pulse width is set to $\bar{u} = 1.5$ ms. The scaling factor used to convert $\delta u(k)$ into a differential pulse width is 100 $\mu$s. The PWM period was set to 10 ms. Referring to Figure 2.4, the conversions performed by the software drivers are

$$\begin{aligned} \delta u(1) = 4 &\quad\Longrightarrow\quad u(1) = 1.5\text{ms} + 4 \times 100\mu\text{s} = 1.9\text{ms} \\ \delta u(2) = 1 &\quad\Longrightarrow\quad u(2) = 1.5\text{ms} + 1 \times 100\mu\text{s} = 1.6\text{ms} \\ \delta u(3) = -3 &\quad\Longrightarrow\quad u(3) = 1.5\text{ms} - 3 \times 100\mu\text{s} = 1.2\text{ms}. \end{aligned} \tag{2.2}$$

## Quadrature encoder inputs

The quadrature decoders are implemented with IXYS502 chips [38]. Each IXYS502 chip has an internal 8-bit accumulator and is buffered externally with a 16-bit sign extended D-latch. There are seven independent channels. They are primarily intended to measure the angular position of a rotating shaft.

Given the number of lines, $L$, of the quadrature encoder, the angular resolution, $\theta_{res}$, of the quadrature decoder is given by $\theta_{res} = \frac{2\pi}{4L}$ rad. The factor $4L$ results from the quadrature encoder generating four edges for each line. There is an upper limit on the angular velocity, $\omega_{max}$, to prevent the quadrature decoder from overflowing. This is a function of the sampling period $T$ and is given by $\omega_{max} = 127\frac{\theta_{res}}{T}$ rad/sec. Without digitally filtering the angle measurements, the maximum angular velocity resolution when using raw encoder data will be $\omega_{res} = \frac{\theta_{res}}{T}$ rad/sec. For this reason, an IIR filter is implemented in the software driver to provide angular velocity measurements for the quadrature encoders (2.3).

The quadrature encoder angular rate inputs in Table 2.1 are not physical devices, rather they are implemented in the real-time software as second order IIR filters on the quadrature encoder angle inputs. The form of the IIR filter is

$$\sum_{i=0}^{N} a_i y(k-i) = \sum_{j=0}^{N} b_j u(k-j) \tag{2.3}$$

where $u$ correspond to the quadrature encoder angle measurements and $y$ the filtered signal corresponding to the quadrature encoder rate measurement.

The specific parameters used in the real-time software are shown in Table 2.3. The resolution of the rate filter is dependent on these IIR coefficients.

| Coefficient | Value | Coefficient | Value |
|---|---|---|---|
| $a_0$ | 1.00 | $b_0$ | 0.00100 |
| $a_1$ | -1.60 | $b_1$ | 0.00200 |
| $a_2$ | 0.64 | $b_2$ | 0.00100 |

Table 2.3: Real-time IIR quadrature encoder rate filter coefficients for (2.3).

## Digital I/O

Digital I/O is provided by two 16-bit ports: one for input and the other for output. These ports allow for custom equipment, e.g., LED indicators, to be easily attached to the experiment. The use of digital I/O requires custom code to be linked with the real-time software.

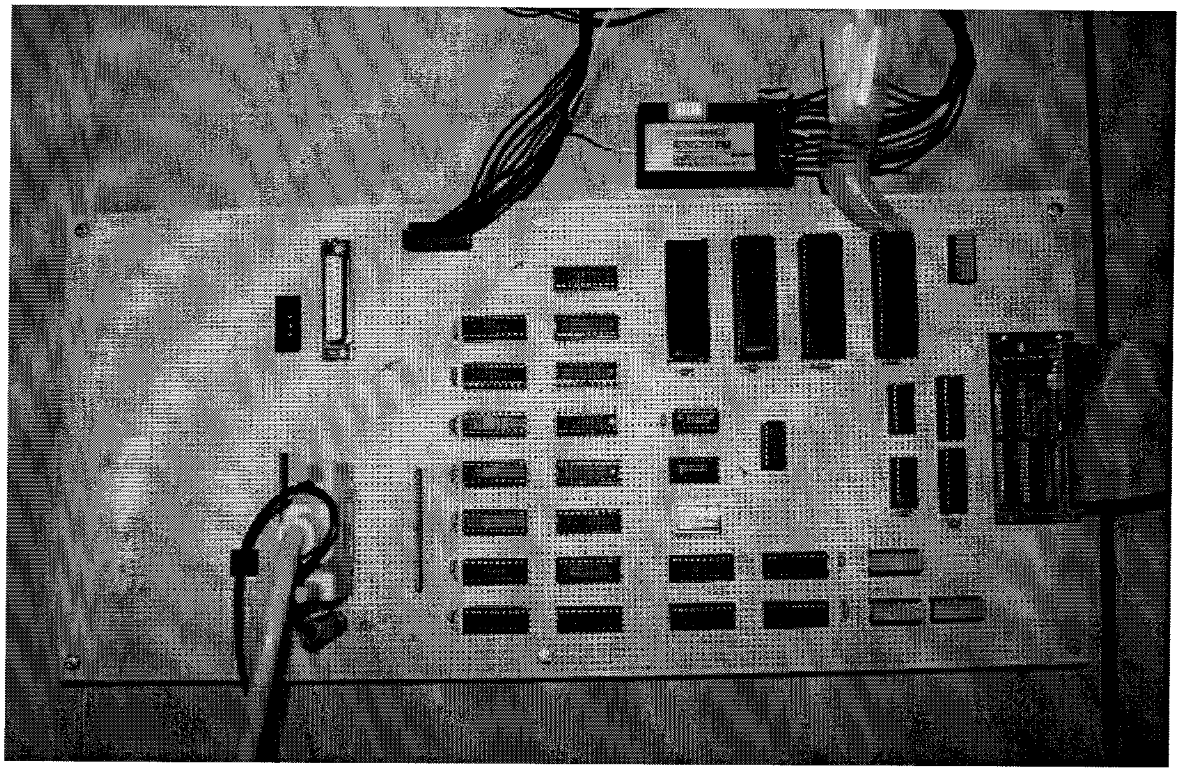

Figure 2.5: Photograph of the I/O board.

## 2.2 Software

In designing the software it was assumed that the bulk of off-line computation would be done with MATLAB toolboxes, such as $\mu$-tools [5], LMI Control [26, 25], and System Identification [45]. With this in mind a layered software architecture, depicted in Figure 2.6, was developed.

At the highest level users interact with data inside MATLAB. There are tools provided which allow MATLAB objects to be saved in a specification file and for log files of real-time experiments to be loaded directly into MATLAB objects. The host system is used to download the specification file and launch the real-time application on the real-time computer, flush and store real-time log files, monitor the progress of the experiment, and handle asynchronous on-line user events. The real-time software handles all real-time computation, data acquisition and signal generation.

### 2.2.1 Host software

The host software, running on the PC, is responsible for bootstrapping the real-time computer and interacting with the user and the real-time computer. The syntax for the program is:

`rt` filename1.rt `-o` filename2.log.

The specification file, `filename1.rt`, is read from disk and downloaded to the real-

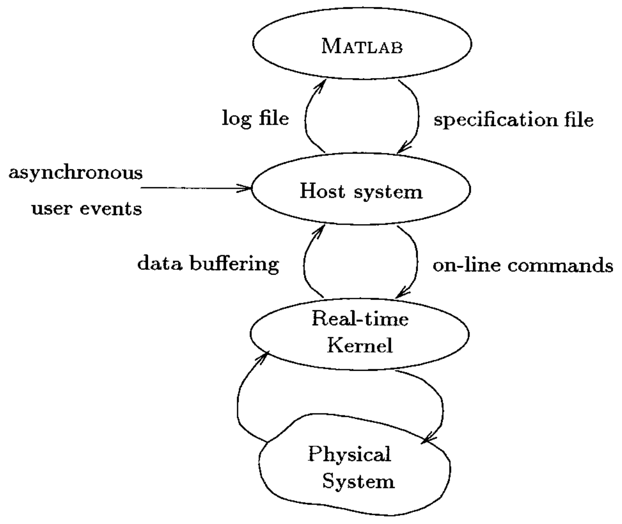

Figure 2.6: Real-time system software architecture.

time computer, then a reset of the real-time software occurs starting the real-time application. The specification file contains all information needed by the real-time application, which is described in Section 2.2.2. As the real-time software is running the host software flushes the logged data from the dual-access buffer to the log file, `filename2.log`, and displays it on the host system screen to provide on-line monitoring of the experiment. If signals go outside pre-specified "safety" ranges this is indicated on the screen. The length of time a logged experiment can run is limited only by available disk space.

While the experiment is in progress external user events, listed in Table 2.4, can be triggered on-line by pressing keys on the keyboard or flipping a switch on the RC transmitter.

| **Event** | **Description** |
|---|---|
| `c` | Toggle the synchronized output tables |
| `k` | Toggle the controller |
| `p` | Add a sequence of pulses to the PWM outputs |
| `q` | End the experiment |
| `t` | Trim the input and output PWM signals |
| `z` | Zero the quadrature encoders |
| `Switch` | Toggle throttle hold |

Table 2.4: Real-time software external events. The events are signaled by pressing the appropriate key on the real-time host computer keyboard.

The user events were dictated mainly by the helicopter experiment described in Chapter 3. Although in the current implementation they are specific to this experiment it is straightforward to implement a generic set of user events applicable to a wide range of experiments.

### 2.2.2 Real-time software

The real-time software was developed specifically for the task of implementing a single-rate discrete-time system. The C30 on-chip 32-bit timer is utilized as the real-time clock. The resolution of the timer is 120 ns, providing a maximum sample period of about 500 sec. The real-time clock generates a non-reentrant interrupt at the sampling rate. If, for some reason, the interrupt occurs before computation of the previous sample was completed, an exception is generated causing the system to shutdown in a well-defined fashion.

The real-time software is bootstrapped by the host system. The specification file is first downloaded into a dual-access memory buffer. Then the host system issues a reset signal which starts program execution on the DSP board. There are several default data structures loaded into the dual-access memory buffer, including data structures used by device drivers defining the type and number of physical inputs and outputs, data needed by `IOtransform`, and data tables used by `TableLookup`. `IOtransform` is a function that defines the input/output mapping; in other words, how mathematical or virtual signals are mapped to or from physical devices. `TableLookup` is a function that implements periodic lookup tables, which might be used to add excitations or trim functions to the system, see Table 2.5.

There are two classes of signals treated in this discussion. *Physical* inputs, denoted $\hat{u}$, and outputs, denoted $\hat{y}$, refer to raw device data. *Virtual* inputs, denoted $u$, and outputs, denoted $y$, are device independent mathematical signals and are processed by software drivers which directly interface with the physical devices. This processing involves scaling and trimming for any DC offset or nominal signal value. Additionally, a user programmable IIR filter is built into the real-time software to compute velocity information from quadrature encoder angular position data (2.3).

The algorithm implemented by the real-time software at each sample $k$ is presented in Table 2.5. $u_i^{nom}$ and $y_i^{nom}$ are the programmable DC offsets (trims) used by the software drivers. Some of the steps in the algorithm are dependent on the external asynchronous user events described in Section 2.2.1 and Table 2.4. For simplicity, steps 5, 6, 7, and 9 of this algorithm were customized for the helicopter experiment discussed in Chapter 3. It is straightforward to generalize this algorithm and remove all such application specific dependencies.

The default transform, `IOtransform`, used by the real-time software, is the linear shift-invariant (LSI) discrete-time system

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) \\ y(k) &= Cx(k) + Du(k), \end{aligned} \tag{2.4}$$

where $x$ is the state vector, $u$ the *virtual* input vector, and $y$ the *virtual* output vector. $A$, $B$, $C$ and $D$ are constant matrices. Note that because the LSI system acts on *virtual* inputs and generates *virtual* outputs, the $B$, $C$ and $D$ matrices must reflect the appropriate scalings as defined by (2.1). Finally, for each sample, the *physical* inputs and outputs are logged into memory and saved and flushed by the host system.

1. Synchronize external user events from Table 2.4.
2. Sample and process each $\hat{u}_i(k)$.
3. Compute *virtual* inputs: $u_i(k) = \hat{u}_i(k) - u_i^{nom}$.
4. If the controller or trim event was toggled on: trim aileron, elevator, and rudder inputs and outputs; otherwise, if the controller was toggled off: reset aileron, elevator, and rudder trim.
5. If the external throttle hold switch was toggled on: set throttle hold value to current throttle command; otherwise, if throttle hold is on: set throttle input to throttle hold value.
6. If the external pulse event was toggled on: enable pulse addition; otherwise, if pulse addition is on: add pulses to aileron, elevator, and rudder inputs.
7. If the controller was toggled on: zero controller state.
8. If the controller is on: compute $y(k) = \texttt{IOtransform}\{u(k), k\}$; otherwise, pass PWM inputs directly to PWM outputs.
9. $y(k) = y(k) + \texttt{TableLookup}(k)$.
10. Compute *physical* outputs: $\hat{y}_i(k) = y_i(k) + y_i^{nom}$.
11. Write each $\hat{y}_i(k)$ to appropriate device.
12. Log $\hat{u}_i(k)$ and $\hat{y}_i(k)$ into dual-access buffer.

Table 2.5: Real-time software sampling iteration.

In cases where an LSI system like (2.4) is not adequate, custom code can be linked into the device-independent function `IOtransform`. No other function need be modified to implement arbitrary nonlinear time-varying functions. For example, FM synthesis at audio rates is easily achievable with a custom `IOtransform`.

The maximum sample rate achievable by this system is driven by two major factors: ($i$) the total number of inputs and outputs, and ($ii$) the number of states. For example, at one extreme the maximum achievable sample rate with 50 states, 25 inputs and 11 outputs is 200 Hz; which corresponds to 8 PWM inputs, 7 quadrature angle inputs, 7 quadrature angular rate inputs, 2 analog inputs, 1 digital input, 8 PWM outputs, 2 analog outputs, and 1 digital output. At the opposite extreme, the maximum achievable sample rate, with no states, 2 analog inputs and 2 analog outputs is 12.5 kHz. Most of the software overhead results from the I/O drivers, which are written in C, and the slow access time of the AMD9513A system timer chips. If the I/O configuration were hard-coded, then a decrease in the software overhead by about a factor of ten could be expected. In lieu of hard-coding the I/O configuration, carefully rewriting the drivers in assembly code could decrease this overhead by about a factor of five.

The real-time software is not limited to implementing controllers. There are many tasks that are directly realizable without reprogramming the default `IOtransform`, including multi-variable FIR and IIR filtering, complex waveform synthesis and data collection. The ability to produce table-driven outputs and sample both inputs and outputs is of great importance in any experimental control research program where identification and real-time controller implementation is necessary.

### 2.2.3 MATLAB interface

The MATLAB interface provides a set of tools to read real-time log files into MATLAB objects and save system specification files. The command

$$[u\ y\ T] = \texttt{loadrt}(\text{filename})$$

reads a real-time log file and stores the logged inputs into $u$, the logged outputs into $y$, and the sample period into $T$. Standard MATLAB functions and toolboxes can then be used to work with $u$ and $y$. For example, to identify an LTI discrete-time model for a system given a parametric state-space structure *thstruc* with the *System Identification Toolbox* use the command

$$[A\ B\ C\ D] = \texttt{th2ss}(\text{pem}([y\ u], thstruc)).$$

Given this model, a controller can be synthesized with $\mu$–tools using the following commands:

$$\begin{aligned} [A, B] &= \texttt{d2c}(A, B, T) \\ sys &= \texttt{pck}(A, B, C, D) \\ K &= \texttt{h2syn}(sys, outputs, inputs) \\ [A_k, B_k, C_k, D_k] &= \texttt{unpck}(K) \\ [A_k, B_k] &= \texttt{c2d}(A_k, B_k, T). \end{aligned}$$

In order to implement the controller first create the real-time specification file with

$$\texttt{savert}(\text{filename}, T, [A_k\ B_k;\ C_k\ D_k], 0, M_i, M_o),$$

where $T$ is the sample period, and $M_i$ and $M_o$ define the input/output mapping. Next, run the host software on the PC as outlined in Section 2.2.1. The controller is then immediately implemented in real-time.

This simple exercise provides just one example of the way that MATLAB tools might be used with the real-time platform. The use of MATLAB and MATLAB toolboxes is for convenience. Creating specification files and reading log files outside of MATLAB only involves writing software compatible with the format used by the host system.

## 2.3 Summary

The real-time platform discussed in this chapter has been in regular use at Caltech for the last three years. We have had great success using it with the helicopter experiment to provide rapid transition from data collection to system identification to controller synthesis to controller implementation. This is essential in experimental control research, as it is usually necessary to go through many iterations of these tasks in order to achieve desirable system behavior.

Development of the real-time platform is ongoing, including enhancing the software to handle multi-rate paradigms, making the host system and MATLAB interfaces more user friendly, and designing custom ASICs both to reduce the physical size of the real-time computer and to more easily interface to sensors/actuators. Additional work has focused on developing generic telemetry and miniaturizing the electronics to allow the entire system to be used on-board small autonomous experiments.

# Chapter 3
# Helicopter Case Study

An RC helicopter was used as a testbed in this dissertation; a photograph of this testbed is shown in Figure 3.1. The RC helicopter is interesting because it shares many of the properties of real helicopters. In particular, RC helicopters are highly nonlinear with complex dynamics and strongly coupled modes, and the experimental testbed has unmeasurable states, sensor and actuator noise, actuator saturation and rate limits, bandwidth limitations, friction, and delays.

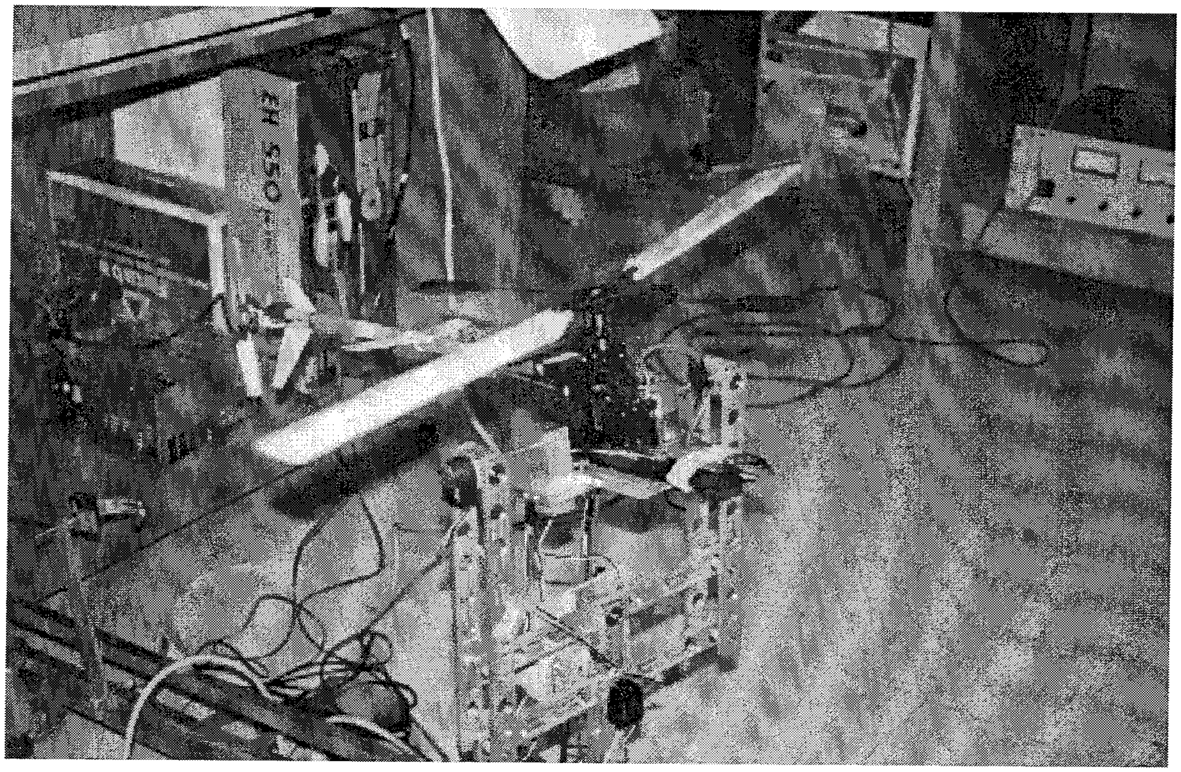

Figure 3.1: Photograph of the helicopter experimental platform.

Because of this, identification, modelling, and controller analysis and synthesis research conducted on the RC helicopter experimental platform will be directly applicable to most systems encountered by the control engineer. Based on experimental results with the RC helicopter, it will be shown in this dissertation that there is strong motivation to develop new theoretical techniques which bridge the gap between identification and robust modelling and robust linear and nonlinear control.

Furthermore, the RC helicopter provides a great testbed for studying problems in real-time control. Helicopter models generally require a large number of states to adequately capture the dynamics. This often results in high order controllers which rely on high speed digital signal processing when implemented. In order to control the helicopter untethered extremely small and accurate sensors and actuators are necessary for measuring the attitude, angular rates, and accelerations. A small, lightweight on-board real-time flight computer, or off-board flight computer and telemetry system is necessary to implement controllers. In many cases, a telemetry system is also needed to supply pilot commands.

Section 3.1 contains a brief description of the helicopter and reviews the principal nonlinearities found in helicopters. A description of the RC helicopter and associated experimental platform used as a case study in this dissertation is discussed in Section 3.2. A parametric structure for a state-space linear model for a standard helicopter operating near hover is developed in Section 3.3. We will delay development of an identified nominal helicopter model until Chapter 4 and robust models until Chapters 6 and 8.

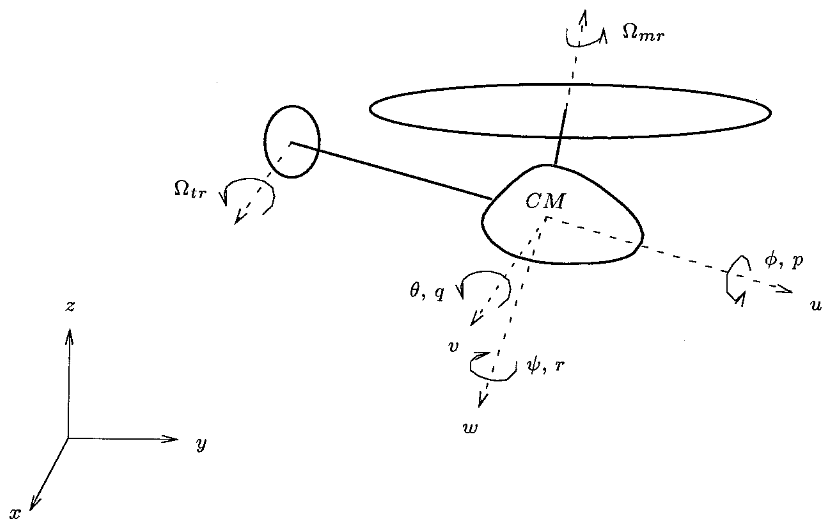

Figure 3.2: Helicopter drawing showing notation conventions.

## 3.1 The helicopter

A discussion and review of nonlinear helicopter modelling is presented to summarize the relevant nonlinearities for a helicopter hovering at low altitude. A comprehensive treatment of helicopter theory is provided by Johnson [39] and Gessow and Meyers [27]. A simplified hybrid linear/nonlinear model used by NASA is discussed by Lewis *et al.* [42, 43]. An analytic derivation of a nonlinear model for the particular case of an RC helicopter mounted on a stand is covered in detail by Weilenmann [72]. Weilenmann further develops a robust model for the RC helicopter operating near hover and in vertical flight.

### 3.1.1 Notation and description of helicopter operation

The notation in Table 3.1 will be used throughout this dissertation when referring to helicopters. Refer to Figure 3.2 for a drawing of a typical helicopter with corresponding notation.

| **Measurements** | **Description** |
|---|---|
| $\phi$ | Body roll angle |
| $\theta$ | Body pitch angle |
| $\psi$ | Body yaw angle |
| $p$ | Body roll angular rate |
| $q$ | Body pitch angular rate |
| $r$ | Body yaw angular rate |
| $u$ | Body longitudinal velocity |
| $v$ | Body lateral velocity |
| $w$ | Body vertical velocity |
| $\Omega_{mr}$ | Main rotor angular velocity |
| $\Omega_{tr}$ | Tail rotor angular velocity |
| $x$ | Inertial position |
| $y$ | Inertial position |
| $z$ | Inertial position |
| **Controls** | **Description** |
| $\Theta_a$ | Lateral cyclic (aileron) |
| $\Theta_e$ | Longitudinal cyclic (elevator) |
| $\Theta_r$ | Tail rotor collective blade pitch (rudder) |
| $\Theta_c$ | Main rotor collective blade pitch (collective) |
| $\Theta_t$ | Engine throttle |
| **Miscellaneous** | **Description** |
| $CM$ | Center of mass |
| $T_E$ | Nonlinear Euler transformation (body rates) |
| $T_{IB}$ | Nonlinear transformation (body to inertial coordinates) |

Table 3.1: Helicopter notation.

A helicopter is an airborne vehicle that generates lift, propulsion, and control with rotating wings. Unlike traditional fixed wing aircraft, the helicopter is able to generate lift even when its velocity is zero. There are several configurations found in helicopters today, e.g., some helicopters have only one main rotor while others have two. To simplify this discussion we will consider a helicopter with a single main rotor consisting of two blades and a single tail rotor also with two blades. Such a helicopter is shown in Figure 3.1.

The rotating blades of the main rotor span a nearly horizontal plane (the rotor disk). Lift is generated by accelerating air downwards through the rotor disk. The amount of lift generated is related to the rotor blade pitch (controlled by the collective) and the rotor rotational velocity (controlled by the throttle). The tail rotor spans a plane perpendicular to the rotor disk. This configuration allows the tail rotor to balance the torque about the main rotor shaft. This balancing torque is controlled by the tail rotor blade pitch (rudder). In addition to providing lift and propulsion, the main rotor is also the control surface for roll, pitch, and vertical control. Roll and pitch motion is accomplished by tilting the rotor disk using the aileron and elevator, respectively.

Because the helicopter can generate lift even when it is stationary, it is capable of vertical takeoff and landing (VTOL) and hover. The VTOL and hover capabilities of the helicopter come not without a price. The helicopter's main rotor is a very complicated mechanical system and is the source of vibration. Furthermore, the helicopter possesses only marginal stability properties, especially in hover, and good flight characteristics and handling are difficult to attain without active feedback.

As will be seen, helicopter dynamics are considerably more complex than traditional fixed wing aircraft. In particular, there are noticeable and significant nonlinearities, the principal of which is the subject of Section 3.1.3.

### 3.1.2 Basic rigid body equations

The inertial, or translational, velocity of the helicopter is a nonlinear kinematic transformation of the body velocities and is given by

$$\frac{d}{dt}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = T_{IB}\begin{bmatrix} u \\ v \\ w \end{bmatrix} \tag{3.1}$$

where the body to inertial transformation is given by

$$T_{IB} = \begin{bmatrix} \cos\theta\cos\psi - \sin\phi\sin\theta\sin\psi & -\cos\phi\sin\psi & \sin\theta\cos\psi + \sin\phi\cos\theta\sin\psi \\ \cos\theta\sin\psi + \sin\phi\sin\theta\cos\psi & \cos\phi\cos\psi & \sin\theta\sin\psi - \sin\phi\cos\theta\cos\psi \\ -\cos\phi\sin\theta & \sin\phi & \cos\phi\cos\theta \end{bmatrix}.$$

The body angles of the helicopter obey standard equations for rigid body motion

$$\frac{d}{dt}\begin{bmatrix} \phi \\ \theta \\ \psi \end{bmatrix} = T_E\begin{bmatrix} p \\ q \\ r \end{bmatrix}, \tag{3.2}$$

where the body rate transformation is given by

$$T_E = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ \sin\theta\tan\phi & 1 & -\cos\theta\tan\phi \\ -\sin\theta\sec\phi & 0 & \cos\theta\sec\phi \end{bmatrix}.$$

The dynamics governing $[p,\ q,\ r]$ and $[u,\ v,\ w]$ are very complicated and are dependent on the physical, structural, and aerodynamic properties of the vehicle. The underlying technique for constructing an analytic model of a rigid body vehicle from first principles consists of identifying the external forces, $\boldsymbol{F}$, and moments, $\boldsymbol{M}$, acting on the vehicle. Once found, written in body coordinates they will satisfy standard rigid body dynamical equations

$$\begin{aligned} \sum_i F_i &= \frac{d}{dt}(mv) = m(\dot{v} + \Omega \times v) \\ \sum_i M_i &= \frac{d}{dt}(I\Omega) = I\dot{\Omega} + \Omega \times I\Omega, \end{aligned}$$

where $mv$ is the translational momentum of the vehicle, $I\Omega$ is the angular momentum of the vehicle, $m$ is the mass of the vehicle, $\Omega$ is the angular velocity of the vehicle, $v$ is the translational velocity of the vehicle, $I$ is the moment of inertia of the vehicle, and the summation is done over all of the external forces and moments on the vehicle. It is assumed that the mass and inertia matrix are constant.

### 3.1.3 Principal helicopter nonlinearities

A very important nonlinear effect present in helicopters but not in fixed wing aircraft are the rotor dynamics. This effect arises from the flapping motion of the rotor blades and changes in the velocity of the rotor disk (the rotor disk is defined as the surface spanned by the rotor blades).

Each rotor blade can be modelled as a one degree-of-freedom (DOF) system, corresponding to the flapping modes along the length of the blade. In hover, Johnson has shown that the transfer function from blade pitch control inputs to blade flap motion is a highly damped second-order response [39]. Rotor flapping has an important effect in almost every aspect of helicopter behavior. However, because the rotor flapping dynamics occur at much higher frequency than pilot commands and exogenous disturbances and gusts, it is acceptable in general to consider only the steady-state or low frequency behavior of the rotor.

Similarly, the varying speed of the rotor disk changes the differential thrust of the rotor, which results in nonlinearities that complicate parameter identification and linear control. In particular, in hover, the differential thrust of the rotor is directly proportional to the square of the rotor speed [27].

Another principal nonlinearity during low altitude flight is ground effect. Ground effect is significant when the helicopter is within one rotor diameter above the ground.

A detailed discussion of ground effect is provided by Johnson [39]. Ground effect can be viewed as increasing the effective rotor thrust at low altitudes. This occurs as a result of the air wake generated by the main rotor "bouncing" off the ground and exerting additional force on the rotor disk. This is significant in that it changes the control authority of the main rotor. A graph of a typical thrust curve over the region where ground effect is significant, taken from Johnson [39], is included in Figure 3.3. In Figure 3.3, $T$ denotes the main rotor thrust, $T_\infty$ denotes the main rotor thrust an infinite distance from the ground, $z$ denotes the altitude of the helicopter, and $R$ denotes the radius of the main rotor. It can be seen that at low altitude the effective thrust of the main rotor is significantly higher than at high altitude for the same level of control.

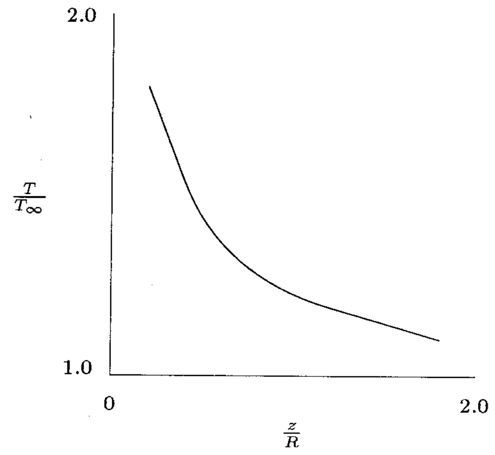

Figure 3.3: Typical thrust curves for significant ground effect, from Johnson [39].

Ground effect primarily affects the helicopter during hover. In forward flight the thrust wake is projected behind the helicopter, preventing an increase in thrust from occurring. Because ground effect occurs when the air wake from the rotors bounces back from the ground, winds and gusting will also change the ground effect thrust.

## 3.2 Experimental helicopter platform

The experimental platform consists of a modified EP Concept RC model helicopter, first described by Morris *et al.* [48, 49]. Refer to Figure 3.1 for a photograph of the helicopter. The 35 inch diameter main rotor is powered by an AstroFlight Cobalt-05 electric motor. The control surfaces of the helicopter are actuated by Futaba S6901 PWM position servos and the engine throttle is regulated by a Team Astro model 207 speed controller. The model helicopter is actuated on each of the three attitude axes: aileron (lateral cyclic), elevator (longitudinal cyclic), and rudder (tail rotor blade pitch). In addition, there are two inputs controlling the main rotor thrust: engine throttle for regulating the rotational velocity of the main rotor and differential collective for regulating the main rotor blade pitch.

Exogenous pilot commands are provided through a six channel Airtronics RC transmitter. An Airtronics receiver is used to convert the transmitter signal into PWM signals which are then connected to the real-time computer I/O board. See Figure 3.4 for the general setup. Note that the pilot commands are not directly connected to the helicopter, rather they are treated as exogenous inputs by the real-time computer and generally used to provide reference commands to the flight control system. In other words, the helicopter is configured as a fly-by-wire system.

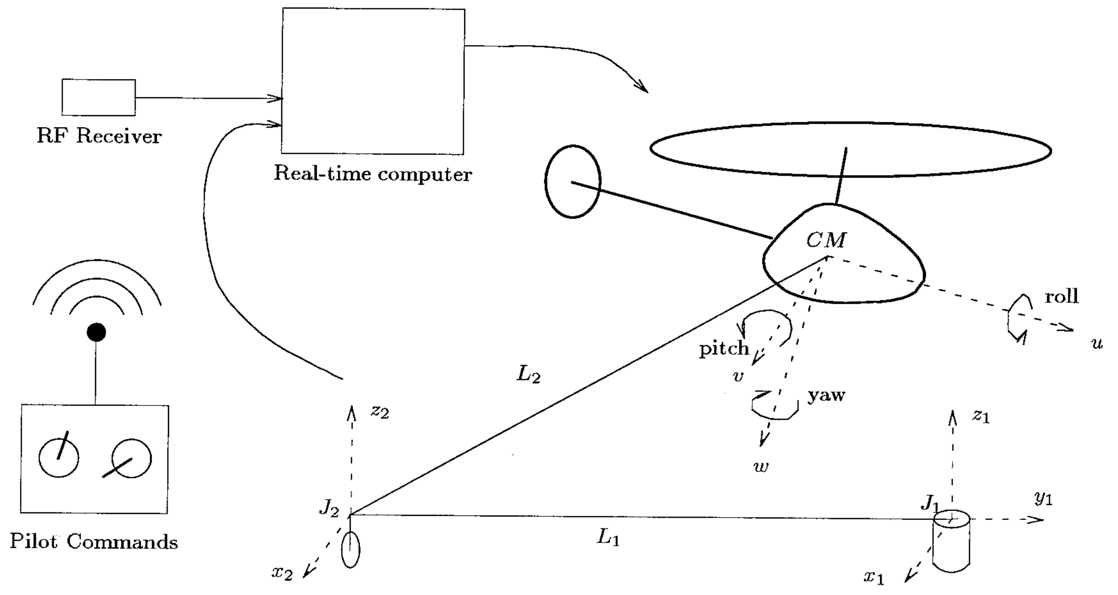

Figure 3.4: Helicopter experimental setup.

The helicopter is mounted on a three DOF wrist which in turn is connected to a three DOF stand (Figure 3.4). The center of mass of the helicopter can be aligned with the rotation axes of the wrist. The wrist allows the attitude (roll, pitch, and yaw axes) of the helicopter to freely rotate. The wrist joint angles are measured using shaft encoders, providing a direct measurement of the helicopter attitude. $J_1$ is a one DOF joint attached to the ground. The link $L_1$ connects joints $J_1$ and $J_2$ and scribes a circle parallel to the ground through the axis $z_1$. $J_2$ is a two DOF joint mounted to a wheel. The link $L_2$ is a four-bar connecting the joint $J_2$ with the helicopter wrist and allows for free rotation about the $z_2$ axis and about +50 degrees of rotation about the $x_2$ axis. The stand allows for six DOF motion in a volume roughly equal to a hemisphere of diameter approximately six feet.

The helicopter is controlled by the real-time computer described in Chapter 2. The joint angles are measured by quadrature shaft encoders and along with the pilot commands from the Airtronics transmitter comprise the inputs. The real-time computer directly drives the helicopter control surfaces through Futaba PWM position servos.

Each servo is nominally driven by a control signal of $\pm 0.5$ ms modulated onto a 50 Hz PWM waveform with a neutral position corresponding to 1.5 ms (Figure 2.4).

At hover the full range ($\pm 0.5$ ms) for each of the servos is not possible, due to the trim command necessary to hold the helicopter at hover equilibrium. The saturation values for each of the servos at hover were determined empirically and are tabulated in Table 3.2.

| **Actuator** | **Saturation Level ($\mu$s)** |
|---|---|
| Aileron | 450 |
| Elevator | 300 |
| Rudder | 380 |
| Collective | 500 |
| Throttle | 500 |

Table 3.2: Actuator saturation levels at hover. The saturation values are the maximum possible deviations from the trim command necessary to maintain hover.

In all future figures displaying PWM servo commands the value of the servo will be shown as a percentage of the saturation value: $+100\%$ equals the positive saturation value and $-100\%$ equals the negative saturation value, where the saturation value is taken from Table 3.2.

Two configurations of the helicopter and stand have been considered: a three DOF setup with the joints about $z_1$, $x_2$ and $z_2$ constrained and a six DOF experiment, without these constraints. At hover the stand affects mainly the mass, so the experimental configuration should principally reflect the behavior of a hovering helicopter.

The joint angles of the stand are instrumented with HEDS 5500-I06 optical encoders [34]. These encoders have 512 lines, corresponding to 2048 counts per revolution. With a sample period of 20 ms, the angular resolution of the sensor is $\theta_{res} \approx 0.18$ deg and the angular velocity saturates at $\omega_{max} \approx 1000$ deg/s. For high-fidelity rate measurements gyros can be easily incorporated into the experiment.

There is strong motivation for mounting the helicopter on a stand as opposed to flying untethered. First, by using the optical shaft encoders, the stand provides a convenient method of measuring the attitude of the helicopter, which is otherwise quite difficult to do accurately given the weight and size restrictions of the helicopter payload. Second, we are able to use the real-time computer without resorting to a telemetry system. Finally, it provides a safe setup for performing experiments in a laboratory. The disadvantages of the stand are that it limits the motion of the helicopter and changes its dynamics.

The tail rotor is directly geared to the main rotor with a gear ratio of about $1:4$, so that we can measure the rotor speed both at the tail rotor and at the main rotor. However, neither the tail rotor nor the main rotor leaves any room for an optical shaft encoder. Also, the shafts are subjected to large lateral and vertical vibrations due to the rotating blades, which are likely to damage an optical shaft encoder. Therefore a reflective object sensor mounted on the tail boom close to the tail rotor shaft provides the best alternative. Furthermore, the tail rotor rotates faster than the main rotor, providing better resolution of the rotor speed. Refer to Appendix 3.A for a detailed discussion of the rotor speed sensor.

### 3.2.1 Helicopter measurements and actuators

A summary of the helicopter measurements is shown in Table 3.3. The bandwidth of the measurements are limited by the sample period $T$ and the sensors.

| Measurement | Sensor |
|---|---|
| $\phi$ | Shaft encoder |
| $\theta$ | Shaft encoder |
| $\psi$ | Shaft encoder |
| $p$ | IIR filter |
| $q$ | IIR filter |
| $r$ | IIR filter |
| $\Omega_{tr}$ | Tachometer |

Table 3.3: Helicopter measurements and corresponding sensors.

A summary of the helicopter actuators is shown in Table 3.4. The bandwidth of the actuators are limited mainly by the PWM servos, which were empirically observed to be insensitive to commands above about 3 to 5 Hz.

| Control | Actuator |
|---|---|
| Aileron | PWM servo |
| Elevator | PWM servo |
| Rudder | PWM servo |
| Collective | PWM servo |
| Throttle | Speed controller |

Table 3.4: Helicopter controls and corresponding actuators.

## 3.3 Development of a linear hover model

This section covers the modelling and control of helicopters in hover. In particular, a hover model must be developed for the model helicopter mounted on the stand discussed in Section 3.2. The stand complicates the dynamics considerably so an identification rather than first principles approach was taken to determine a hover model for the helicopter. A detailed and comprehensive first principles analysis and model of a helicopter mounted on a stand was performed by Weilenmann [74, 72].

Linearization of standard nonlinear aerodynamical equations at hover provides a suitable state-space structure for modelling the helicopter. Johnson provides a classic treatment of nonlinear aerodynamical helicopter modelling [39]. Lewis *et al.* discuss the hybrid linear/nonlinear TMAN model used extensively at NASA for piloted simulations and war games [42, 43]. Schroeder *et al.* provide a specific example of a linear parametric transfer function approach to modelling an Apache helicopter [61]. Houston and Black consider identification of a hover model incorporating higher order rotor dynamics for a Puma helicopter [35]. Tischler discusses a mixed nonparametric frequency-domain and parametric state-space identification method for a BO-105 helicopter [69]. Fu and Kaletka consider higher order models explicitly incorporating the rotor dynamics [24]. The general conclusion of all of these efforts at modelling helicopters through identification is that it is possible to identify good models for helicopters when the identified model is based on parametric first principles analysis.

In this dissertation a simplified sixth order linear hover model is considered. According to Fu and Kaletka, a sixth order rigid body helicopter model can accurately model the low- and mid-frequency behavior [24]. In such a model, the effect of the rotor dynamics are absorbed into the parameters of the model and effective time delays for the control inputs. As will be seen in Chapters 6 and 8, robust control techniques can be used to account for unmodelled dynamics with model uncertainty above mid-frequency.

The states, $x(t)$, used in the linear model are

$$x(t) = \begin{bmatrix} \phi(t) \\ \theta(t) \\ \psi(t) \\ p(t) \\ q(t) \\ r(t) \end{bmatrix}. \tag{3.3}$$

These states are suitable for attitude control, but not inertial tracking of the helicopter. Because of the limitations of the stand, it was determined that inertial control was not feasible.

The inputs, $u(t)$, used in the linear model were

$$u(t) = \begin{bmatrix} \Theta_a(t) \\ \Theta_e(t) \\ \Theta_r(t) \end{bmatrix}. \tag{3.4}$$

At hover equilibrium, the state $x$ and the input $u$ can be described by

$$\begin{aligned} x(t) &= \bar{x} + \delta x(t) \\ u(t) &= \bar{u} + \delta u(t), \end{aligned} \tag{3.5}$$

where $\bar{x} = 0$, $\bar{u}$ is the non-zero trim command necessary to hold the helicopter at the hover equilibrium, and $\delta x(t)$ and $\delta u(t)$ are small time-varying perturbations around hover.

The principal assumptions used to develop the parametric state-space model were: (*i*) at hover the altitude is constant, eliminating nonlinearities due to ground effect, and (*ii*) the rotor speed is kept constant, minimizing the rotor nonlinearities. Under these assumptions, the most simple linear decoupled model of the helicopter near hover is

$$\begin{aligned}
\dot{\phi} &\simeq p \\
\dot{\theta} &\simeq q \\
\dot{\psi} &\simeq r \\
\dot{p} &\simeq L_\phi \phi + L_p p + L_{\Theta_a} \delta\Theta_a \\
\dot{q} &\simeq M_\theta \theta + M_q q + M_{\Theta_e} \delta\Theta_e \\
\dot{r} &\simeq N_r r + N_{\Theta_r} \delta\Theta_r,
\end{aligned} \tag{3.6}$$

where $L_\phi$, $L_p$, $L_{\Theta_a}$, $M_\theta$, $M_q$, $M_{\Theta_e}$, $N_r$, $N_{\Theta_r}$, are aerodynamical coefficients, typically computed through wind tunnel testing as a function of different quasi-static operating points. Note that an actual helicopter is not truly decoupled at hover. Some cross-coupling terms must be taken into account, as will be seen in Chapter 4.

To control the helicopter during vertical flight the dynamics of the helicopter body velocities ($u$, $v$, $w$) must be added. Suitable dynamical models of $u$, $v$, and $w$ are contained [42, 43, 39, 72]. Once dynamics of $u$, $v$, and $w$ are included in the helicopter model, the helicopter altitude $z$ results from the nonlinear kinematic transformation (3.1). Note that the rotor dynamics couple into the dynamics of $w$. This is described in great detail by Johnson [39] and Houston *et al.* [35].

## 3.4 Summary

In this chapter standard nonlinear helicopter modelling techniques were reviewed. It is clear that the principal nonlinearities in helicopters are the rotor dynamics, ground effect, and structural flexure.

A reduction to a linear hover model was made, which involved several simplifications and assumptions. The major assumption, to minimize the main rotor gyroscopic forces and nonlinearity, was that the helicopter be flown at constant altitude with constant rotor speed. Flying at constant altitude has the additional benefit of eliminating ground effect.

An analytic model of the effect of the stand on the helicopter was not considered; the experimental helicopter model, developed in Chapter 4 is based on parametric state-space models using identification techniques with real experimental data. Thus, the effect of the stand will be implicitly contained in the estimated parameters. Similarly, it can be shown that in the low- to mid-frequency range, the principal effect of the rotor dynamics can be absorbed into the low order model parameters and control input delays.

In order to continue work on the helicopter, new sensors will need to be integrated with the experimental platform, including a tachometer to measure the rotor speed, rate gyros for explicit measurement of the body angular rates, an electronic compass to measure yaw in free flight, accelerometers for inertial and rate measurements, and charge-coupled devices (CCDs) for imaging. These would allow the helicopter to be flown in free flight with suitable telemetry or on-board real-time computing systems.

## 3.A Appendix: tachometer sensor

As discussed previously, varying rotor speed on a helicopter results in substantial nonlinearities. To minimize these effects it is helpful to design a single-input/single-output (SISO) controller which regulates the actual rotor speed, using throttle to provide rotor speed commands. To design such a controller requires the development of a sensor for measuring the rotational velocity of the main rotor. Such a sensor was first described by Van Nieuwstadt and Morris [70].

Measurement of the tail rotor speed is equivalent to measurement of the main rotor speed, because the tail rotor is directly geared to the main rotor. Designing a sensor which measures the tail rotor speed offers many benefits. The principal benefit is avoiding any mechanical connection to the main rotor hub.

The tail rotor speed sensor is designed around a reflective object sensor, the Optek OPB 745. The Optek OPB 745 consists of a light emitting diode (LED), which emits a focused beam of infrared light, and a darlington photo transistor, which is sensitive to infrared light. The light reflects on the tail rotor blades each time they pass by, and is received by the photo transistor. Aluminum foil was applied to the side of the rotor blades facing the sensor, to increase reflectivity. The output signal of the reflective object sensor is a pulse whose width is directly proportional to the width of the reflective surface, and inversely proportional to the rotor speed. Since the distance between the sensor and the rotor blades changes when the blade pitch is changed, the amplitude of this pulse varies over a range, from 1 V to 5 V. At the tail rotor this range is smaller than at the main rotor, which is another advantage of mounting the sensor at the tail rotor. To make the sensor less sensitive to the varying amplitude of the pulse, the output of the photo transistor is amplified with high gain, then Schmitt triggered and inverted. Figure 3.5 shows a schematic of the sensor circuit. The capacitor $C_1$ after the Schmitt trigger is necessary to suppress

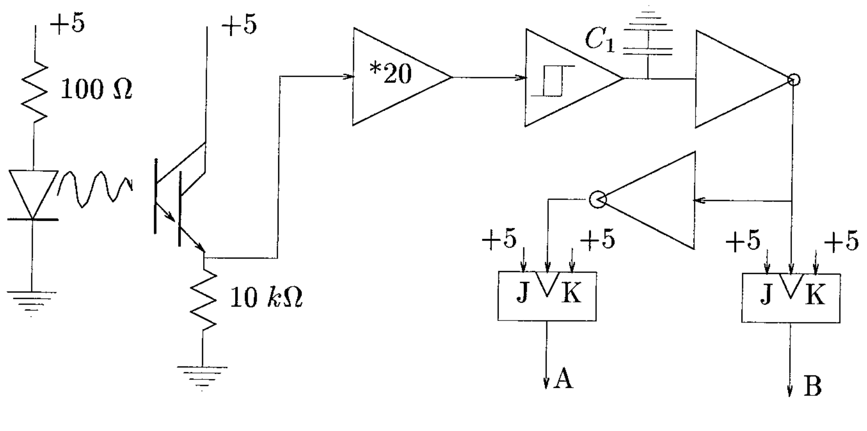

Figure 3.5: Rotor speed sensor circuit. A and B are quadrature signals.

spurious edges. Its value is about 1.7 nF. Both the inverted and the non-inverted signal are input to a J-K flip-flop to generate a quadrature signal that has a phase difference equal to the width of the sensed pulse. The quadrature signal is decoded with a quadrature decoder which registers 4 counts per revolution, 1 each for each rising and falling edge of the 2 quadrature signals generated when a tail rotor blade passes the sensor. The quadrature decoder is directly read by the real-time computer system, as outlined in Chapter 2.

The rotor tachometer is limited by the sample period; the minimum measurable angular velocity is given by $\frac{1}{LT}$ revolutions per second (RPS) and the maximum measurable angular velocity is $\frac{127}{2LT}$, where the sample period is $T$ and the number of rotor blades is $L$. For the helicopter $L$ is 2, so the minimum measurable rotor speed is 25 RPS and the maximum is 1587.5 RPS, for a sample period of 20 ms. As this is not adequate resolution for measuring the rotor speed of the helicopter a filter will be employed to enhance this resolution by averaging measurements over larger periods of time than the 20 ms sample period. This is discussed further in Appendix 4.A.

The sensor itself has a mass of 10 grams, and the mass of the signal conditioning circuit is about 25 grams. This is well within the payload of the model helicopter. Although a slotted optical switch or a slit wheel is more reliable mechanically and offers a higher resolution, it requires major mechanical reassembly of essential parts, so even if there were room for a slit wheel, it would remain preferable to use the sensor described herein. The reflective object sensor is non-invasive and easily applied.

# Chapter 4

# Nominal Hover Model Identification

System identification is a necessary part of any model-based control design. The objective of this chapter is to identify a nominal linear time-invariant (LTI) model of the helicopter operating near hover based on the structure developed in Chapter 3. Because the helicopter has significant cross-couplings, MIMO identification methods are essential.

Frequency-domain parameter identification was successfully employed to identify a model for the Apache helicopter by Schroeder *et al.* [61], by Houston and Black for a Puma helicopter [35], and by Fu and Kaletka for a BO-105 helicopter [24]. Tischler further studied identification requirements for rotorcraft [69]. Weilenmann developed a mixed analytic/parameter identification approach [72].

Section 4.1 discusses the experimental techniques used for identification of the RC helicopter. State-space MIMO identification techniques were employed because MIMO frequency-domain identification techniques tend to introduce unnecessary states. The prediction error method (PEM), a general method for identifying MIMO systems, is discussed in Section 4.2. A rigid body model of the helicopter, which ignores rotor dynamics, is identified in Section 4.3 using PEM techniques. A preliminary model of the main rotor is presented in Appendix 4.A.

## 4.1 Identification experiments

From the identification point of view, a sequence of excitations at each input of the plant is defined to make it possible to identify the dominant modes and cross-couplings. Ideally we would like to be able to drive the helicopter with a sufficiently rich input to excite all the dominant modes and cross-couplings. There are two major obstacles to this. First, the helicopter is open-loop unstable, so a stabilizing "trim" command is necessary when taking data. Second, the helicopter is nonlinear so we can only expect to get reasonable results when we use small-signal excitations around the trim. This restricts the types of excitation that we can use to be of the form

$$u(t) = \bar{u}(t) + \delta u(t), \tag{4.1}$$

where $\bar{u}(t)$ is the manual pilot trim command which maintains the helicopter bounded-input bounded-output stable near hover and $\delta u(t)$ is a small-signal excitation. If we assume that $\bar{u}(t)$ varies slowly and $\delta u(t)$ is of small enough magnitude then we can model the helicopter about the trim $\bar{u}(t)$ as an LTI system.

Due to the complexity of the helicopter, an LTI model cannot be expected to completely capture the dynamical behavior of the helicopter. Identification methods which produce good estimates of the modelling error and which allow *a priori* information to be included in the estimation algorithm greatly simplify the task of finding a reasonable uncertainty description for covering the experimental data.

The identification experiments were conducted with a 50 Hz sample rate and consisted of finding a model of the helicopter using the aileron, elevator and rudder inputs, and measurements of the attitude, $\phi$, $\theta$, and $\psi$. The RC transmitter was used to provide the trim necessary to keep the helicopter operating near hover. In particular, it was necessary to use the throttle and collective to maintain constant rotor speed and thrust. This is very important since the actuator model is highly dependent on the thrust of the main rotor, and variations in the rotor speed cause gyroscopic effects which increase the nonlinear behavior of the helicopter. A small-signal excitation was superimposed on top of the aileron, elevator, and rudder trims.

The trim command introduces feedback into the system which correlates output noise with the control inputs. Also, the trim command is slowly-varying. Both of these effects result in a statistical bias of the parameter estimates which was ameliorated through high-pass filtering of the experimental data.

## 4.2 Identification using PEM

The prediction error method is a MIMO estimation algorithm which seeks to minimize the quadratic error between the predicted value of the plant and the data. Standard references on these techniques are Ljung [44] and Söderström and Stoica [64]. A state-space discrete-time version of the PEM algorithm developed by Ljung is used [45]. The PEM algorithm allows for the incorporation of structure and known parameters into the model.

We begin with a state-space model of the helicopter dynamics linearized near hover. This model structure is based on the development in Chapter 3. It consists of a sixth order model of the helicopter and a model for the actuators. The actuator model can be used to account for the effect of rotor dynamics.

The model helicopter near hover is described by an LTI system given in discrete-time by

$$\begin{aligned} x_1(k+1) &= x_1(k) + Tx_2(k) + Hx_2(k) \\ x_2(k+1) &= A_{21}x_1(k) + A_{22}x_2(k) + Bv(k) \\ y(k) &= x_1(k), \end{aligned} \tag{4.2}$$

with states $x_1 = (\phi,\, \theta,\, \psi)^T$ and $x_2 = (p,\, q,\, r)^T$, control inputs $u = (\Theta_a,\, \Theta_e,\, \Theta_r)^T$, and measurement $y$. $T$ is the sampling period and $v$ is the output of the actuator model. The structure of the actuator model to be estimated is defined by

$$\begin{aligned} z(k+1) &= Fz(k) + Gu(k-d+1) \\ v(k) &= z(k), \end{aligned} \tag{4.3}$$

where $z$ are the states of the actuator model, $u$ is the pilot command, and $v$ is the actuator response. $F$ and $G$ are diagonal matrices and $d$ is the delay, expressed in number of samples, introduced by the actuators. Based on preliminary data from a six DOF force-torque sensor, the actuator model should be block diagonal, with each actuator modelled as a first (or second) order transfer function. However, in order to simplify identification and minimize parameters, the actuators are modelled as pure delay. Note that in (4.2) $x_2$ is roughly the forward difference approximation of the time derivative of $x_1$ (3.6).

The parameters of the state-space realization consist of the elements of the $A_{21}$, $A_{22}$, and $B$ matrices. $H$ corresponds to an extra term used to provide hysteresis in the integrators. The choice of the parameters to be estimated is based on their identifiability. It was found that the parameters in $A_{21}$ had very little effect on the estimation errors, and in addition were not very repeatable on different experiments. The diagonal components of $A_{22}$ and $B$ were very repeatable on different experiments. Note that the most significant cross-couplings are introduced by the yaw rate. This leads to the parametric structure

$$\begin{aligned} A_{21} &= \begin{pmatrix} a_{11}^{21} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad A_{22} = \begin{pmatrix} a_{11}^{22} & 0 & a_{13}^{22} \\ a_{21}^{22} & a_{22}^{22} & a_{23}^{22} \\ a_{31}^{22} & 0 & a_{33}^{22} \end{pmatrix} \\ B &= \begin{pmatrix} b_{11} & 0 & 0 \\ 0 & b_{22} & 0 \\ 0 & 0 & b_{33} \end{pmatrix}, \quad H = \begin{pmatrix} 0 & h & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \end{aligned} \tag{4.4}$$

where the actuators are modelled by pure delay, $F = 0$ and $G = I$ in (4.3).

As the structure of the model is simple, it is not reasonable to identify the system over all frequencies. So, the identification is focused on the relevant frequency band necessary for hover control. In particular, the experimental data are preprocessed to reduce the effects of the manual trim which was necessary to maintain stable operation near hover during the identification experiments. Once a data range is selected, the initial condition is removed and the data is passed through a fourth order high-pass Butterworth filter, $F$, with cut-off above 0.3 Hz, as depicted in Figure 4.1. The overall system in (4.2) can also be modelled by the transfer function, $y(t) = G(\Theta, q)u(t)$, where $\Theta$ denotes the set of parameters to be identified and $q$ is the standard forward shift operator. Given a transfer function description $G$ properly parametrized by the specific form in (4.2) and (4.4) and the filtered input–output data, $u_f$ and $y_f$, the prediction error $e(t) = y_f(t) - G(\Theta, q)u_f(t)$ can be computed.

For multi-output systems, the identification method consists in determining the

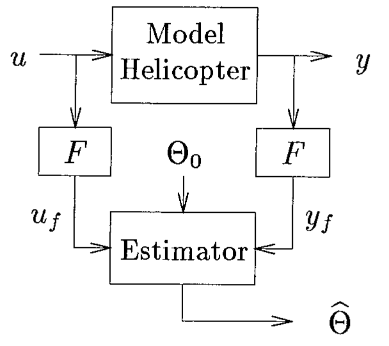

Figure 4.1: Identification schema. $F$ represent filters used to pre-process data prior to identification.

parameter estimates by minimizing the quadratic criterion

$$\hat{\Theta} = \arg \min_{\Theta} \det \left[ \frac{1}{N} \sum_{t=1}^{N} e(t,\Theta) e^T(t,\Theta) \right] \tag{4.5}$$

using an iterative Gauss-Newton algorithm [44], where arg returns a value of $\Theta$ achieving the minimum.

The identified models will be denoted by $\hat{G}_3$ and $\hat{G}_6$, referring to the three DOF and six DOF configurations of the helicopter experiment, respectively. Note that there is no difference in the structure of these models, they were just obtained with different configurations of the stand. The purpose for considering both $\hat{G}_3$ and $\hat{G}_6$ was to verify that the principal dynamics during hover were unchanged when the helicopter was restricted to three DOF.

Providing the algorithm with appropriate initial parameter values $\Theta_0$ (Figure 4.1) speeds the convergence of the estimation algorithm [44]. This becomes critical when dealing with long data records, since for each iteration the estimation algorithm computes the inverse of a matrix which can be ill-conditioned without a good initial guess. Moreover, when there are many parameters, long data records are required for convergence of the estimator. These initial values can be obtained from physical considerations and also from preliminary SISO identification on shorter data records.

## 4.3 Nominal LTI helicopter hover model

As the helicopter body rates could not be directly measured, limitations on the accuracy of the identified model are expected. The final values of the estimated parameters for $\hat{G}_3$ are given by

$$A_{21} = \begin{pmatrix} -0.0320 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad A_{22} = \begin{pmatrix} 0.5432 & 0 & -0.0140 \\ 0.0404 & 0.6135 & 0.0118 \\ -0.0105 & 0 & 0.9661 \end{pmatrix}$$

$$B = \begin{pmatrix} 1170 & 0 & 0 \\ 0 & -1238 & 0 \\ 0 & 0 & -1446 \end{pmatrix}, \quad H = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -0.0022 \\ 0 & 0 & 0 \end{pmatrix}. \qquad (4.6)$$

The parametric uncertainty from the estimated model (4.6) is tabulated in Table 4.1. The percentage uncertainty in Table 4.1 corresponds to the ratio of the standard deviation of the parameter and the parameter value. By comparing the variations in estimated parameter values of the model defined by (4.2) and (4.3) with different parameter structures and on different experiments an indication of the amount of parametric uncertainty and unmodelled dynamics in the model was obtained. This information is directly applicable to constructing a robust model.

| Parameter | Parameter Value | Percentage Uncertainty |
|---|---|---|
| $a_{11}^{21}$ | -0.032 | 27 |
| $a_{11}^{22}$ | 0.5432 | 3 |
| $a_{13}^{22}$ | -0.0140 | 8 |
| $a_{21}^{22}$ | 0.0404 | 5 |
| $a_{22}^{22}$ | 0.6135 | 2 |
| $a_{23}^{22}$ | 0.0118 | 7 |
| $a_{31}^{22}$ | -0.0105 | 9 |
| $a_{33}^{22}$ | 0.966 | 0.04 |
| $b_{11}$ | 1170 | 4 |
| $b_{22}$ | -1238 | 3 |
| $b_{33}$ | -1446 | 0.8 |

Table 4.1: Parametric estimation error in the helicopter model. The percentage uncertainty is the ratio of the standard deviation of the parameter with the parameter value.

Note that $H$ was defined *a priori* as a constant using precomputed rate data and $\hat{G}_6$ was identified in a similar manner to $\hat{G}_3$ but with six DOF helicopter data. It can be seen from (4.6) and Table 4.1 that the principal axes are well estimated, with much larger uncertainty in the cross-couplings. The cross-couplings in $B$ were hard to estimate but were present in the experimental data. This motivates the choice of full block multiplicative input uncertainty, as will be seen in Section 6.2. The parameters for $\hat{G}_6$ are not presented.

The simulated time-domain responses of $\hat{G}_3$ (dashed) are plotted against the experimental data (solid) which was used for identification in Figures 4.2 through 4.4. The simulation was performed using the same input used for identification.

The simulation matches the experimental data very well on pitch and more particularly on roll. The results are not as good on yaw as an asymmetry in the actuation has not been taken into account.

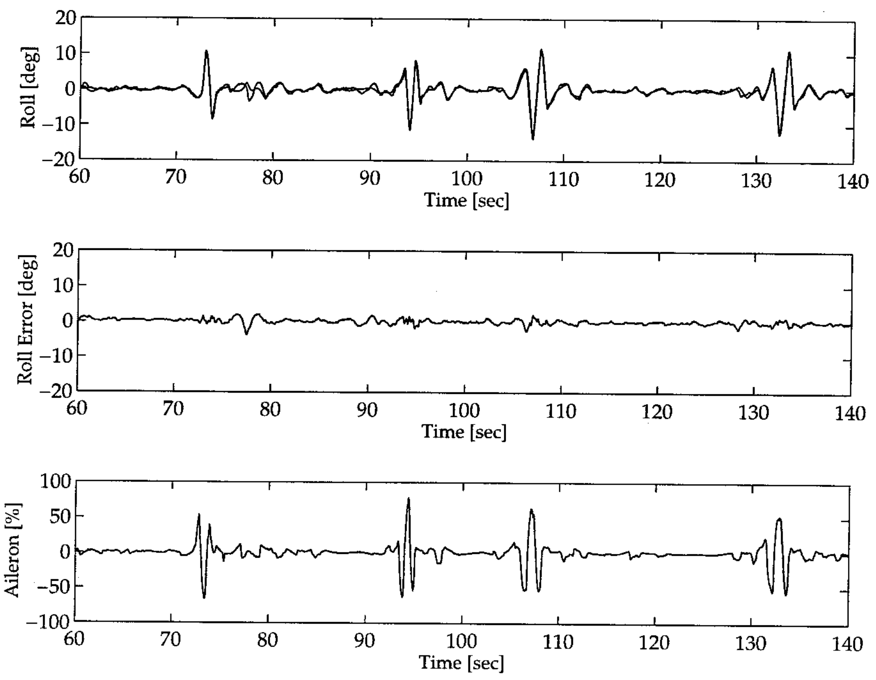

Figure 4.2: Helicopter roll axis. Top: experimental data (solid), $\hat{G}_3$ simulation (dashed). Middle: simulation error. Bottom: experimental input.

Finally, to check the ability of the identified model to predict the behavior of the model helicopter in hover, the model was simulated on a different data set from the one with which it was estimated. Such a procedure was successful for both three DOF and six DOF experiments; this verified the specific form used for the parametrization. See Chapter 8 for the frequency response of each channel in the helicopter model.

The impulse-responses of $\hat{G}_3$ (solid) and $\hat{G}_6$ (dashed) are shown in Figure 4.5. The responses of the principal axes are comparable, with the exception of the pitch axis. The cross-couplings are also comparable, with the exception of rudder–roll. The observed differences between $\hat{G}_3$ and $\hat{G}_6$ on the aileron–pitch and rudder–roll transfer functions result primarily from change in dynamics that results when clamping the motion of the helicopter to three DOF.

The major problem found with the PEM routine was the inability to process long data records, particularly when there were many parameters in the model. This results in a decrease in the achievable bandwidth of the model, since we are forced to use shorter data records. This bandwidth limitation makes it much more difficult to obtain good results with continuous-time robust control synthesis techniques.

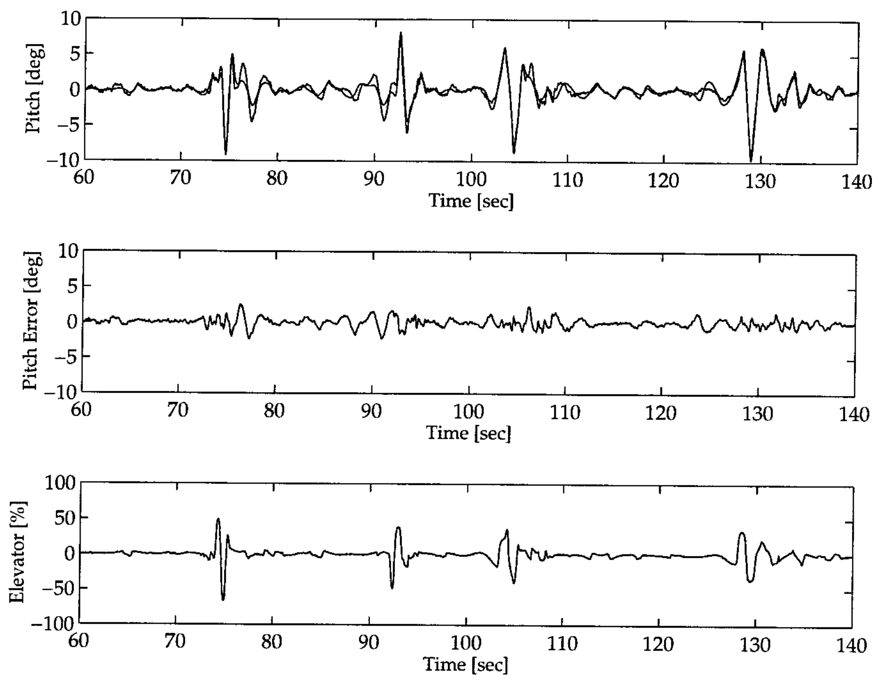

Figure 4.3: Helicopter pitch axis. Top: experimental data (solid), $\hat{G}_3$ simulation (dashed). Middle: simulation error. Bottom: experimental input.

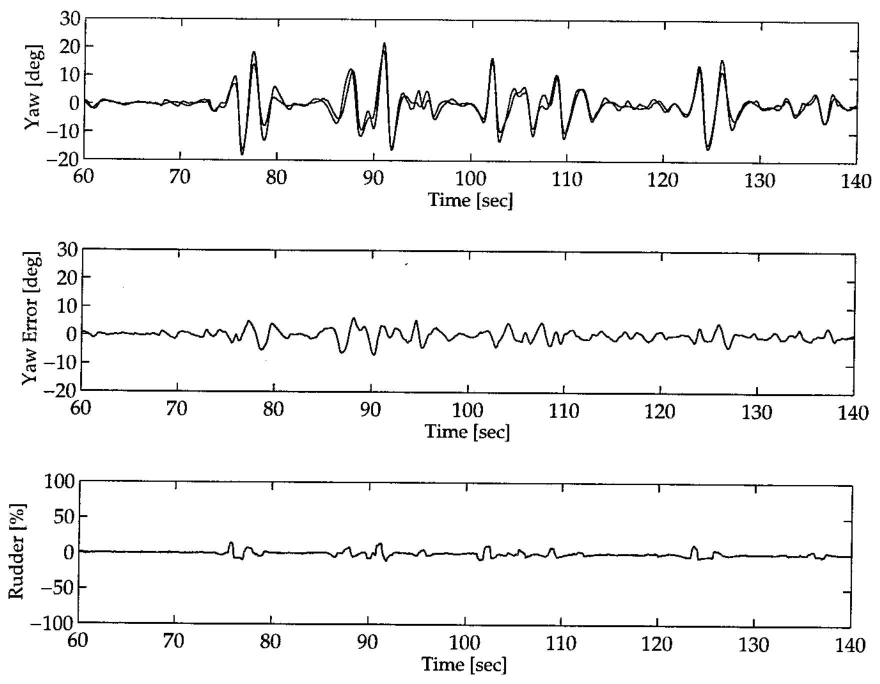

Figure 4.4: Helicopter yaw axis. Top: experimental data (solid), $\hat{G}_3$ simulation (dashed). Middle: simulation error. Bottom: experimental input.

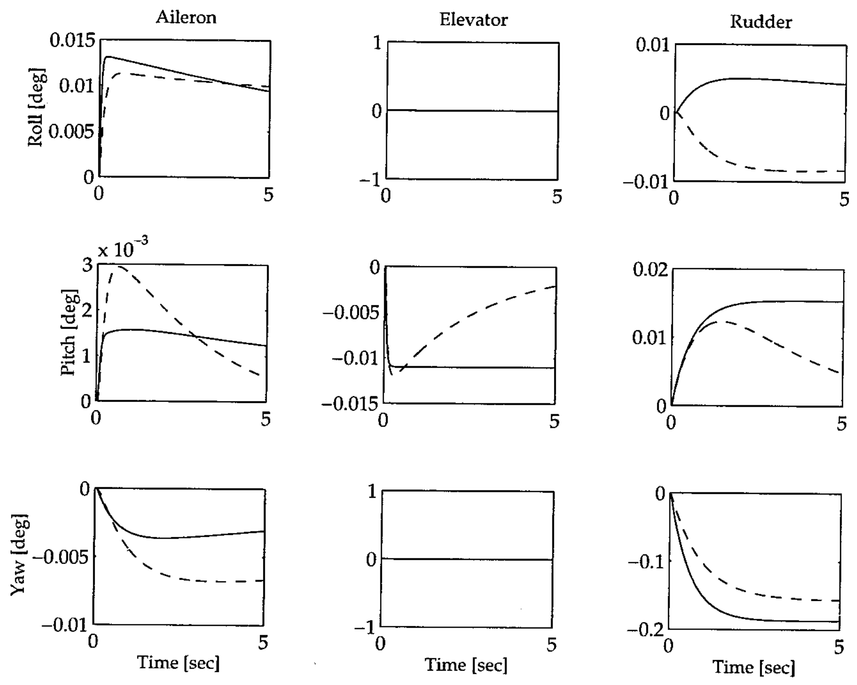

Figure 4.5: Impulse-responses of $\hat{G}_3$ (solid) and $\hat{G}_6$ (dashed).

## 4.4 Summary

In this chapter standard state-space linear estimation techniques were employed to identify a MIMO LTI model for the helicopter operating near hover. State-space techniques offer a clear advantage over frequency-domain techniques when the parametric structure of the dynamics are well understood, as was the case with the helicopter. Also, when identifying MIMO systems, state-space techniques often result in a realization with fewer states. The principal advantage of frequency-domain techniques, which is lost for state-space techniques, is the ability to work with long data records and the use of averaging to minimize noise. This issue manifested itself when using the PEM estimation technique: choosing good initial conditions for the estimator was essential to insure convergence of the estimator when there were either many parameters or long data records.

The identified helicopter hover model accounted for the experimental data very well in a predictive nature. However there were several major problems: during normal operation near hover there is significant parameter variation resulting from unmodelled dynamics and forces (such a rotor dynamics, structural flexure, and ground effect), the actuators are not truly decoupled (the linear model is), actuator saturation is not taken into account, and the rotor dynamics are ignored. It will be essential to incorporate the principal effects of these unmodelled nonlinearities and dynamics into a robust model so that a robust hover controller can be designed.

Further work will be necessary to develop a vertical motion model for the helicopter. This involves regulating the main rotor speed so that the collective can be used to control the vertical motion, requiring the rigid body hover and SISO main rotor models to be combined into a single model. A model suitable for free flight and inertial control could then be developed by adding the nonlinear inertial kinematics discussed in Chapter 3 and scheduling the helicopter over relevant operating regions.

## 4.A Appendix: SISO main rotor model

The motivation for developing a model for the main rotor of the helicopter is so that an inner loop controller can be developed which eliminates or minimizes problems resulting from rotor flapping and gyroscopic effects by regulating the rotor rotational velocity. As discussed in Chapter 3, complex dynamics resulting from rotor flapping and nonlinear gyroscopic effects primarily result from variations in the main rotor speed. Thus, by carefully regulating the main rotor speed these problems can be mitigated, without requiring a detailed model of main rotor. Furthermore, in a linearized hover model the actuator derivative coefficients (the $B$ matrix) are directly proportional to the square of the rotor speed. The SISO model for the main rotor was developed by Van Nieuwstadt and Morris [70].

The rotor speed is directly controlled with a PWM signal driving a speed controller that regulates the current through a DC motor. This PWM signal is called the throttle. The speed controller insures that the current through the DC motor is kept constant; however, varying loads on the helicopter will change the rotor speed even if the current is kept constant. To guarantee that the rotor speed is held constant, the transfer function from the throttle to the rotor speed must be modelled, so that a regulator can be designed.

The raw output signal of the rotor speed sensor is the number of times per sample period that a rotor blade passes the sensor, and is therefore intrinsically discrete. At a sampling rate of 100 Hz, the sensor would typically register between 4 and 5 counts per sample period. This discrete signal was passed through a second order low-pass Butterworth filter with a cut-off frequency of 0.4 times the sampling frequency, $T = 100$ Hz, to obtain a continuous valued signal.

Referring to Figure 4.6, it is clear that the steady state rotor speed is a nonlinear function of the throttle command. The sensitivity of the rotor speed to throttle decreases at higher rotor speed. This nonlinearity was accounted for by an empirically computed static nonlinear transformation, $G$, operating on the experimental data. For identification purposes the experimental data was preprocessed by $G$. The resulting data was then used to identify a linear plant $P$, which represents the linearized dynamics of the main rotor.

The operating range of interest for hover typically corresponds to a nominal rotor speed of 80 tail rotor rotations per second, or 20 main rotor rotations per second. A second order linear model was estimated at this operating point using auto-regressive software in the MATLAB System Identification Toolbox [44, 45, 64].

The estimated model had a time delay of 32 samples at 100 Hz. This delay was modelled with a first order Padé approximation. The estimated model, augmented with the Padé approximation, is given by the third order SISO model

$$\begin{aligned} x_{k+1} &= Ax_k + Bu_k \\ y_k &= Cx_k + Du_k, \end{aligned} \tag{4.7}$$

where $u_k$ is the steady state rotor speed in rotations per second and $y_k$ is the rotor speed in rotations per second. The specific values for the model are

$$\left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] = \left[\begin{array}{ccc|c} 1.9015 & 1.0 & 0.0002 & -0.0019 \\ -0.9030 & 0 & -0.0001 & 0.0007 \\ 0 & 0 & 0.9394 & 0.9697 \\ \hline 1.0 & 0 & 0.0001 & -0.0007 \end{array}\right]. \tag{4.8}$$

Figure 4.6 shows the experimental data (solid) and a simulation (dashed) of the estimated model. The difference in steady state gain can be clearly seen. The magnitude of the throttle command was equal in negative and positive directions, whereas the resulting change in rotor speed is much smaller in the positive direction.

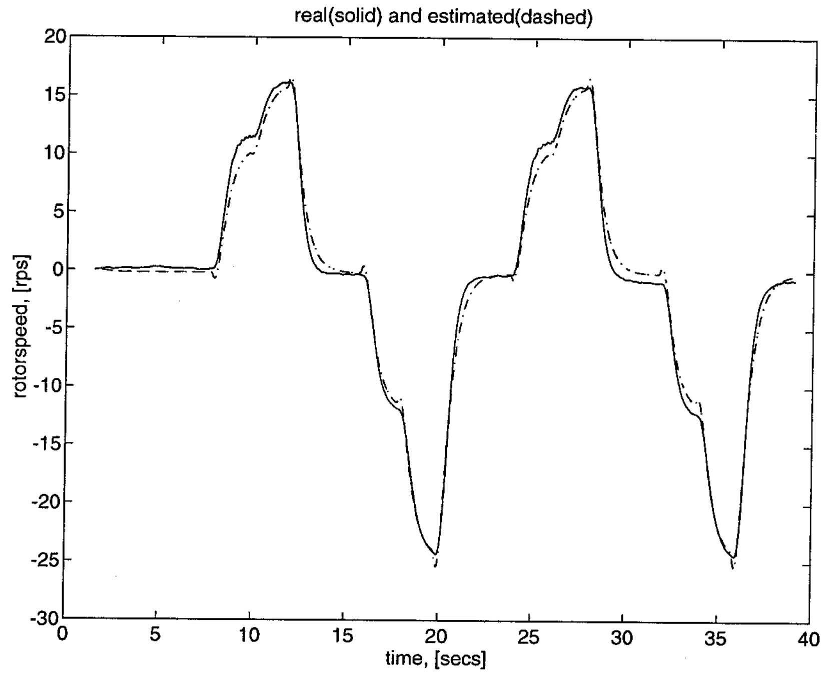

Figure 4.6: Simulation of the main rotor model. Rotorspeed is shown as the differential value from the nominal of 80 RPS. Experimental data are solid, simulated data are dashed. The magnitude of the throttle command was equal in both directions.

# Chapter 5
# Review of Robust Control

The principles of frequency-domain robust control lie with the seminal work of Zames [77, 78]. The early work in $\mathcal{H}_\infty$control provided the impetus to study the incorporation of structured uncertainty into the control problem, leading to the development of $\mu$ theory by Doyle [16]. A major innovation came with state-space solutions to the standard $\mathcal{H}_\infty$ problem by Doyle *et al.* [19], with a computational solution by Glover and Doyle [28]. An introduction to SISO robust control is presented by Doyle *et al.* [18]. A comprehensive treatment of MIMO robust control is provided by Zhou *et al.* [79].

More recently, linear matrix inequalities (LMIs) have been shown to form a unifying framework for modern control. A review of this subject is presented by Doyle *et al.* [21]. Packard *et al.* provide examples of several important problems in robust control which can be reduced to LMIs [58]. LMIs form an attractive framework for studying problems because they are convex, readily computable in polynomial time, and there are software packages currently available for computing solutions. Beck provides a review of the computational issues for solving LMIs [7].

Section 5.1 introduces standard notation used in frequency-domain robust control theory. A review of mixed $\mu$-analysis and $\mu$-synthesis theory is presented in Sections 5.2 and 5.3, respectively. Finally, an introduction to LMIs is provided in Section 5.4.

## 5.1 Notation

The concept of linear fractional transformations (LFTs) is frequently found in control theory. There are many applications of LFTs in control theory. See [21] for a comprehensive review. The LFT is defined on systems of the form

$$\begin{aligned} P(s) &= \left[\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right] \\ &= C(sI-A)^{-1}B + D, \end{aligned} \tag{5.1}$$

which denotes the system defined by the state-space equations

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du. \end{aligned}$$

A common use of (5.1) is for systems in which the coefficients, $A$, $B$, $C$, and $D$, have a partitioned structure. One such structure is a MIMO transfer system defined by

$$\begin{aligned} P(s) &= \left[\begin{array}{c|cc} A & B_1 & B_2 \\ \hline C_1 & D_{11} & D_{12} \\ C_2 & D_{21} & D_{22} \end{array}\right] \\ &= \begin{bmatrix} P_{11}(s) & P_{12}(s) \\ P_{21}(s) & P_{22}(s) \end{bmatrix}. \end{aligned} \tag{5.2}$$

The LFT is defined on (5.2), and can be thought of as closing loops with controllers or uncertainty perturbations. There are two general ways to close loops: either on the upper or lower block of (5.2). These transformations are defined as

$$\mathcal{F}_{\mathrm{u}}(\Delta(s), P(s)) = \Delta(s) \star P(s) \ = P_{22} + P_{21}\Delta(I - P_{11}\Delta)^{-1}P_{12} \tag{5.3}$$

$$\mathcal{F}_{\ell}(P(s), K(s)) = P(s) \star K(s) = P_{11} + P_{12}K(I - P_{22}K)^{-1}P_{21}, \tag{5.4}$$

where it is assumed that $P(s)$ is partitioned appropriately. The existence of the inverses is a necessary condition for (5.3) and (5.4) to be well defined; hence, we will always assume that the inverses exist whenever an LFT is considered. We will also assume that the functional dependence on $s$ is implicit when defining LFTs. Note that the $\star$ notation is more general than $\mathcal{F}_{\mathrm{u}}(\cdot,\cdot)$ and $\mathcal{F}_{\ell}(\cdot,\cdot)$. Newlin provides a more detailed discussion of LFTs and $\star$ [50]. Figure 5.1 illustrates the LFT $\mathcal{F}_{\ell}(P, K)$ which closes the loop on a plant, $P$, with a controller, $K$.

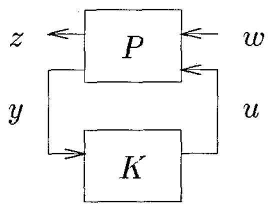

Figure 5.1: Feedback interconnection of $P$ and $K$.

In robust control, perturbations or uncertain operators are used to define sets of systems. Structured uncertainties were introduced by Doyle [16] and Safanov [60]. See [53, 75] for extensions to the $\mu$ robust control framework. As will be seen in subsequent sections, $\mu$ is dependent on the underlying block structure. In the $\mu$ framework, uncertainties are defined to be block diagonal. There are several types of blocks which can be allowed: complex full blocks, complex repeated scalars and real repeated scalars. Notation for representing the block structure will be borrowed from [75]. Let $m_C$ be the number of complex full blocks, $m_c$ the number of complex repeated scalar blocks, and $m_r$ the number of real repeated scalar blocks. Then define the block structure to be the $m$-tuple $\mathcal{K}(m_r, m_c, m_C)$ given by

$$\mathcal{K} = \{k_1, \ldots, k_{m_r}, k_{m_r+1}, \ldots, k_{m_r+m_c}, k_{m_r+m_c+1}, \ldots, k_{m_r+m_c+m_C}\}\,. \tag{5.5}$$

Referring to (5.5), $\mathcal{K}(m_r, m_c, m_C)$ defines the dimensions of the blocks in a perturbation set, which is then defined by

$$\begin{aligned}\boldsymbol{\Delta} = \{\Delta : \Delta = \; & \mathrm{diag}(\delta_1^r I_{k_1}, ..., \delta_{m_r}^r I_{k_{m_r}}, \delta_1^c I_{k_{m_r+1}}, ..., \delta_{m_c}^c I_{k_{m_r+m_c}}, \\ & \Delta_1^C, ..., \Delta_{m_C}^C), \delta_i^r \in \mathbb{R}, \delta_i^c \in \mathbb{C}, \\ & \Delta_i^C \in \mathbb{C}^{k_{m_r+m_c+i} \times k_{m_r+m_c+i}}\}. \end{aligned} \tag{5.6}$$

Note that (5.6) defines the perturbation as block diagonal, where each block is square and there is a specific ordering of blocks. While this may seem restrictive, all results presented in this chapter are easily generalized to the case where the complex full blocks are non-square and the order of blocks in the set $\boldsymbol{\Delta}$ is unimportant. These assumptions are made purely for ease of presentation.

In many circumstances it is necessary to restrict the size of an operator or uncertainty to lie within some norm-bounded set. Let $\boldsymbol{\Delta}$ define a set of operators. Then define the norm-bounded structured uncertainty set by

$$\boldsymbol{B\Delta} = \{\Delta \in \boldsymbol{\Delta} : \|\Delta\|_\infty \leq 1\},$$

which serves to restrict the allowable uncertainty to be of size 1.

## 5.2 Introduction to mixed $\mu$-analysis

In this section we will briefly review methods for analyzing the stability and performance properties of interconnected systems subject to norm-bounded structured uncertainty. Any linear interconnection of inputs, outputs, and uncertainty perturbations can be rearranged to fit the interconnection structure of Figure 5.2 [16, 20, 17, 21, 55, 71]. $M$ describes a plant along with all the weighting functions on the

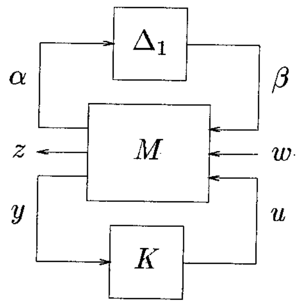

Figure 5.2: General interconnection with structured uncertainty.

inputs and outputs used to scale the norm-bounds to 1. $K$ is a controller mapping the measurements, $y$, to the plant control inputs, $u$. $\Delta_1$ is a norm-bounded structured uncertainty perturbation, and the mapping $\beta \rightarrow \alpha$ characterizes the uncertainty model being used. The mapping $w \rightarrow z$ characterizes the (desired) performance, where $w$ and $z$ will be required to satisfy $\|w\|_2 < \infty$ and $\|z\|_2 < \infty$. Mixed $\mu$ will be used to refer to problems which have both real and complex structured uncertainty. Complex $\mu$ refers to problems with only complex structured uncertainty.

There are a few important properties of the closed-loop system worth considering: *nominal performance*, *robust stability*, and *robust performance*.

The first property is concerned only with the performance properties of the nominal closed-loop system $\mathcal{F}_\ell(\mathcal{F}_u(0,M),K)$, and is defined as nominal performance.

**Definition 5.2.1 (Nominal Performance)** *A closed-loop system is said to have nominal performance if a controller, $K$, stabilizes the system $\mathcal{F}_u(0,M)$ and further satisfies the performance objective $\|\mathcal{F}_\ell(\mathcal{F}_u(0,M),K)\|_\infty < 1$.*

The second property is concerned with the stability of the perturbed closed-loop system and is defined as robust stability [17].

**Definition 5.2.2 (Robust Stability)** *A closed-loop system is said to have robust stability if a controller, $K$, stabilizes the set of plants*

$$\{P : P = \mathcal{F}_u(\Delta_1, M), \forall \Delta_1 \in \boldsymbol{B\Delta_1}\}.$$

The final property is concerned with the stability and performance of the perturbed closed-loop system, and is defined as robust performance [16, 17].

**Definition 5.2.3 (Robust Performance)** *A closed-loop system is said to have robust performance if a controller, $K$, stabilizes the set of plants*

$$\{P : P = \mathcal{F}_u(\Delta_1, M), \forall \Delta_1 \in \boldsymbol{B\Delta_1}\}$$

*and further satisfies the performance objective*

$$\|\mathcal{F}_\ell(\mathcal{F}_u(\Delta_1, M), K)\|_\infty < 1, \ \forall \Delta_1 \in \boldsymbol{B\Delta_1}.$$

A matrix function $\mu$ will now be discussed which can be used to analyze the stability and performance properties of the interconnection structure in Figure 5.2.

Define $P = \mathcal{F}_\ell(M,K)$. Assume that all of the matrices in Figure 5.2 are functions of a frequency parameter $\omega$. We will now close the loop from $z$ to $w$ with a complex full block and define the matrix function $\mu$ at a single frequency $\omega_0$.

Referring to Figure 5.3 we will evaluate $M$, $K$, and $\Delta$ at $\omega_0$. Let $\boldsymbol{\Delta_1}$ be defined compatible with the block structure $\mathcal{K}_1(m_r, m_c, m_{C_1})$ as in (5.6). Robust performance can be defined on an augmented block structure $\mathcal{K}(m_r, m_c, m_C)$ by adding a complex full block. This is equivalent to defining $m_C = m_{C_1} + 1$ and adding an integer equal to the dimension of the performance vectors to the $m$-tuple $\mathcal{K}$. Let $n_p$ be equal to the dimension of the performance vectors $w$ and $z$. Then the augmented block structure can be defined by

$$\boldsymbol{\Delta} \equiv \left\{\Delta : \Delta = \operatorname{diag}(\Delta_1, \Delta_2), \Delta_1 \in \boldsymbol{\Delta_1}, \Delta_2 \in \mathbb{C}^{n_p \times n_p}\right\}. \qquad (5.7)$$

$\mu$ can now be defined on the robust performance block structure $\boldsymbol{\Delta}$.

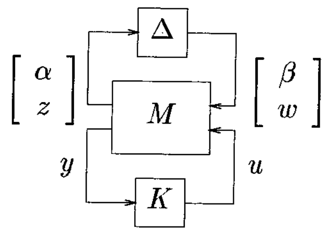
$\Delta$
$\begin{bmatrix} \alpha \\ z \end{bmatrix}$
$M$
$\begin{bmatrix} \beta \\ w \end{bmatrix}$
$y$
$u$
$K$

Figure 5.3: General interconnection for $\mu$-analysis and $\mu$-synthesis.

**Definition 5.2.4** (**$\mu$ for constant matrices [16]**)

$$\mu_{\boldsymbol{\Delta}}(P(\omega_0)) = \begin{cases} \left(\min\limits_{\Delta \in \boldsymbol{\Delta}} \{\bar{\sigma}(\Delta) : \det(I + P(\omega_0)\Delta) = 0\}\right)^{-1} & \textit{or} \\ 0 \textit{ if no } \Delta \in \boldsymbol{\Delta} \textit{ satisfies } \det(I + P(\omega_0)\Delta) = 0. \end{cases}$$

Now with the previous definitions in mind a precise characterization of robust performance can be made.

**Theorem 5.2.1** (**Main Loop [53]**)

$$\mu_{\boldsymbol{\Delta}}(P(\omega_0)) \leq 1 \iff \mu_{\boldsymbol{\Delta}_1}(P_{11}(\omega_0)) \leq 1 \textit{ and } \max_{\|\Delta_1\|<1} \mu_{\boldsymbol{\Delta}_2}(\mathcal{F}_u(\Delta_1, P(\omega_0))) \leq 1.$$

Note that $\mu_{\boldsymbol{\Delta}_2}(\cdot) \equiv \bar{\sigma}(\cdot)$ because of the block structure of $\boldsymbol{\Delta}_2$ [16].

An interpretation of the Main Loop Theorem follows. If $\mu_{\boldsymbol{\Delta}_1}(P_{11}(\omega_0)) < 1$ and $P(\omega_0)$ is stable then there are no $\Delta_1 \in \boldsymbol{B\Delta}_1$ which can destabilize $\mathcal{F}_u(\Delta_1, P(\omega_0))$ and if $\max_{\|\Delta_1\|<1} \mu_{\boldsymbol{\Delta}_2}(\mathcal{F}_u(\Delta_1, P(\omega_0))) < 1$ then performance is achieved at $\omega_0$ since $\bar{\sigma}(\mathcal{F}_u(\Delta_1, P(\omega_0))) < 1 \;\; \forall \Delta_1 \in \boldsymbol{B\Delta}_1$. By a similar analysis it can be shown that if the right hand side of Theorem 5.2.1 is greater than or equal to one then robust performance is not achieved. Hence, Theorem 5.2.1 demonstrates the equivalence of $\mu_{\boldsymbol{\Delta}}(P(\omega_0)) < 1$ and robust performance at $\omega_0$.

In the sequel it will be useful to establish an upper bound for $\mu$. With this in mind the following sets are defined:

$$\begin{aligned} \mathcal{D} \equiv \{D : D = \; & \mathrm{diag}(D_1, \ldots, D_{m_r+m_c}, d_1 I_{k_{m_r+m_c+1}}, \ldots, \\ & d_{m_C} I_{k_{m_r+m_c+m_C}}), \\ & D_i = D_i^* \in \mathbb{C}^{k_i x k_i}, D_i > 0, d_i \in \mathbb{R}, d_i > 0\} \end{aligned} \tag{5.8}$$

$$\begin{aligned} \mathcal{G} \equiv \{G : G = \; & \mathrm{diag}(G_1, \ldots, G_{m_r}, 0_{k_{m_r+1}}, \ldots, 0_{k_{m_r+m_c+m_C}}, \\ & G_i = G_i^* \in \mathbb{C}^{k_i x k_i}\}. \end{aligned} \tag{5.9}$$

It is clear from (5.6) and (5.8) that elements of $\mathcal{D}$ and $\mathbf{\Delta}$ commute, similarly for $\mathcal{G}$ and $\mathbf{\Delta}$. Note that the blocks of $G$ corresponding to complex blocks in $\mathbf{\Delta}$ are 0. An upper bound for $\mu$ can now be written as:

**Theorem 5.2.2 (Upper Bound [17, 55, 53, 75])** *Given $P = P(\omega_0)$ and $\mathbf{\Delta}$, let $\mathcal{D}$ and $\mathcal{G}$ be defined as in (5.8) and (5.9).*

$$\mu_{\mathbf{\Delta}}(M) \leq \inf_{\substack{D\in\mathcal{D}\\ G\in\mathcal{G}}} \min_{\substack{0\leq\beta\\ \beta\in\mathbb{R}}} \left\{\beta : P^*DP + j(GP - P^*G) - \beta^2 D \leq 0\right\}.$$

Henceforth it will be understood that all operations on matrices which are functions of $\omega$ will be shorthand for the supremum over $\omega$, e.g., $\mu_{\mathbf{\Delta}}(P) \equiv \sup_{w\in\mathbb{R}+} \mu_{\mathbf{\Delta}}(P(\omega))$.

1. Estimate the pointwise scaling matrices $D(\omega)$ and $G(\omega)$.
2. Fit state-space realizations $\hat{D}$ and $\hat{G}$ to the pointwise scaling matrices $D(\omega)$ and $G(\omega)$.
3. Compute the scaled system $M_{DG}$ as a function of $M$ and the state-space realizations $\hat{D}$ and $\hat{G}$. Note that $M_{DG}$ is constructed analogous to $DMD^{-1}$. Refer to Young for a precise definition [75].
4. Compute an $\mathcal{H}_\infty$ sub-optimal controller such that $\|\mathcal{F}_\ell(M_{DG}, K)\|_\infty < \gamma$ for values of $\gamma$ chosen close to the optimal.
5. Find $D(\omega)$ and $G(\omega)$ minimizing the mixed $\mu$ upper bound.
6. If $D(\omega)$, $G(\omega)$ are close to the previous values then quit, otherwise return to step 2.

Table 5.1: D,G-K iteration.

## 5.3 Introduction to mixed $\mu$-synthesis

Complex $\mu$-synthesis was proposed by Doyle [17]. It involves an iteration between computing the scaling sets achieving the complex $\mu$ upper bound and synthesizing an $\mathcal{H}_\infty$ optimal (or sub-optimal) controller for the scaled system. It relies on the fact that for the scaling matrix $D$, $\mu(P)$ and $\mu(DPD^{-1})$ are equal, whereas $\overline{\sigma}(P)$ and $\overline{\sigma}(DPD^{-1})$ are not equal.

The $\mu$-synthesis algorithm was extended to work with mixed $\mu$ by Young [75]. The algorithm in Table 5.1 is a summary of the mixed $\mu$-synthesis algorithm from [75]. Refer to [75] for a detailed exposition on mixed $\mu$-synthesis.

This algorithm is used to find a controller satisfying $\mu < \gamma$. Such a controller is guaranteed to provide robustness to uncertainties bounded by $1/\gamma$. It is assumed that the weighted open-loop plant with the uncertainty and performance blocks (Figure 5.3) is given by $M$.

It is noted that the $D$,$G$-scalings are computed as the argument of the infimum in the above algorithm, which yields a matrix of frequency varying data points. The multiplication of the $D$,$G$-scalings implied in the algorithm is done before the frequency varying data are fit to a transfer function. This prevents the states of the $D$,$G$-scalings from accumulating in each iteration of the algorithm.

If we assume that at each step $D(\omega)$ and $G(\omega)$ fit perfectly, then the iteration is monotonically non-increasing, so it will converge to a controller attempting to minimize $\mu$. However, there is no *a priori* guarantee that the algorithm will converge to the global minimum, as it is a non-convex optimization and thus has only local minima.

## 5.4 Introduction to LMIs

Many results in robust control theory can be reduced to conditions on a set of LMIs. A tutorial overview of LMIs was presented by Doyle *et al.* [21]. Packard *et al.* provide examples of several important problems in robust control which can be reduced to LMIs [58]. Packard and Wu developed an automatic gain scheduling technique using LMIs [57, 54]. Apkarian and Gahinet provide a general solution for linear parameter-varying (LPV) $\mathcal{H}_\infty$ controllers [2], which is applied to a missile autopilot problem [4]; a general LPV $\mathcal{H}_\infty$ design example is presented by Apkarian *et al.* [3].

The general LMI problem consists of linear and affine hermitian constraints on a block diagonal hermitian matrix of decision variables. Consider the set of decision variables defined by the $m$-tuple $\mathcal{K}(m_F, m_S) = (k_1, \ldots, k_{F+S})$ given by

$$\begin{aligned} \mathcal{X} \equiv \{X : X = \operatorname{diag}\left[X_1, \ldots, X_F, x_1 I_{k_{F+1}}, \ldots, x_S I_{k_{F+S}}\right], \\ X_i = X_i^* \in \mathbb{C}^{k_i \times k_i}, x_i \in \mathbb{R}\}. \end{aligned} \tag{5.10}$$

An LMI consists of any set of linear or affine hermitian constraints on decision variables; the decision variables have structure compatible with the set $\mathcal{X}$ in (5.10).

Several examples of standard problems which can be represented by LMIs will be presented. These examples were drawn from the work of Doyle *et al.* [21].

**Example 5.4.1** *The stability of a discrete-time system is equivalent to checking if $\rho(A) < 1$, where $x(k+1) = Ax(k)$. It can be shown that the condition* *on the spectral radius of $A$ is equivalent to the following conditions.*

$$\begin{aligned} & \rho(A) < 1 \\ \Longleftrightarrow & \exists T \; : \; \overline{\sigma}\left(TAT^{-1}\right) < 1 \\ \Longleftrightarrow & \exists T \; : \; TAT^{-1}(TAT^{-1})^* - I < 0 \\ \Longleftrightarrow & \exists T \; : \; A(T^*T)^{-1}A^* - (T^*T)^{-1} < 0 \\ \Longleftrightarrow & \exists X > 0 \; : \; AXA^* - X < 0. \end{aligned}$$

*This condition is defined by two LMIs, $X > 0$ and $AXA^* - X < 0$, where $X = X^*$ is an unstructured decision variable.*

*Similarly, the well known Lyapunov condition for checking the stability of the continuous-time system $\dot{x}(t) = Ax(t)$ is defined by two LMIs: $X > 0$ and $AX + XA^* < 0$, where $X = X^*$ is an unstructured decision variable.*

**Example 5.4.2** *The upper bound for complex (and mixed) $\mu$ is equivalent to an LMI.*

$$\begin{aligned} & \mu(M) < \gamma \\ \Longleftarrow & \exists D > 0 \; : \; \overline{\sigma}\left(DMD^{-1}\right) < \gamma \\ \Longleftrightarrow & \exists D > 0 \; : \; D^{-1}M^*D^2MD^{-1} - \gamma^2 I < 0 \\ \Longleftrightarrow & \exists D > 0 \; : \; M^*D^2M - \gamma^2 D^2 < 0 \\ \Longleftrightarrow & \exists X > 0 \; : \; M^*XM - \gamma^2 X < 0. \end{aligned}$$

*This condition is equivalent to solving the two LMIs: $X > 0$ and $M^*XM - \gamma^2 X < 0$, where $X = X^*$ is a decision variable sharing the same structure as $D$.*

Once a problem has been reduced to a set of LMIs there are several commercially available software packages which can compute their solution. One such package is the LMI Control Toolbox, which works within MATLAB, by Gahinet *et al.* [26, 25]. A limitation of the LMI Control Toolbox is that it can only solve LMIs whose coefficients are real matrices.

Let $L(X) < 0$ represent a complex-valued hermitian LMI with decision variable $X \in \boldsymbol{\mathcal{X}}$. Feasible solutions to $L(\cdot)$ are determined by checking that the eigenvalues of $L(X)$, for a fixed decision variable $X$, are negative. In other words, $X$ is a feasible solution to the LMI $L(\cdot)$ if $z^*L(X)z < 0$, $\forall z \neq 0 \in \mathbb{C}^n$, where $n$ is the dimension of $L(\cdot)$. In order to use the LMI Control Toolbox $L(\cdot)$ must be reduced to a set of real-valued symmetric LMIs.

For complex-valued hermitian LMIs, this feasibility test can be reformulated in terms of real-valued LMIs. Given a complex matrix, $M$, define the $\hat{M}$ operation as

$$\hat{M} = \begin{bmatrix} \text{Real}(M) & -\text{Imag}(M) \\ \text{Imag}(M) & \text{Real}(M) \end{bmatrix}. \tag{5.11}$$

Now define the matrix operator equivalent of $j = \sqrt{-1}$ by

$$J = \begin{bmatrix} 0 & -I \\ I & 0 \end{bmatrix}. \tag{5.12}$$

Note that $J$ is skew-symmetric ($J^T = -J$). Following are several technical lemmas which will prove useful when converting complex-valued hermitian LMIs to real-valued symmetric LMIs.

**Lemma 5.4.1** *Let $A, B \in \mathbb{C}^{n \times m}$ be represented by $A = u+jv$ and $B = x+jy$, where $u, v, x, y \in \mathbb{R}^{n \times m}$, then*

$$\widehat{A+B} = \hat{A} + \hat{B}.$$

**Proof:**

$$\begin{aligned}
\widehat{A+B} &= \widehat{(u+jv)+(x+jy)} \\
&= \widehat{(u+x)+j(v+y)} \\
&= \begin{pmatrix} (u+x) & -(v+y) \\ (v+y) & (u+x) \end{pmatrix} \\
&= \begin{pmatrix} u & -v \\ v & u \end{pmatrix} + \begin{pmatrix} x & -y \\ y & x \end{pmatrix} \\
&= \hat{A} + \hat{B}.
\end{aligned}$$

■

**Lemma 5.4.2** *Let $A \in \mathbb{C}^{n \times m}$ and $B \in \mathbb{C}^{m \times n}$ be represented by $A = u + jv$ and $B = x + jy$, where $u, v \in \mathbb{R}^{n \times m}$ and $x, y \in \mathbb{R}^{m \times n}$, then*

$$\widehat{AB} = \hat{A}\hat{B}.$$

**Proof:**

$$\begin{aligned}
\widehat{AB} &= \widehat{(u+jv)(x+jy)} \\
&= \widehat{(ux-vy)+j(vx+uy)} \\
&= \begin{pmatrix} (ux-vy) & -(vx+uy) \\ (vx+uy) & (ux-vy) \end{pmatrix} \\
&= \begin{pmatrix} u & -v \\ v & u \end{pmatrix} \begin{pmatrix} x & -y \\ y & x \end{pmatrix} \\
&= \hat{A}\hat{B}.
\end{aligned}$$

■

From Lemmas 5.4.1 and 5.4.2 it is clear that the $\hat{}$ operator preserves the ring structure of operators and matrices in $\mathbb{C}^{n \times n}$. Because of this, the $\hat{}$ operator will prove useful when reducing a complex-valued hermitian LMI to a real-valued symmetric LMI.

Now it is possible to reformulate the complex-valued hermitian LMI, $L(X)$, in terms of real-valued symmetric LMIs.

**Theorem 5.4.1** *Let $M = M^* \in \mathbb{C}^{n \times n}$ be represented by $M = A + jB$, where $A = A^T \in \mathbb{R}^{n \times n}$ and $B = -B^T \in \mathbb{R}^{n \times n}$.*

$$M < 0 \quad \textit{if and only if} \quad \hat{M} < 0.$$

*Note that $\hat{M}$ is a symmetric matrix.*

**Proof:**

*Necessary condition. We must demonstrate that $M < 0$ implies that $\alpha^* \hat{M} \alpha < 0, \forall \alpha \neq 0 \in \mathbb{C}^{2n}$. Define $\alpha$ by*

$$\alpha = \begin{pmatrix} c \\ d \end{pmatrix} + j \begin{pmatrix} e \\ f \end{pmatrix}.$$

*Let $x_1 = c + jd$ and $x_2 = e + jf$, then $M = M^*$ implies that*

$$\begin{aligned} \alpha^* \hat{M} \alpha &= \begin{pmatrix} c \\ d \end{pmatrix}^T \begin{pmatrix} A & -B \\ B & A \end{pmatrix} \begin{pmatrix} c \\ d \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}^T \begin{pmatrix} A & -B \\ B & A \end{pmatrix} \begin{pmatrix} e \\ f \end{pmatrix} \\ &= (c^T A c + d^T A d + 2 d^T B c) + (e^T A e + f^T A f + 2 f^T B e) \\ &= x_1^* M x_1 + x_2^* M x_2. \end{aligned}$$

*Because $x^* M x < 0, \forall x \neq 0 \in \mathbb{C}^n$ by supposition, it follows that $\alpha^* \hat{M} \alpha < 0, \forall \alpha \neq 0 \in \mathbb{C}^{2n}$.*

*Sufficient condition. We must demonstrate that $\hat{M} < 0$ implies that $x^* M x < 0, \forall x \neq 0 \in \mathbb{C}^n$. By supposition $\alpha^* \hat{M} \alpha < 0, \forall \alpha \neq 0 \in \mathbb{C}^{2n}$. Let $\alpha$, $x_1$, and $x_2$ be defined as above, then*

$$\alpha^* \hat{M} \alpha = x_1^* M x_1 + x_2^* M x_2 < 0, \forall \alpha \neq 0 \in \mathbb{C}^{2n}.$$

*In particular, for $x_1 = c + jd \neq 0 \in \mathbb{C}^n$, choose*

$$\alpha = \begin{pmatrix} c \\ d \end{pmatrix} + j0.$$

*Then $x_1^* M x_1 = \alpha^* \hat{M} \alpha < 0, \forall x_1 \neq 0 \in \mathbb{C}^n$.*

■

# Chapter 6

# Design and Comparison of Hover Controllers

The primary difficulties in designing control systems for helicopters are their inherent instability and nonlinearity. This is aggravated when flying the helicopter in extreme conditions, such as in nap-of-the-earth (NOE) flight [11, 12, 13] or in high gust hover situations.

The traditional solution to the helicopter problem in the aircraft industry has been gain scheduling [59, 62]. The basic technique is to isolate a finite number of operating points along an equilibrium manifold within the flight envelope of the aircraft and linearize the aircraft model at each of these operating points. Scheduling parameters are generally chosen so that they are slowly varying and reflect large changes in the dynamic behavior of the system. A controller is then designed for each of the linearized models and a switching algorithm is used to select the proper controller depending on the state of the aircraft. Figure 6.1 shows a typical gain-scheduled controller structure. The main problem with this control strategy is that

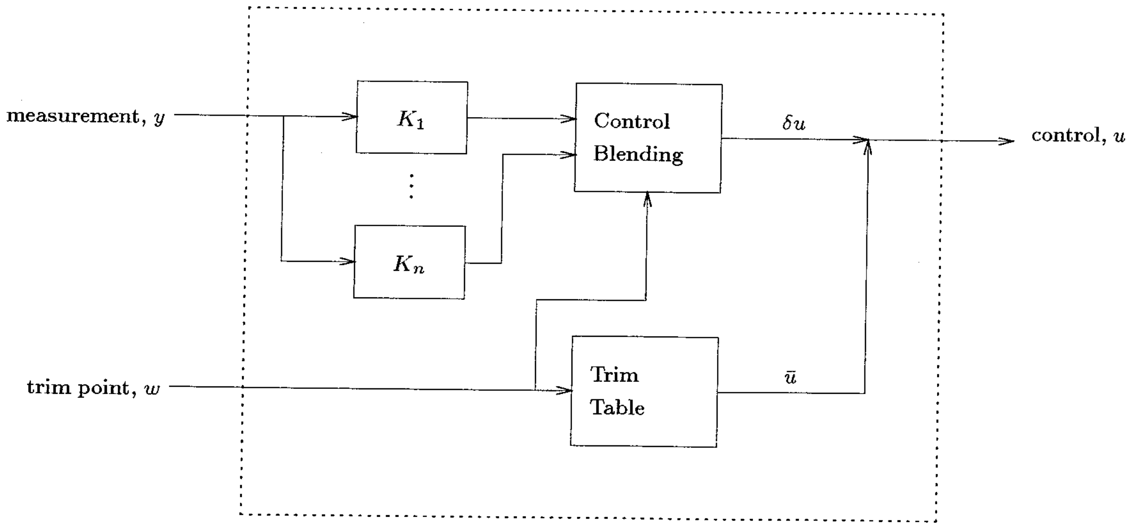

Figure 6.1: Generic structure for a gain scheduled controller.

in order to guarantee performance and stability for the global scheduled system the scheduling parameters need to be slowly varying and capture essential nonlinearities of the system. This generally precludes the use of large maneuvers over the operating region.

Another approach, which has not yet been very successful, is dealing with nonlinearities directly in the fashion of Isidori, Krener and others [36, 37, 41]. The main problem with this approach is that it requires an exact, analytic nonlinear model for the system which, in general, is not possible. In some cases approximate models are adequate [41].

An "automatic" gain scheduling technique was developed by Packard [54]. Another approach to this problem was taken by Apkarian and Gahinet [2]. The automatic gain-scheduling technique is applicable to a broad range of systems characterized by LPV models. The LPV technique appears to be a reasonable compromise between directly accounting for nonlinearities in a system and robust linear modelling.

The LPV technique has been successfully employed in the design of robust gain scheduled controllers for an experimental thrust-vectored ducted fan engine by Bodenheimer *et al.* [10]. A more detailed discussion of the work on the ducted fan engine can be found in [9].

The focus in this chapter is the development of hover controllers for the helicopter. This is a necessary first step in order to understand the behavior of the helicopter. The applicability of this work addresses exactly the sort of problems experienced by helicopter pilots when flying in extreme weather situations during mountainous search and rescue missions. In particular, while flying near cliffs a pilot must maintain an extremely stable hover in the presence of high gusting and significant ground effect.

Section 6.1 discusses the main objectives of the helicopter controller. $\mathcal{H}_\infty$ and LQG controllers satisfying the main objectives are presented in Sections 6.2 and 6.3. A comparison of both controllers is made in Section 6.4. An evaluation of a preliminary controller which regulates the main rotor speed is presented in Appendix 6.A.

## 6.1 Hover control objectives

The main control objective is to reject disturbances, track computer generated reference commands on the helicopter attitude, and decouple the modes at hover. An example of typical disturbances acting on the helicopter are wind gusts.

The helicopter is piloted using a bank-to-turn technique, with yaw used to reduce the couplings between pitch and roll. Thus, the controller is designed such that the dynamics of yaw are fast relative to those of pitch and roll. The choice for controlling the attitude directly, as opposed to the more conventional body rates, results from the objective of achieving autonomous flight through the use of computer generated trajectories. Low and Garrard provide a review of traditional handling qualities specifications for piloted rotorcraft [46].

Because the underlying dynamics of the helicopter are extremely nonlinear, with strongly coupled modes, and the model used for control is linear and valid only at hover, a robust control technique is essential.

A similar effort, developing controllers for a model helicopter, has been carried out by Weilenmann *et al.* See [74] for a discussion of their helicopter workbench. Several hover and vertical flight controllers were developed and compared by Weilenmann *et al.* [15, 73]. A detailed discussion is contained in [72].

## 6.2 $\mathcal{H}_\infty$ control design

The identified discrete-time helicopter model is first converted to continuous-time using a norm-preserving bilinear transformation. A controller is then designed using continuous-time $\mathcal{H}_\infty$ sub-optimal synthesis, with weighting functions specified in continuous-time [19, 28]. A discrete-time $\mathcal{H}_\infty$ controller suitable for implementation on the experimental helicopter platform is obtained using a bilinear transformation on the continuous-time controller.

### 6.2.1 Uncertainty description

The uncertainty is not described in a detailed manner, but rather all of the effects are gathered into full-block uncertainty at the input of the plant. A multiplicative-error provides a description of the plant mismatch as well as a characterization of robust stability. This type of uncertainty description is common in the design of helicopter control systems [23]. For robustness to hold, the controller must stabilize the set of plants defined by

$$\{G : G = G_0(I + \Delta_m W_m),\ \Delta_m \in \mathbb{C}^{3\mathrm{x}3},\ \|\Delta_m\|_\infty \leq 1\},$$

where $G_0$ represents the nominal model $\hat{G}_3$ identified in Chapter 4 and $W_m$ is the multiplicative uncertainty weight specifying the amount of uncertainty in the model as a function of frequency.

Robust stability is satisfied if and only if $\|W_m\, T_i\|_\infty < 1$, where $T_i = G_0K(I + G_0K)^{-1}$ is the plant input complementary sensitivity transfer function. The uncertainty weight is of the form $W_m = \mathrm{diag}\,(w_m^1,\ w_m^2,\ w_m^3)$, The $w_m^i$ are stable bi-proper minimum phase scalar valued rational functions with large magnitude outside the frequency range of the experiment to account for unmodelled dynamics, and in particular for the actuator dynamics modelled by a pure delay, $d = 1$, in (4.2), which are not included in the design model. The low frequency magnitude of $W_m$ serves to limit the size of actuator commands and is adjusted to prevent actuator saturation.

### 6.2.2 Performance specifications

Consider the generalized system depicted in Figure 6.2. The selection of $w$ and $z$ was based on performance requirements: the exogenous input, $w$, contains a disturbance input, *dist*, acting on the output and a perturbation, *pert*, acting on the control, whereas the error signals $z$ are weighted outputs and the measurement $e$ is the tracking error. The exogenous disturbance, *dist*, is treated as a pilot command or reference signal. Note that *dist* can also be interpreted as measurement noise. The interpretation of *dist* as a reference signal or measurement noise depends on the choice of the weighting function, $W_p$. For reference signals $W_p$ will tend to have high gain at lower frequencies to minimize low frequency tracking errors. For measurement noise $W_p$ will tend to have high gain at high frequencies to minimize the effect of high frequency noise. Performance of the closed-loop is evaluated in terms of a weighted $\mathcal{H}_\infty$ norm of

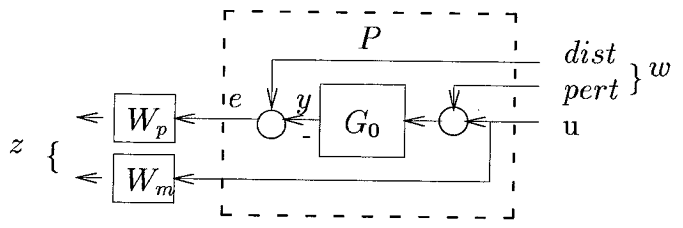

Figure 6.2: Helicopter robust synthesis interconnection.

the output sensitivity transfer function. Nominal performance is achieved if and only if $\|W_p\, S_o\|_\infty < 1$, where $S_o = (I + KG_0)$ is the output sensitivity transfer function. The performance weight $W_p$ is of the form $W_p = \text{diag}\left(w_p^1,\, w_p^2,\, w_p^3\right)$, with $w_p^i$ stable bi-proper minimum phase scalar valued rational functions incorporating integral action in order to minimize steady-state error between step commands, *dist*, and the helicopter attitude, $y$.

## 6.2.3 $\mathcal{H}_\infty$ synthesis

Consider the standard feedback interconnection, shown in Figure 5.1, where $K$ is the controller and $P$ is the generalized plant as defined in Figure 6.2. The objective is to minimize over all frequencies the maximal energy captured by the closed-loop transfer function from exogenous input $w$ to the error signal $z$. This transfer function is denoted $\mathcal{F}_\ell\,(P, K)$. The synthesis problem involves finding a controller $K$ such that performance requirements are satisfied under the prescribed uncertainties.

The weights $W_m = \text{diag}\,(w_m^1,\, w_m^2,\, w_m^1)$ and $W_p = \text{diag}\left(w_p^1,\, w_p^2,\, w_p^3\right)$ are defined as

$$w_m^i = g_i \frac{s+z}{s+p_i},\ i = 1,\ 2 \tag{6.1}$$

$$w_p^i = \gamma_p \frac{s^2 + 2\xi\alpha_i s + \alpha_i^2}{s^2 + 2\xi\beta_i s + \beta_i^2},\ i = 1, \ldots, 3, \tag{6.2}$$

where $g_1 = 220$, $g_2 = 440$, $z = 10$, $p_1 = 2000$, $p_2 = 4000$, $\xi = 0.7$, $\alpha_1 \approx 39$, $\alpha_2 \approx 85$, $\alpha_3 \approx 63$, $\beta_1 \approx 0.01$, $\beta_2 \approx 0.012$, $\beta_3 \approx 0.01$, and $\gamma_p = 10^{-8}$. The weights $W_m$ and $W_p$ are shown in Figure 6.3. In Chapter 8, frequency-domain model validation

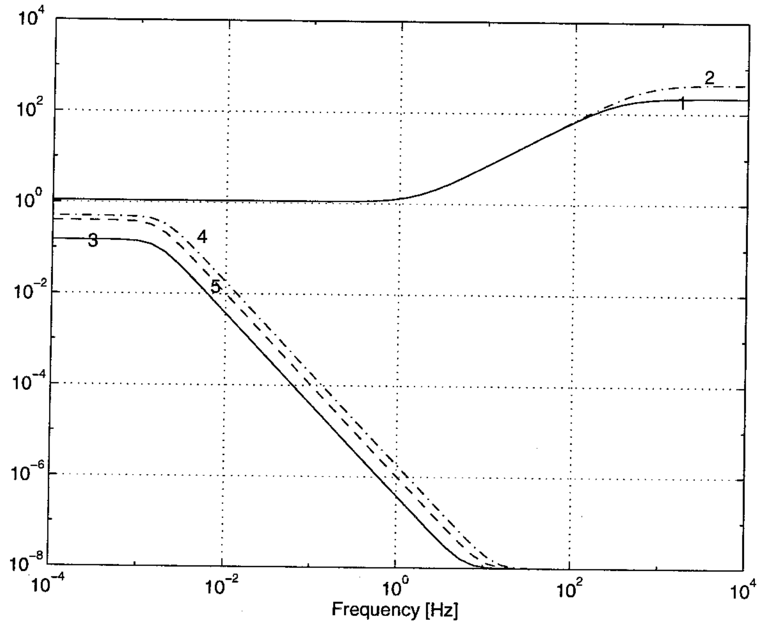

Figure 6.3: Frequency Response of $W_m$ and $W_p$: 1 is $w_m^1$ and $w_m^3$, 2 is $w_m^2$, 3 is $w_p^1$, 4 is $w_p^2$ and 5 is $w_p^3$. Only the magnitude is shown.

techniques with open-loop input-output experimental data signals will validate the interconnection in Figure 6.2 with these weights.

For sub-optimal $\mathcal{H}_\infty$ synthesis, the objective is to find a stabilizing controller $K$ such that $\|\mathcal{F}_\ell(P,K)\|_\infty < \gamma$ for the smallest value of $\gamma$. The minimization is carried out iteratively and is known as $\gamma$-iteration [19]. The resulting sub-optimal $\mathcal{H}_\infty$ controller achieved a closed-loop $\infty$-norm of approximately 1.24.

Figure 6.4 shows simulated step-responses of the closed-loop control system. Input–output signals corresponding to the models $\hat{G}_3$ and $\hat{G}_6$ are plotted with solid lines and mixed lines, respectively. The reference signals are plotted with dashed lines. It can be seen that the $\mathcal{H}_\infty$ controller decouples the modes very well. The responses are fast. The controller rolls off at high frequencies, as can be seen in Figure 6.5.

## 6.3 LQG control design

The objective in $\mathcal{H}_2$ control design is to minimize the average energy over all frequencies captured by the closed-loop transfer function from exogenous inputs $w$ to the error signal $z$ (Figure 6.2). The plant output error is augmented with integrators so that zero steady-state tracking error can be achieved. For $\mathcal{H}_2$ synthesis, the objective is to find a stabilizing controller $K$ which minimizes the 2-norm of the closed-loop transfer matrix, noted $\|\mathcal{F}_\ell(P,K)\|_2$. In the classical LQG framework, the objective consists of finding an admissible controller which minimizes the quadratic cost

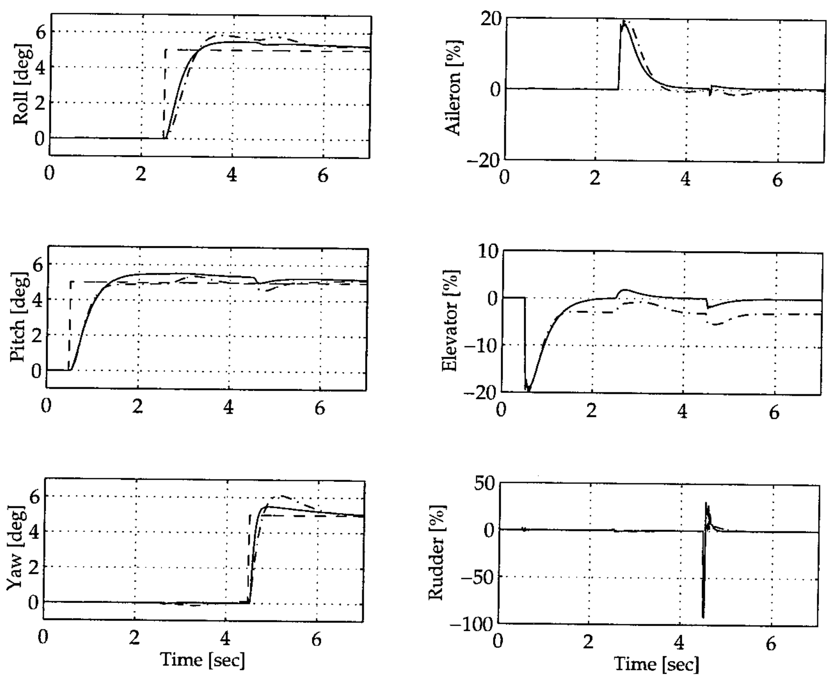

Figure 6.4: Simulated step responses with $\mathcal{H}_\infty$ controller: $\hat{G}_3$ (solid), $\hat{G}_6$ (mixed), command (dashed).

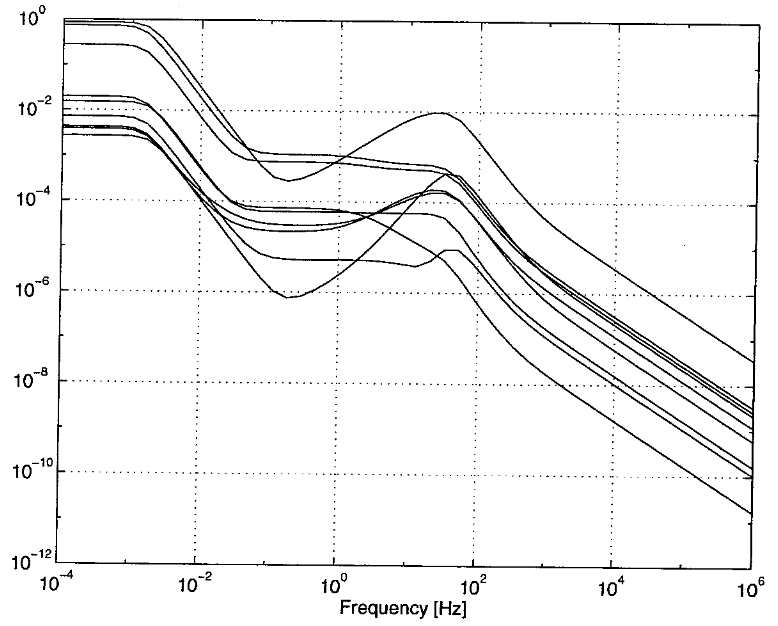

Figure 6.5: Magnitude of the frequency response of the $\mathcal{H}_\infty$ controller.

function

$$J_{LQG} = \lim_{N\to\infty} E\left(\frac{1}{N}\sum_{t=1}^{N} x(t)^T Q x(t) + u(t)^T R u(t)\right), \tag{6.3}$$

where $x = (\int\phi, \int\theta, \int\psi, \phi, \theta, \psi, p, q, r)^T$ corresponds to the augmented state vector, including the states of the integrators, the attitude, and the attitude angular rate, and $u = (\Theta_a, \Theta_e, \Theta_r)^T$. $Q \geq 0$ and $R > 0$ are weighting matrices and $E(\cdot)$ denotes the expectation operator. Substituting $z = (x^T Q^{1/2},\ u^T R^{1/2})^T$ in Figure 6.2 and assuming $w$ is a white noise signal, the LQG cost, $J_{LQG}$, is simply the variance of $z$, which is given by $J_{LQG} = \|\mathcal{F}_\ell(P,K)\|_2^2$, i.e., it is captured in the $\mathcal{H}_2$ framework.

The controller was designed with standard discrete-time LQG software, which bypassed the need to convert to continuous-time. After designing the controller, $K$, in Figure 6.2, the integrators were wrapped back into the controller that was implemented on the helicopter.

A state feedback regulator design is first done, to determine the maximum achievable performance with all information about the states available. Then, for the output feedback case, an estimator is designed. The noise covariance matrices are set to identity, for the estimator design. For the regulator design, the noise covariance matrices, $Q$ and $R$, were tuned to insure that a good step response was achieved without saturating the control inputs. In particular, the velocity states and the controls were heavily penalized, to try to obtain an over-damped step response. This is desirable, since it will prevent excitation of the unmodelled high frequency dynamics of the helicopter. With this approach, some measure of robustness should be obtained but cannot be guaranteed. The weighting matrices $Q$ and $R$ for the regulator are

$$\begin{aligned} Q &= \operatorname{diag}(.001,\ .001,\ .001,\ 100,\ 20,\ 10,\ 10,\ .1,\ 50) \\ R &= 5\times 10^5 \operatorname{diag}(1,\ 1,\ 2)\,. \end{aligned}$$

Figure 6.6 shows the step-responses of the control system. Input–output signals corresponding to simulations of $\hat{G}_3$ and $\hat{G}_6$ are plotted with solid lines and mixed lines, respectively. The reference signals are plotted with dashed lines. The LQG controller decouples the modes well, however the responses appear to be slower than those with the $\mathcal{H}_\infty$ controller. This is explained in Section 6.4. Note that higher performance specification could not be achieved as the LQG technique does not provide enough degrees of freedom for the controller to roll off fast enough, as shown in Figure 6.7.

## 6.4 Results and comparison

To make a fair comparison, both types of controllers were designed to have comparable performance with respect to step command tracking in simulation. When designing a controller, a key factor prior to implementation is to check that the actuators are not over utilized. For a 5 degree step on the pilot commands, the rule of thumb employed in this work was to allow the aileron and elevator to use no more than 50%

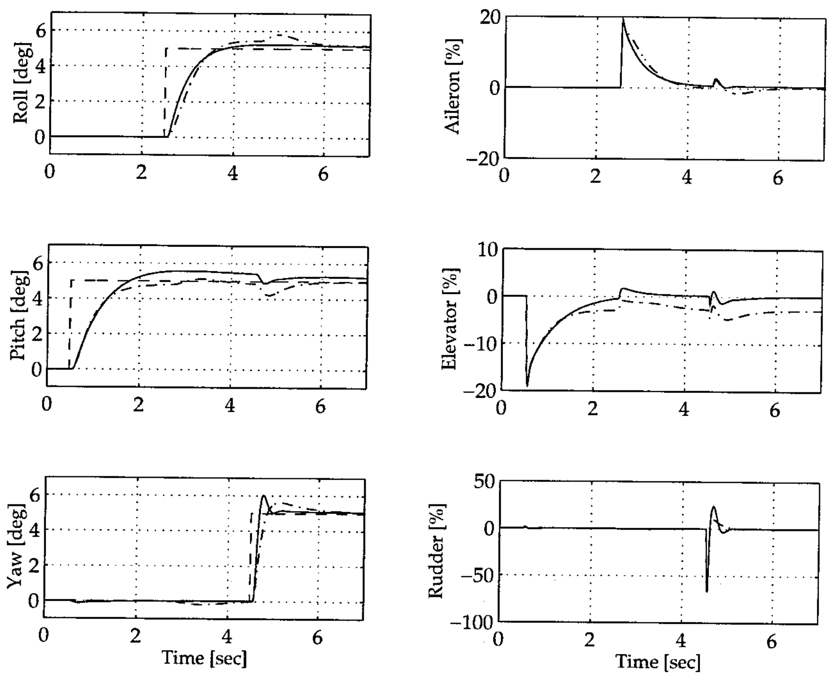

Figure 6.6: Simulated step responses with LQG controller: $\hat{G}_3$ (solid), $\hat{G}_6$ (mixed), command (dashed).

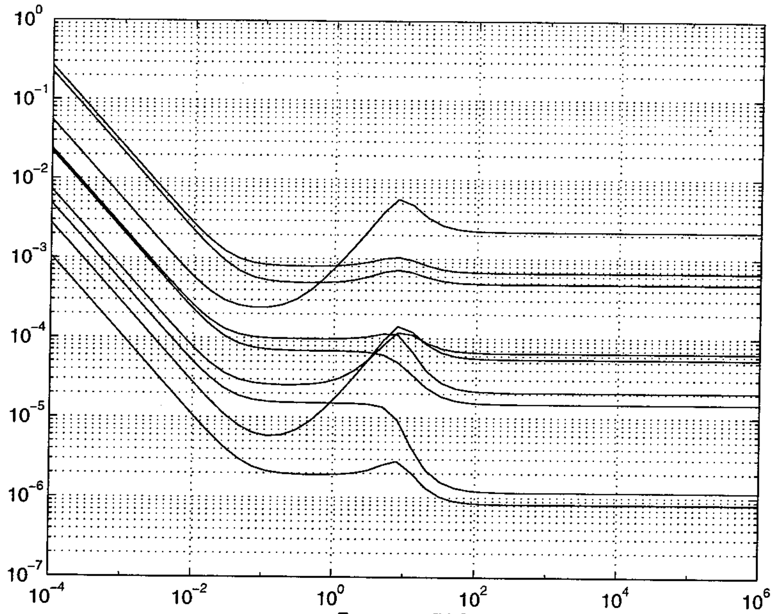

Figure 6.7: Magnitude of the frequency response of the LQG controller.

and the rudder 100% of the maximum servo command. The performance of each controller was compared by measuring the rise time, overshoot, and settling time on each axis. The rise time was computed from 10% to 90%, overshoot was computed in the standard fashion, and settling time was defined as the time it takes to fall below 15% ripple on the steady-state value.

Comparable time responses on simulations were obtained with both types of controllers, as seen in Figures 6.4 and 6.6. However, some of the LQG controllers designed were closed-loop unstable when implemented on the actual plant, so the LQG performance specifications had to be substantially reduced. Although the LQG controller presented is not likely to represent the best possible LQG design, a better design requires moving closer to closed-loop instability. Furthermore, the simulations did not predict this instability, so it became unsafe to continue design iteration and testing of LQG controllers since they were sometimes unstable in implementation. For the $\mathcal{H}_\infty$ design, once an appropriate uncertainty description was obtained, it was possible to significantly change the performance weights without risking closed-loop instability. Beyond a certain limit, saturation occurred on one or more axes, which limited the achievable performance.

Both the $\mathcal{H}_\infty$ and LQG controllers were implemented on the model helicopter in the three DOF configuration. The helicopter was first piloted to hover. Then the controller was switched on and trimmed. The pilot (in this case the real-time computer) provided exogenous reference commands on roll, pitch, and yaw. The commands consisted of one positive and one negative pulse of 5 degrees on each axis independently. Disturbances were simulated by tapping on the helicopter.

### 6.4.1 $\mathcal{H}_\infty$ controller

Figure 6.8 contains experimental tracking responses for the model helicopter with the $\mathcal{H}_\infty$ controller (solid) along with the commands (dashed). Additionally, for comparison, the simulated responses for $\hat{G}_3$ and $\hat{G}_6$ are shown (mixed) and (dotted), respectively. Figure 6.9 contains the resulting control signals, with the same conventions.

The helicopter response to a pilot command on the roll angle is fast, with a rise time of about half a second. It takes about eight seconds to settle and has overshoot of about 20%. The response on pitch is quite fast with about half a second of rise time, settling time of about four seconds, and 20% overshoot. There is some coupling in the yaw axis. The response on yaw is highly damped with very fast rise time, however, the tail rotor collective saturates the yaw axis actuator. This is largely due to an unmodelled asymmetry in this actuator. Although the yaw axis appears to track the reference command there is a great deal of oscillation. For this reason no measures were computed. The disturbance rejection properties of the $\mathcal{H}_\infty$controller were tested, but the results are not shown. At hover, the helicopter appears to be quite stiff and disturbing it requires a significant force. The response to a simultaneous disturbance on all three axes dies off in about a half second on the roll and yaw axis, and about one second, with some ringing, on the pitch axis. This behavior was predictable as

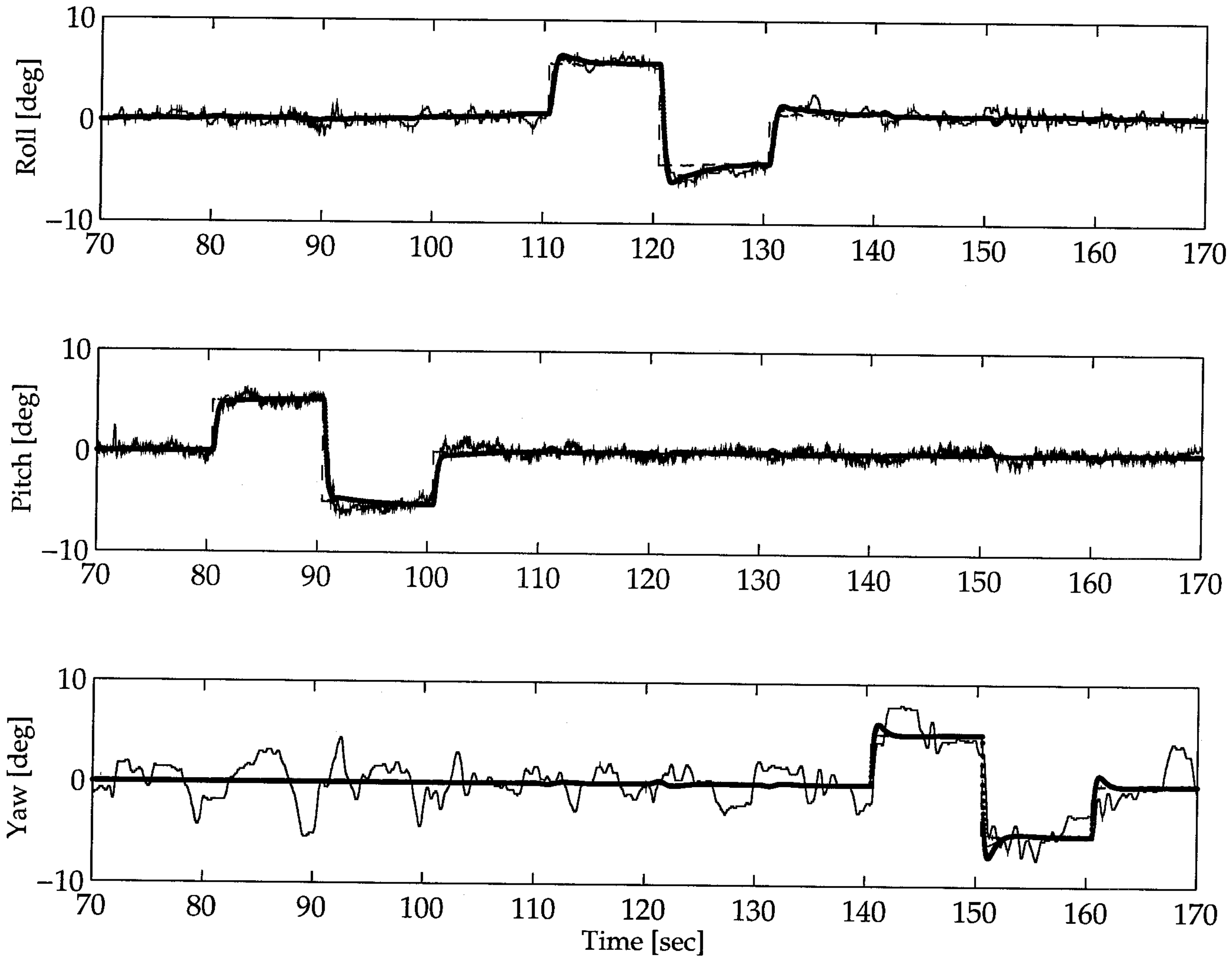

Figure 6.8: Experimental step responses using the $\mathcal{H}_\infty$controller: experimental (solid), commanded (dashed), $\hat{G}_3$ simulation (mixed), $\hat{G}_6$ simulation (dotted).

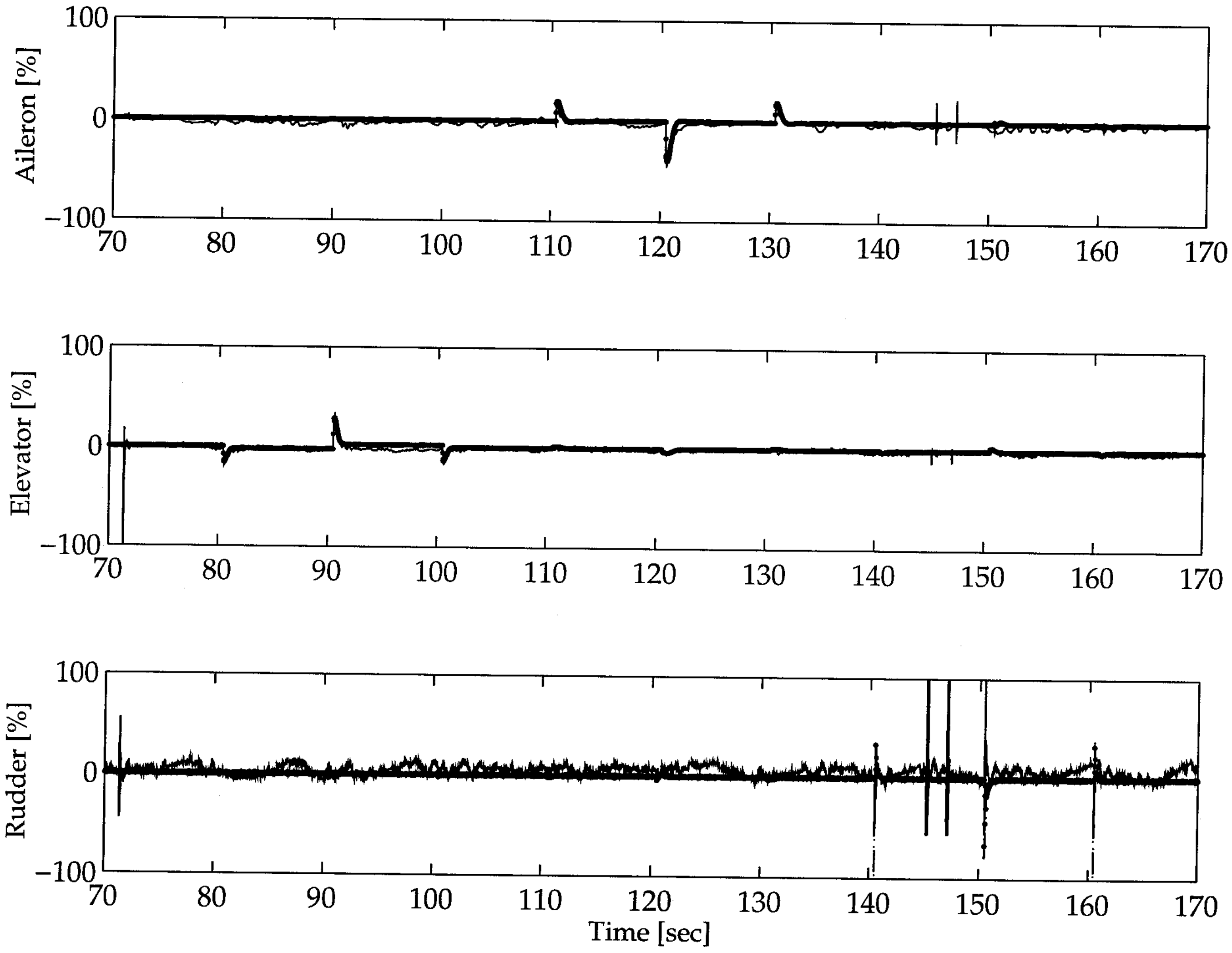

Figure 6.9: Experimental control signals for the $\mathcal{H}_\infty$ controller resulting from step commands: experimental (solid), $\hat{G}_3$ simulation (mixed), $\hat{G}_6$ simulation (dotted).

the roll and yaw axes are highly damped and the pitch axis is less damped.

### 6.4.2 LQG controller

Figure 6.10 contains experimental tracking responses (solid) for the model helicopter with the LQG controller along with the commands (dashed). For comparison, the simulated responses for $\hat{G}_3$ and $\hat{G}_6$ are shown (mixed) and (dotted), respectively. Figure 6.11 contains the resulting control signals, with the same conventions. The

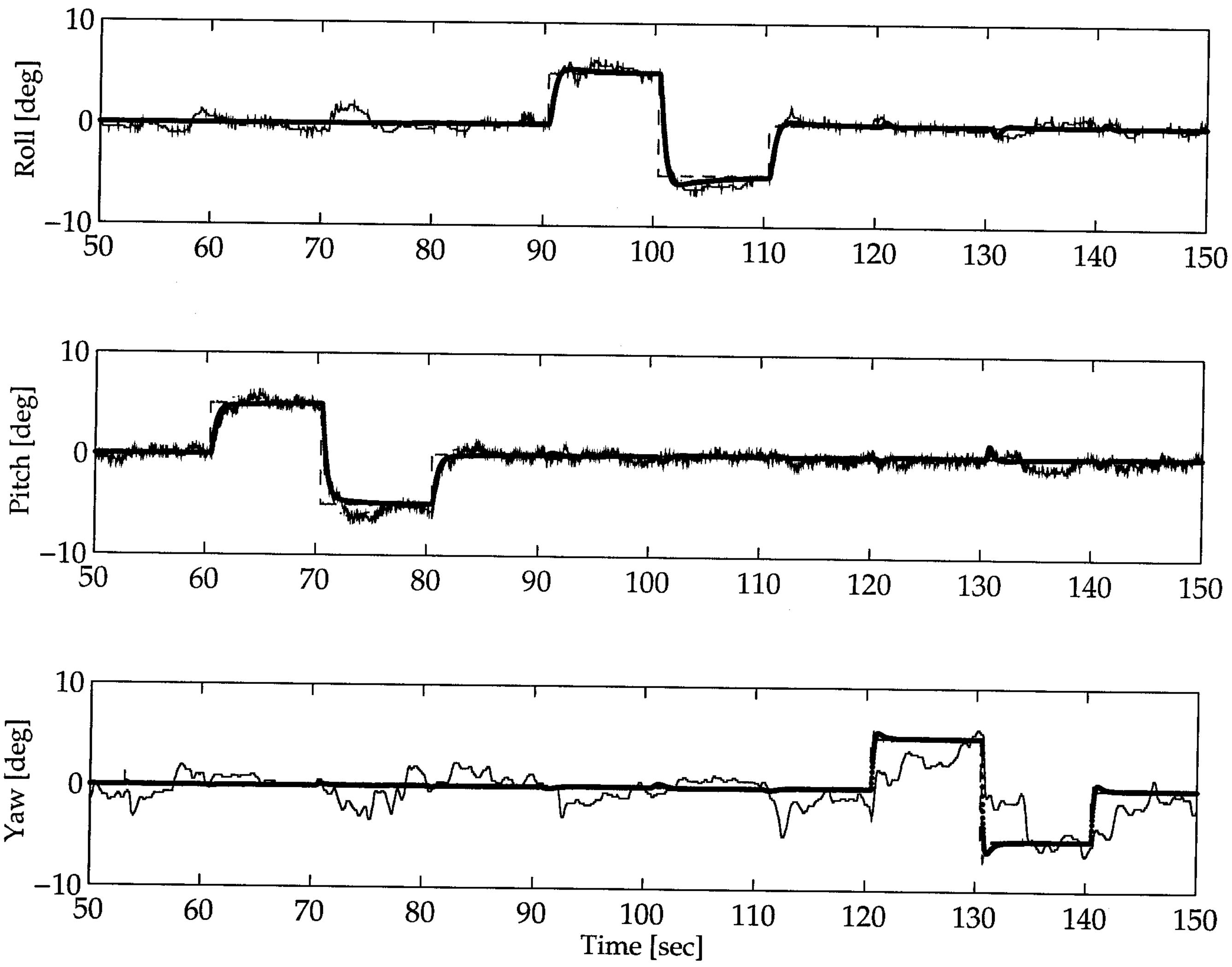

Figure 6.10: Experimental step responses using the LQG controller: experimental (solid), commanded (dashed), $\hat{G}_3$ simulation (mixed), $\hat{G}_6$ simulation (dotted).

helicopter response on roll is relatively fast with a rise time of about half a second, settling time of about eight seconds, and overshoot of 30%. Unlike the $\mathcal{H}_\infty$ controller, there is some coupling resulting from a pitch command in the roll axis, as yaw is not as fast as it was with the $\mathcal{H}_\infty$ controller. The pitch axis has a rise time of more than one second, a settling time of about five seconds, and 20% overshoot. The yaw angle does not track the command well. The tail rotor collective saturates with a step change of −10 degrees, but does not contribute much otherwise. Surprisingly, there is very little coupling of the yaw response in the pitch and roll axes. The disturbance rejection

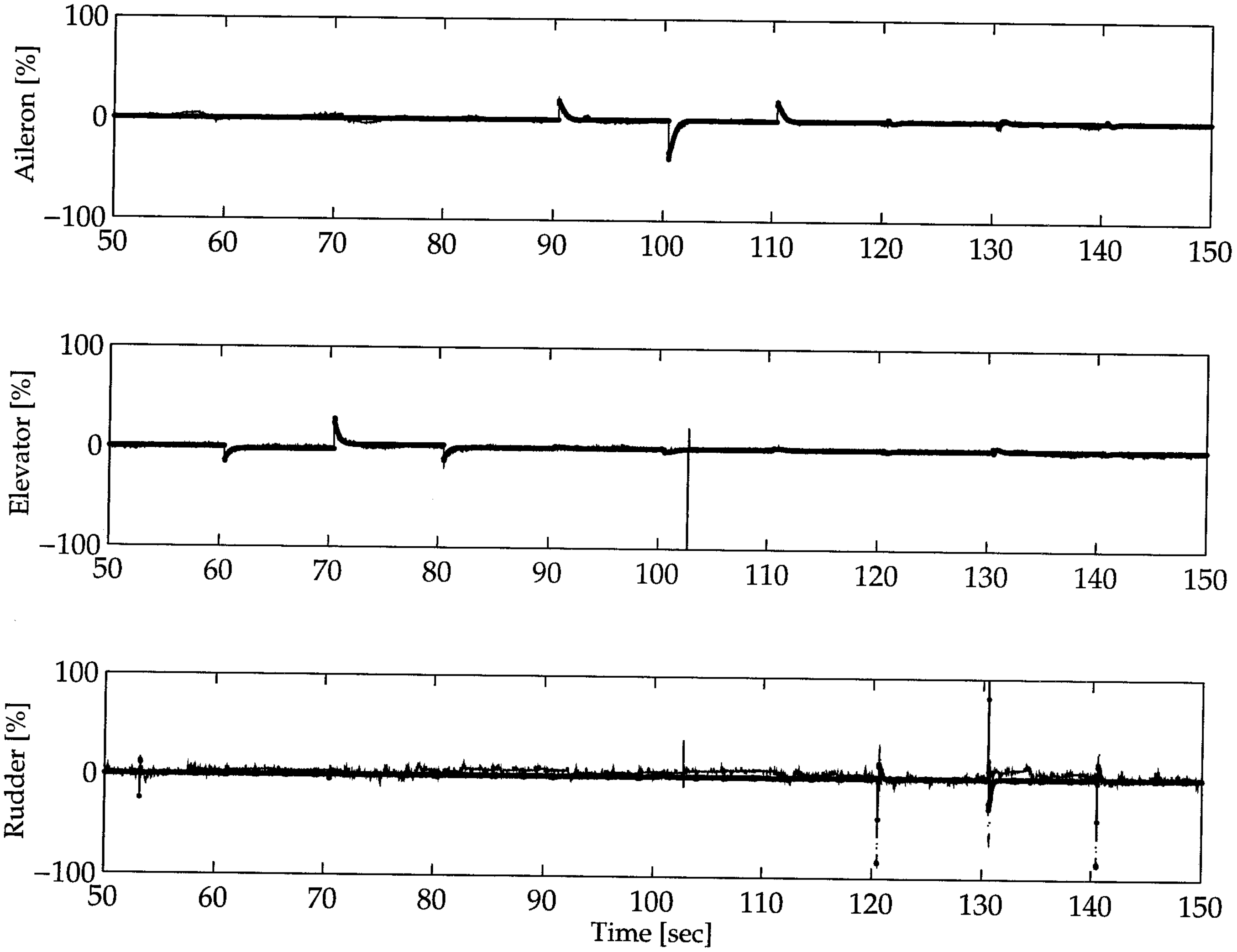

Figure 6.11: Experimental control signals for the LQG controller resulting from step commands: experimental (solid), $\hat{G}_3$ simulation (mixed), $\hat{G}_6$ simulation (dotted).

properties of the LQG controller were tested similarly to the $\mathcal{H}_\infty$ controller. The disturbance was generated by tapping on the boom of the helicopter. The helicopter at hover does not appear to be as stiff as with the $\mathcal{H}_\infty$ controller, and the response was very lightly damped in the pitch axis with ringing taking 3 seconds to die off. The effect of disturbances on the pitch and yaw axes were also considered. The pitch axis performed about as well as the roll axis, while the yaw axis was much inferior. This is due to poor modelling of the yaw axis. Under normal open-loop operating conditions, tremendous pilot workload is necessary to stabilize the hover operating point in the presence of disturbances such as these.

### 6.4.3 Comparison of $\mathcal{H}_\infty$ and LQG controllers

Close examination of the experimental results for both controllers reveals the achieved performances with the $\mathcal{H}_\infty$controller are clearly better. The rise times are almost three times as fast on the pitch axis as those with the LQG controller. This comparison is meaningful as both controllers apply the same magnitude of control inputs: both aileron and elevator use no more than 20% of the maximum servo commands for a 5 degrees change on pitch and roll, and tail rotor collective saturates the yaw axis actuator. These controllers were also tested with 20 degree commands and performed comparably. Refer to Table 6.1 for a summary of the time-domain and frequency-domain properties of the two controllers. The yaw axis was omitted because large oscillations in the experimental responses of both controllers prevented reasonable computation of performance measures.

| | $\mathcal{H}_\infty$ | LQG |
|---|---|---|
| $\phi$ rise time, seconds | 0.54 | 0.54 |
| $\phi$ settling time, seconds | 7.8 | 7.7 |
| $\phi$ overshoot, % | 22 | 30 |
| $\theta$ rise time, seconds | 0.52 | 1.34 |
| $\theta$ settling time, seconds | 3.70 | 5.0 |
| $\theta$ overshoot, % | 20 | 23 |
| Nominal Performance | 0.00004 | 0.0005 |
| Robust Stability | 1.24 | 1.68 |
| Robust Performance | 1.24 | 1.68 |

Table 6.1: Summary of $\mathcal{H}_\infty$and LQG controller performance and robustness measures resulting from experimental implementation.

In simulations, the yaw axis behaved just as well as roll and pitch, however, when implemented, the performance on the yaw axis was significantly less than that in the simulations. This is due to a poor model for the yaw axis. We suspect that an important reason for the poor yaw model is an asymmetry and nonlinearity in the yaw actuation. It is much easier to yaw in the direction opposite to the rotation direction of the main rotor as a result of the torque the main rotor places on the helicopter. Due to the nonlinearity in the actuation of the yaw axis, saturating the yaw actuator appears to be a good strategy to control this axis. The yaw response with the $\mathcal{H}_\infty$ controller is faster and also highly damped, resulting in much faster rejection of disturbances and less cross-coupling between axes. The steady-state tracking of both controllers was comparable. Both were able to keep the helicopter from yawing when the main rotor was powered down ("landed"). We also studied the effect of adding an integrator to the pilot yaw command, so that the pilot effectively commands the yaw rate, $r$. This controller had similar performance to those which command setpoints in the attitude. Comparing simulation with implementation on the helicopter, it appears that if the weight on the yaw axis is rolled off too early, i.e., the yaw axis is too slow, then there is significantly more ripple and loss of control authority.

For both the $\mathcal{H}_\infty$ and LQG controller, the closed-loop system was analyzed with respect to structured uncertainty using $\mu$ [55, 20], with the structure given by

$$\mathbf{\Delta} = \left\{ \begin{bmatrix} \Delta_p & \\ & \Delta_m \end{bmatrix} : \Delta_p,\ \Delta_m \in C^{3\text{x}3} \right\}, \tag{6.4}$$

where $\Delta_p$ and $\Delta_m$ are norm bounded perturbations accounting for performance and uncertainty, respectively. The bounds for $\mu$ are plotted in Figure 6.12.

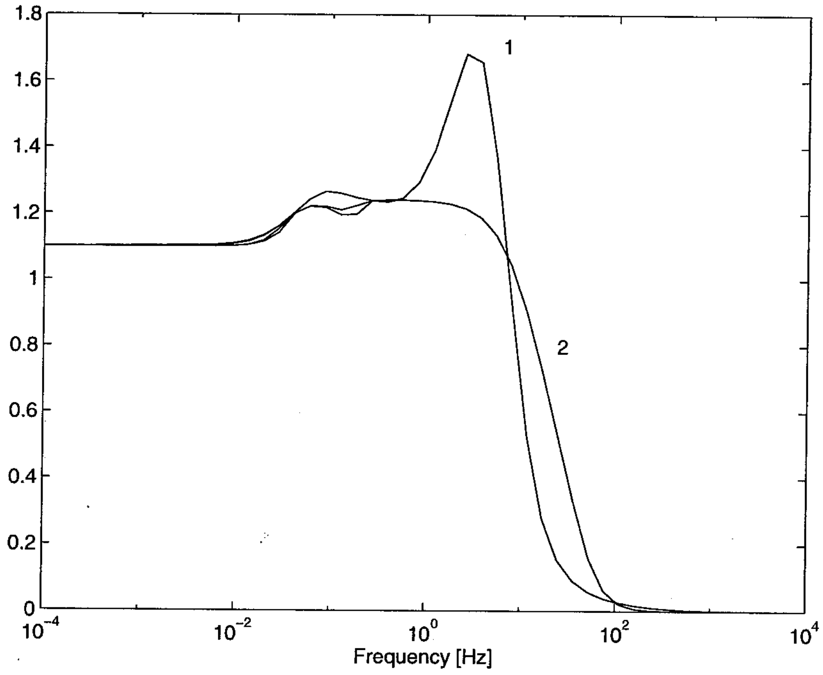

Figure 6.12: Closed-loop $\mu$-analysis: LQG controller (1) and $\mathcal{H}_\infty$ controller (2).

The peak values of $\mu$ for the $\mathcal{H}_\infty$ and LQG controllers are approximately 1.24 and 1.68, respectively (Figure 6.12). This implies that the closed-loop system satisfies robust performance for all perturbations satisfying $\|\Delta\|_\infty \leq 0.8$, for the $\mathcal{H}_\infty$controller, and $\|\Delta\|_\infty \leq 0.6$, for the LQG controller. In other words, the $\mathcal{H}_\infty$ controller is much more robust than the LQG controller, as expected.

As the $\mu$ plots for both robust stability and robust performance lie on top of one another, and those for nominal performance are so low that they are not visible, it is clear that uncertainty is dominant and performance is not a limiting factor. This suggests that nominal performance can be increased, but unfortunately this causes a saturation. Thus a tradeoff between robustness and performance is not viable for this setup.

## 6.5 Summary

In this chapter, an $\mathcal{H}_\infty$ and an LQG controller based on an identified model of a model helicopter in hover were designed, implemented, and compared. Both controllers were designed with the same nominal model, with an uncertainty description included for the design of the $\mathcal{H}_\infty$ controller. The design objective of both controllers was the same: to stabilize an open-loop unstable plant in the presence of a large amount of uncertainty and to use a fast yaw response to decouple the modes and achieve fast but damped response to setpoint changes in attitude in order to achieve zero steady-state tracking error and fast rejection of impulsive disturbances.

The $\mathcal{H}_\infty$ controller achieved faster responses on the roll and pitch axes and had greater damping in the yaw axis. This proved critical for disturbance rejection, resulting in faster rejection of disturbances and less cross-coupling between axes.

The application of robust control design techniques using an adequate uncertainty model was decisive. In particular, with the $\mathcal{H}_\infty$ design, performance was substantially increased on the pitch axis without introducing instability. This was not the case with the LQG design.

The main advantage of the $\mathcal{H}_\infty$ controller is that when implemented on the model helicopter, it behaved much more closely to the simulated responses and introduced more damping than the LQG controller. This advantage results from using a robust control design technique which explicitly takes into account an appropriate description for model errors. The standard LQG technique is not suitable to solve this stabilization problem. Note that to overcome this deficiency, frequency dependent weights could be introduced in (6.3) as described in [29]. However, this technique is *ad hoc.*

The design model was identified from experimental data using a parametric MIMO state-space model. This is particularly well-suited to the development of LPV models to cover a larger flight envelope than hover. By incorporating the rotor speed regulator, discussed in Appendix 6.A, LPV vertical flight models for the model helicopter could be developed. Future work will focus on developing and using such LPV models to control the helicopter away from hover.

## 6.A Appendix: rotor speed regulator

In this appendix, the feasibility of using the tail rotor sensor and throttle actuator to provide a regulator for rotor speed is demonstrated. This work was first published by Van Nieuwstadt and Morris [70]. By designing a SISO controller from the tail rotor sensor to the throttle actuator, both the main rotor and tail rotor speeds can be regulated, which allows the collective and rudder to be used as independent controls for main rotor and tail rotor thrust, respectively.

Direct regulation of the rotor velocity is a necessary first step to design controllers operating over a larger flight envelope than hover. As discussed in Chapter 3, by controlling the rotational velocity of the main rotor, a substantial reduction of the nonlinear and gyroscopic effects of the rotor disk can be obtained. In addition, the collective and throttle can be decoupled, simplifying the design of free flight controllers.

Implementation of rotor speed controllers can also simplify the identification process by increasing the static stability of parameters in the $B$ matrix of the linearized helicopter model. This is important when identifying an LTI model with constant parameters. Scheduling the helicopter model as a function of the rotor speed can also be done so that controllers can be designed which stabilize the helicopter during take-off and landing, when the thrust generated by the rotor disk undergoes dramatic changes.

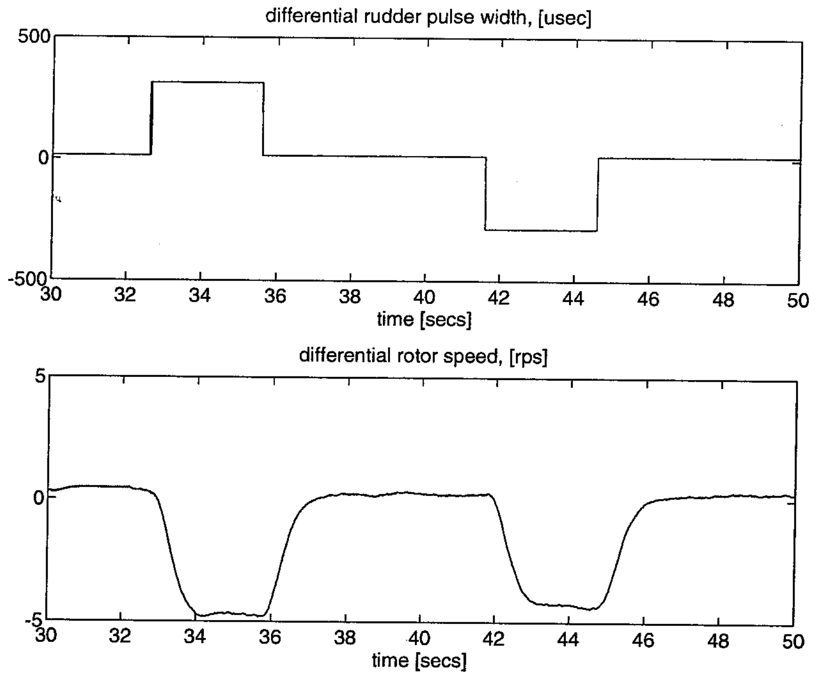

Figure 6.13: Effect of a disturbance on the open-loop main rotor. Top: differential rudder pulse width in microseconds. Bottom: rotor speed deviation in RPS.

The control objective is to maintain constant rotor speed under varying loads on the main rotor. The principal actions which change the main rotor load are changes in both the main rotor and tail rotor collective. The disturbances are naturally modelled as additive perturbations on the plant input. An $\mathcal{H}_\infty$ design technique is used to design this rotor speed regulator.

To simulate the effect of disturbances (or varying rotor loads), a torque was generated on the yaw axis using the rudder actuator. Figure 6.13 shows experimental data representing how the rotor speed decreases if this torque is applied when the system runs open-loop. A maximum deviation of 4.8 RPS occurred. In this plot, and in subsequent plots, the rotor speed is the deviation from the nominal value of 87 RPS, and the rudder PWM is the deviation in microseconds from the nominal 1.5 ms. Both a positive and negative yaw torque will change the load on the rotor, and therefore change the rotor speed.

As expected, the rotor speed is insensitive to tilting of the rotor disk, that is, pitch and roll motions. A change in the collective (main rotor blade pitch angle) had an effect of similar magnitude to the yaw disturbance: a decrease in collective increases the rotor speed and an increase in collective decreases the rotor speed.

Recall from Appendix 4.A that the rotor was modelled as a linear plant, $P$, with a nonlinear input transform, $G$, and a second order smoothing filter at the output. The filter is implemented in discrete-time as a front-end to the rotor speed sensor. Figure 6.14 shows the interconnection used to design the controller, $K$. The controller, $K$, was designed for the linear model, $P$, and subsequently augmented with the inverse static gain $G^{-1}$, resulting in the actual controller used for implementation, $K_R = G^{-1}K$.

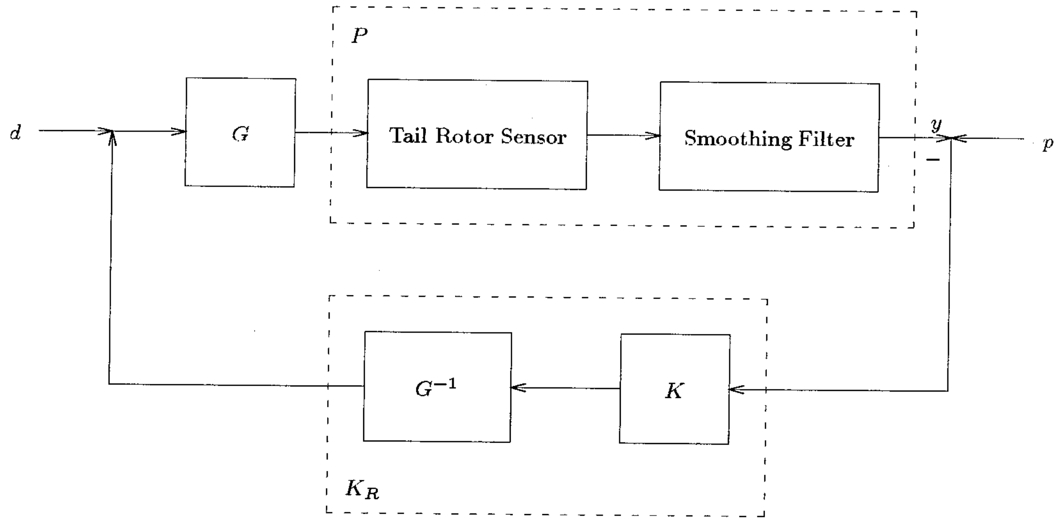

Figure 6.14: Structure of the rotor speed regulator.

Referring to Figure 6.15, the control objective is to provide disturbance rejection, i.e., to minimize the transfer function from $d$ to $y$. $p$ represents setpoint commands provided by a pilot. The controller must be designed so that it has no frequency content above the cut-off frequency of the preconditioning filter for the sensor (see Appendix 3.A). To design the $\mathcal{H}_\infty$ controller, the model is converted to continuous-time. Figure 6.15 shows the interconnection structure used for the $\mathcal{H}_\infty$ controller. $W_k$ weights the controller frequency content, and $W_p$ weights the performance. The

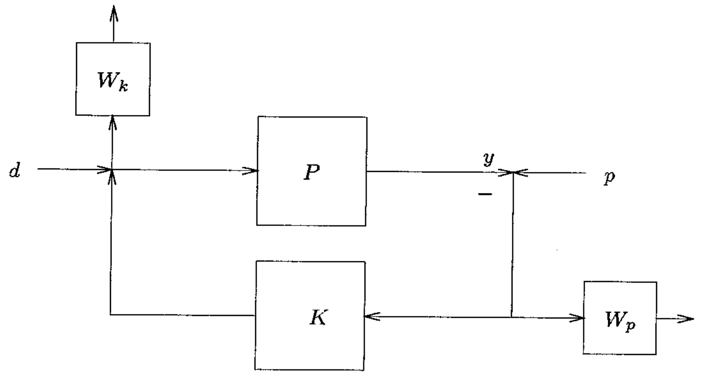

Figure 6.15: $\mathcal{H}_\infty$ interconnection for the rotor speed regulator.

weighting functions are used to tune the controller to achieve the desired closed-loop properties. We have to deal with actuator saturation, and therefore need a means of limiting the control effort. By making $W_k$ large for high frequencies, we make the controller insensitive over that range. In this case, $W_k$ should be made large above the filter cut-off frequency. $W_k$ took the form $W_k = k_k \frac{s+f_k}{s+100f_k}$ where $k_k$ and $f_k$ are tuning parameters. The weight $W_p$ is large for low frequencies, where good performance is necessary, and rolls off at higher frequencies, taking the form $W_p = k_p \frac{s+100f_p}{s+f_p}$. The values $k_k = 0.001, f_k = 0.005, k_p = 1.0, f_p = 0.005$ were used. Refer to Figure 6.16 for a graph of the weighting functions, where the dotted line corresponds to $W_p$ and the solid line to $W_k$.

The rotor speed regulator was designed using standard continuous-time $\mathcal{H}_\infty$ sub-optimal synthesis. The response of the closed-loop system is shown in Figure 6.17.

The resulting closed-loop system does not have an infinity norm less than 1, but since $W_k$ is not interpreted as plant uncertainty, this can be ignored. Note that the closed-loop system exhibits ringing at about 0.6 Hz. This most likely results from the sensor lacking high frequency content, thus limiting the controller performance at high frequencies. The $\mathcal{H}_\infty$ controller reduced the maximum deviation from 4.8 RPS (open-loop) to 1.8 RPS (closed-loop). This reduction is substantial, considering that the nonlinear effect we would like to minimize is proportional to the square of the rotor speed.

Future work will focus on modelling the helicopter with the rotor speed controller in place. This should result in better linear models, due to the reduction in the nonlinear effect of the rotor dynamics. Integration of a rotor speed controller with a hover controller should increase robustness and allow for greater performance, resulting from the use of the collective as an additional control input.

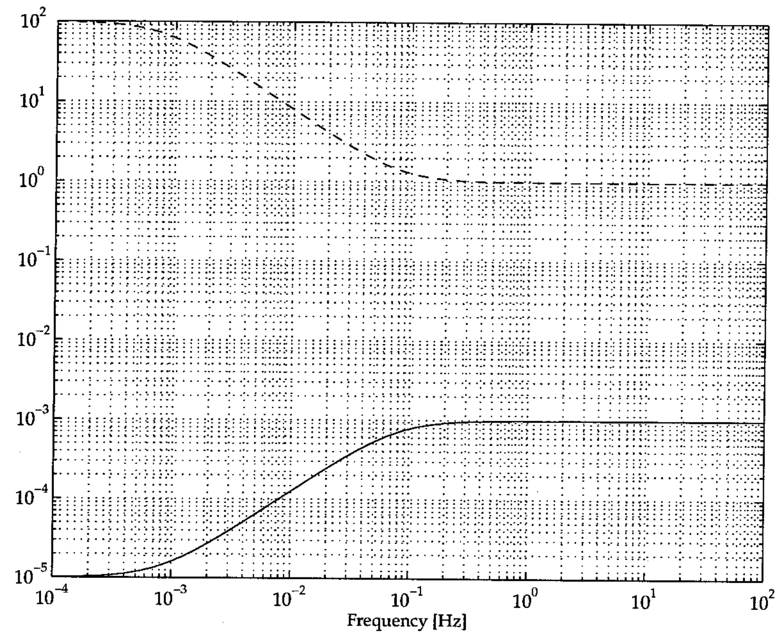

Figure 6.16: Rotor speed regulator weighting functions: $W_k$ solid, $W_p$ dashed.

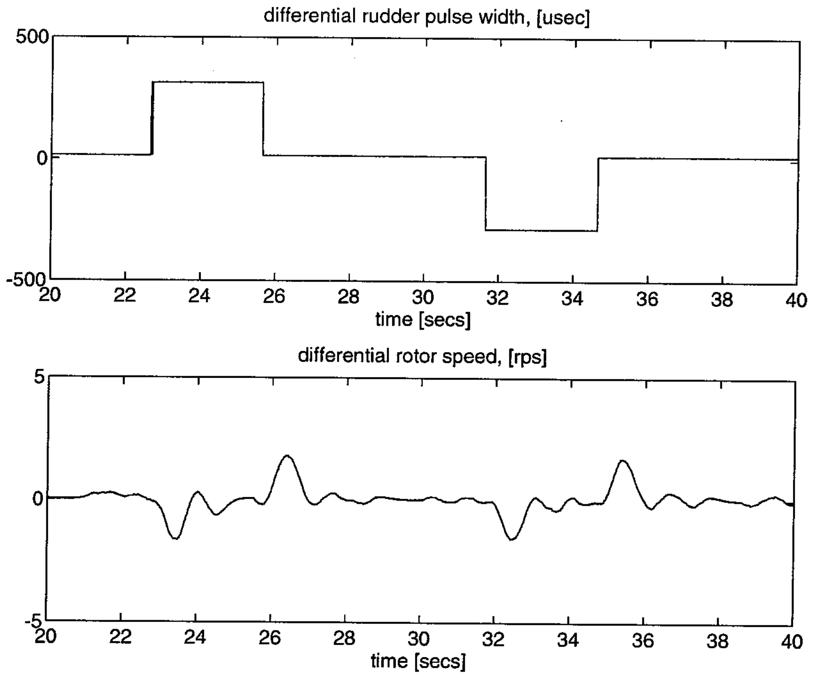

Figure 6.17: Effect of a disturbance on the closed-loop main rotor. Top: differential rudder pulse width in microseconds. Bottom: rotor speed deviation in RPS.

# Chapter 7
# Frequency-domain Model Validation

A robust control model is a model for a system which contains not only a model of the system dynamics, but an uncertainty and noise description as well. Models of this type are used extensively in the $\mathcal{H}_\infty$ and $\mu$ design framework [55]. The "model validation" problem was originally formulated by Smith and Doyle to provide a connection between a general LFT robust model and data measured from a physical system [63]. Most work in model validation does not address the case of general LFT robust models, but rather a restriction to the case where the uncertainty enters in an affine way. All model validation results presented in this dissertation are applicable to the case of general LFT robust models. Model validation seeks to answer the question: "Does the robust model account for the measurements from the physical system?"

A constant matrix version of the model validation problem was formulated as a generalization of the structured singular value, $\mu_g$, by Newlin and Smith [52]. The main result is that if $\mu_g \geq \gamma$, the data and uncertain model are consistent when the norms of noise, disturbances, and uncertainty are of size less than or equal to $1/\gamma$. An upper bound for $\mu_g$ using linear matrix inequalities was formulated by Newlin and Smith [52], and a general algorithm to compute this upper bound was developed by Morris and Newlin [47]. The upper bound is tight in the sense that equality between the upper bound and $\mu_g$ is achieved for certain classes of uncertainty [52]. In the following, this upper bound of $\mu_g$ will be denoted by $\gamma_{ub}$. Additionally, a modified power algorithm for computing a lower bound of $\mu_g$ was developed by Newlin and Morris [51], which will be denoted by $\gamma_{lb}$. Newlin provides a more general treatment of the theoretical background for the model validation lower bound [50].

Two steps are implicit when applying the constant matrix model validation results to a mathematical robust model and experimental data. First, if the experimental data are in the time-domain, it must be transformed into the frequency-domain. Second, a frequency sweep is used, where at each frequency the robust model and the frequency-domain data are reduced to a constant matrix problem. It is not necessary to utilize all available data in the frequency sweep; a subset of the data corresponding to the important frequency points can be used instead in a manner similar to $\mu$-analysis and $\mu$-synthesis [20].

There are two types of information that the model validation technique provides:

($i$) magnitude, and ($ii$) frequency shape. The magnitude gives an indication of the size of the weights on the noise and uncertainty necessary to achieve data consistency. The frequency shape of the bound provides information on which frequency regions are posing problems and what the frequency shape of the weights should be. This suggests the following iterative methodology: use the frequency varying upper (lower) bound(s) to make tradeoffs between the frequency shape and magnitude of the noise and uncertainty weights. The tuning heuristic would be to minimize the overall size of uncertainty and noise by trading off the magnitude between each of them. *A priori* information about the system can be incorporated into the tuning heuristic to make more accurate and physically motivated tradeoffs.

In addition to providing a bound for $\mu_g$, the lower bound (and upper bound in special cases) provides a perturbation from the uncertainty set achieving the bound. In the case of parametric uncertainty, these perturbations can be used to determine the validity of the nominal model. In particular, parameters in the nominal model can be iteratively "tuned" by the value of the parametric perturbation at those frequencies where the lower bound is a minimum. This "tuning" provides the capability to decrease the magnitude of the scaling weights of the parametric uncertainty. This provides the previously missing connection between experimental data and both nominal and uncertainty modelling.

A generalization of the structured singular value, $\mu_g$, and its application to constant matrix model validation along with constant matrix algorithms for computing lower and upper bounds for $\mu_g$ are presented in Section 7.1. Section 7.2 formulates the application of these constant matrix results to model validation in the frequency-domain. A discussion of a general methodology for iteratively tuning parameters in the nominal model and weights in the robust model is in Section 7.3. Section 7.4 discusses practical limitations of the model validation software. A simulated spring-mass system with parametric uncertainty in the damping and stiffness coefficients provides an example of the use of both the upper and lower model validation bounds in Section 7.5. Finally, the model validation methods are used to analyze the validity of a robust model used in a ducted fan experiment in Section 7.6.

## 7.1 Constant matrix results

The model validation problem is concerned with determining whether a mathematical model is consistent with (or covers) a collection of experimental data. The mathematical model consists of a nominal LTI system and uncertainty, generally described by bounded perturbations and denoted by $\Delta \in \mathbf{\Delta}$, acting on the LTI nominal model. Figure 7.1 corresponds to a general interconnection for a robust model which is suitable for model validation.

Here $u$ is the control input, $y$ is the measurement, and $\Delta$ represents the model uncertainty, which is a norm-bounded structured perturbation, consisting of possibly many different blocks. Measurement noise corresponds to $n$, which is explicitly weighted by $W_n$, and there are additional exogenous disturbances $d$ acting directly

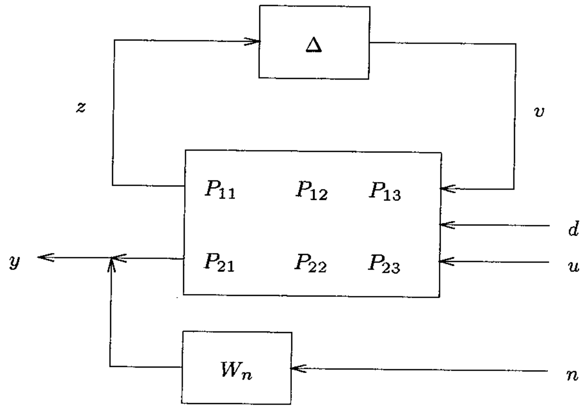

Figure 7.1: Block diagram of the model validation problem. $P$ represents the model augmented with uncertainty and disturbance weights, uncertainty $\Delta \in \mathbf{\Delta}$, noise weight $W_n$, exogenous disturbance $d$, input $u$, measurement $y$, and measurement noise $n$.

on the plant $P$. Uncertainty and disturbance weights are implicitly contained in $P$. The dimensions of the measurements and disturbances will be denoted by $n_m$ and $n_d$, respectively. Note that the noise is of the same size as the measurements. The term "robust model" will refer to a system description of the form illustrated in Figure 7.1.

Following is a definition of the statement "the robust model and data are consistent", and particularly when they are consistent for a specific value of $\gamma$.

**Definition 7.1.1** *A robust model and data are $\gamma$-consistent if $\exists \Delta \in \mathbf{\Delta}$, $\|\Delta\| \leq 1/\gamma$, $\|d\| \leq 1/\gamma$ and $\|n\| \leq 1/\gamma$ such that $y = W_n n + (\Delta \star P)\begin{bmatrix} d \\ u \end{bmatrix}$, where $P$, $W_n$, $d$, $n$, $u$ and $y$ are as in Figure 7.1, with $u$ and $y$ experimental data, $d$ exogenous disturbance and $n$ measurement noise. $\mathbf{\Delta}$ describes a set of structured uncertainty.*

## 7.1.1 Review of $\mu_g$

This section reviews $\mu_g$ as presented by Newlin and Smith [52]. $\mu_g$ is an extension of the $\mu$ framework where perturbation blocks are divided into two sets: one satisfying maximum norm contraints (similar to $\mu$) with the other satisfying minimum norm constraints. It will be seen that such a formulation solves the model validation problem.

Following is a reformulation of several theorems relating to model validation in the $\mu_g$ framework presented [52]. The proofs are omitted, and the reader is referred to [52] for proofs and further details. Note that in [52] all results are scaled with noise, disturbances, and uncertainty of size 1. In this dissertation the relative size of

$\mu_g$ and the size of the noise, disturbances, and uncertainty is explicitly contained in $\gamma$.

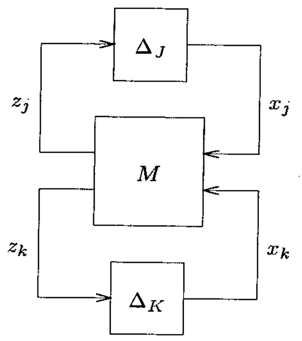

Figure 7.2: Interconnection structure for $\mu_g$. $M$ is a constant matrix and $\Delta_J$ and $\Delta_K$ are special augmented uncertainties.

**Definition 7.1.2** *Consider the interconnection in Figure 7.2 where the uncertainty set* $\mathbf{\Delta}$ *is divided into two pieces defined by*

$$\mathbf{\Delta} = \begin{bmatrix} \mathbf{\Delta_J} & 0 \\ 0 & \mathbf{\Delta_K} \end{bmatrix},$$

*where the block structures of* $\mathbf{\Delta_J}$ *and* $\mathbf{\Delta_K}$ *are defined by the m-tuples* $\mathcal{K}_J$ *and* $\mathcal{K}_K$*, respectively. Let* $M$ *be partitioned according to the block structure* $\mathbf{\Delta}$*. Then*

$$\mu_g(M) \equiv \max_{\|x\|=1} \left\{ \gamma : \begin{array}{l} \|x_j\|\gamma \leq \|z_j\|, \ \forall j \in J \\ \|x_k\| \geq \|z_k\|\gamma, \ \forall k \in K \end{array} \right\}.$$

For a special block structure $\mu_g$ solves the model validation problem. Referring to Figure 7.1, define the constant matrix $\tilde{P}$, formed from $P$, $W_n$, input vector $u$ and measurement vector $y$, as

$$\tilde{P} = \begin{bmatrix} P_{11} & P_{12} & P_{13}u \\ 0 & 0 & 1 \\ W_n^{-1}P_{21} & W_n^{-1}P_{22} & W_n^{-1}(P_{23}u - y) \end{bmatrix}. \tag{7.1}$$

Because $W_n$ is a stable bi-proper weighting function its inverse will always exist. Define $\mathbf{\Delta_J}$, $\mathbf{\Delta_K}$, and $\tilde{\mathbf{\Delta}}$ as

$$\begin{aligned} \mathbf{\Delta_J} &= \left\{ \text{diag}(\Delta, \Delta_d) : \Delta \in \mathbf{\Delta}, \Delta_d \in \mathbb{C}^{n_d \times 1} \right\} \\ \mathbf{\Delta_K} &= \left\{ \Delta_n : \Delta_n \in \mathbb{C}^{1 \times n_m} \right\} \\ \tilde{\mathbf{\Delta}} &= \text{diag}(\mathbf{\Delta_J}, \mathbf{\Delta_K}). \end{aligned} \tag{7.2}$$

The following theorem shows that $\mu_g$ solves the constant matrix model validation problem. Recall that in (7.2) $\mathbf{\Delta}$ defines the uncertainty description and $\tilde{\mathbf{\Delta}}$ defines the structure for the corresponding model validation problem.

**Theorem 7.1.1** *Let $\tilde{P}$ be defined as in (7.1), with $\tilde{\Delta}$ a member of some norm-bounded structured uncertainty set $\tilde{\mathbf{\Delta}}$ as defined in (7.2). Then $\mu_g(\tilde{P}) \geq \gamma$ if and only if the robust model and data are $\gamma$-consistent.*

Because $\mu_g$ is not easily computable in general, it is necessary to develop computable bounds. A lower bound for $\mu_g$ is presented in Section 7.1.2 and an upper bound in Section 7.1.3.

### 7.1.2 Constant matrix model validation lower bound

A power method for computing the lower bound for complex $\mu$ was presented by Packard *et al.* [56]. This method was extended to the mixed $\mu$ problem by Young [75]. The model validation lower bound can be computed using modified power algorithm techniques. The initial power algorithm method for computing the model validation lower bound was developed by Newlin for a simplified block structure [51]. An extension is described herein.

The power algorithm for the model validation lower bound is developed in a similar fashion to the power algorithm for $\mu$ [75]. Figure 7.3 shows a block diagram of the general constant matrix problem, where $M$ denotes an appropriately partitioned constant matrix, $\delta^r$, $\delta^c$, and $\Delta^C$ are real repeated scalar, complex repeated scalar, and complex full block uncertainties, respectively, and $\Delta_n$ corresponds to the noise. Note that for the interconnection in Figure 7.3 it is assumed that $m_r = m_c = m_C = 1$. This is not restrictive, since the algorithm is developed such that the uncertainty is treated on a block by block basis.

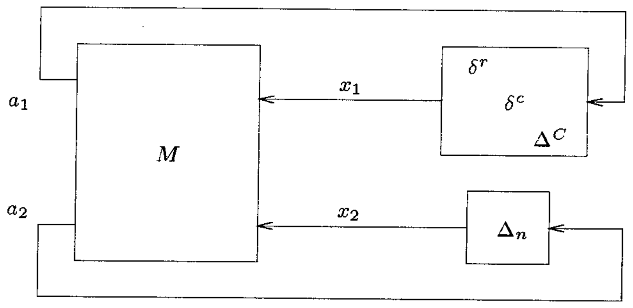

Figure 7.3: Block diagram for the model validation lower bound. $M$ is a constant matrix, $\delta^r$, $\delta^c$, and $\Delta^C$ are uncertainties, and $\Delta_n$ corresponds to noise.

Figure 7.3 is applicable to the standard mixed $\mu$ lower bound problem if blocks corresponding to $\Delta_n$ are ignored. The following equalities result from Figure 7.3

$$\begin{aligned} a &= Mx \\ x &= \Delta a \\ y^* &= z^* M \\ z^* &= y^* \Delta, \end{aligned} \tag{7.3}$$

where $a = [a_1^T, a_2^T]^T$, $x = [x_1^T, x_2^T]^T$, and $\Delta = \text{diag}(\delta^r, \delta^c, \Delta^C, \Delta_n)$. Note that $y$ and $z$ are defined in a similar fashion to $a$ and $x$, respectively, for the dual problem with $M^*$.

**Definition 7.1.3** *Referring to Figure 7.3, any vectors $x$, $a$, $z$, and $y$ satisfying the equalities in (7.3), for a specified uncertainty description, $\mathbf{\Delta}$, are said to be feasible.*

The partitioning of the vectors $a$, $x$, $y$, and $z$ imparts a natural partition of $M$, given by

$$M = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}. \tag{7.4}$$

Given the interconnection in Figure 7.3, and partitioning (7.3) appropriately for the perturbation block structure, the following assignments can be made:

$$\begin{aligned} x_1^r &= \delta^r a_1^r \\ x_1^c &= \delta^c a_1^c \\ x_1^C &= \Delta^C a_1^C, \end{aligned} \tag{7.5}$$

where $x_1 = [x_1^{rT}, x_1^{cT}, x_1^{CT}]^T$ and $a_1 = [a_1^{rT}, a_1^{cT}, a_1^{CT}]^T$. (7.5) can then be used to compute a perturbation from feasible vectors at a local maximum, resulting in a solution given by

$$\begin{aligned} \delta^r &= \text{Real}(\frac{a_1^{r*} x_1^r}{a_1^{r*} a_1^r}) \\ \delta^c &= \frac{a_1^{c*} x_1^c}{a_1^{c*} a_1^c} \\ \Delta^C &= \frac{x_1^C a_1^{C*}}{a_1^{C*} a_1^C}, \end{aligned} \tag{7.6}$$

where Real$(\cdot)$ returns the real component of its complex argument.

## Power algorithm for mixed $\mu$

To simplify the development of the model validation lower bound power algorithm, the power algorithm steps applied to the standard mixed $\mu$ case as presented by Young [75] are summarized. The only difference between the model validation lower bound and the standard $\mu$ lower bound is the presence of the noise block, $\Delta_n$.

The $a_1$ and $x_1$ ($y_1$ and $z_1$) vectors in Figure 7.3 are partitioned to be compatible with the block structure $\delta^r$, $\delta^c$, and $\Delta^C$. Note that the "1" subscript, e.g., $x_1^r$, will be dropped from all vectors in this section. The resulting power algorithm iteration is given by

$$
\begin{gathered}
\tilde{\beta}_{k+1} a_{k+1} = M x_k \\
z^r_{k+1} = \tilde{q}_{k+1} y^r_k \quad z^c_{k+1} = \frac{y^{c*}_k a^c_{k+1}}{|y^{c*}_k a^c_{k+1}|} y^c_k \quad z^C_{k+1} = \frac{|y^C_k|}{|a^C_{k+1}|} a^C_{k+1} \\
\hat{\beta}_{k+1} y_{k+1} = M^* z_{k+1} \\
x^r_{k+1} = \hat{q}_{k+1} a^r_{k+1} \quad x^c_{k+1} = \frac{a^c_{k+1}{}^* y^c_{k+1}}{|a^c_{k+1}{}^* y^c_{k+1}|} a^c_{k+1} \quad x^C_{k+1} = \frac{|a^C_{k+1}|}{|y^C_{k+1}|} y^C_{k+1},
\end{gathered}
\tag{7.7}
$$

where $|\cdot|$ denotes the norm of its argument, $\tilde{\beta}_{k+1}$ and $\hat{\beta}_{k+1}$ are chosen to be real and positive scalars such that $|a_{k+1}|$ and $|y_{k+1}|$ are unity. Note that the decision to normalize $a_{k+1}$ and $y_{k+1}$ to unity is arbitrary. This will be exploited when developing a comparable algorithm for the model validation problem. The iteration for $\tilde{q}$ and $\hat{q}$ is given by

$$
\begin{gathered}
\tilde{\alpha}_{k+1} = \mathrm{Sgn}(\hat{q}_k) \frac{|x^r_k|}{|a^r_{k+1}|} + \mathrm{Real}(a^r_{k+1}{}^* y^r_k) \\
\text{If } |\tilde{\alpha}_{k+1}| \geq 1 \text{ then } \tilde{q}_{k+1} = \mathrm{Sgn}(\tilde{\alpha}_{k+1}) \text{ else } \tilde{q}_{k+1} = \tilde{\alpha}_{k+1} \\
\hat{\alpha}_{k+1} = \mathrm{Sgn}(\tilde{q}_{k+1}) \frac{|x^r_k|}{|a^r_{k+1}|} + \mathrm{Real}(a^r_{k+1}{}^* y^r_{k+1}) \\
\text{If } |\hat{\alpha}_{k+1}| \geq 1 \text{ then } \hat{q}_{k+1} = \mathrm{Sgn}(\hat{\alpha}_{k+1}) \text{ else } \hat{q}_{k+1} = \hat{\alpha}_{k+1},
\end{gathered}
\tag{7.8}
$$

where $\mathrm{Sgn}(\cdot)$ returns the sign of its argument.

If the iterations defined by (7.7) and (7.8) converge to some equilibrium point and $\tilde{\beta} = \hat{\beta}$ then the vectors $x$, $a$, $y$, and $z$ achieve the mixed $\mu$ lower bound. Further extensions and refinements to the algorithm are contained in [75].

## Model validation lower bound power algorithm

The model validation lower bound power algorithm, outlined in Table 7.1, consists of several steps which are virtually identical to the standard mixed $\mu$ power iteration; however, because of the addition of the noise blocks, there is an inner implicit loop.

The algorithm is developed on a block by block basis. Because the model validation problem is naturally non-square (consider the dimension of $\Delta_n$), index sets are used to partition the vectors to the appropriate dimension. The algorithm presented is based on the assumption that the blocks are square; it is a simple modification to use index sets to account for non-square blocks. Further, note that the algorithm has been developed with the normalizations

$$
\begin{aligned}
a_2 &= \frac{z_2}{|z_2|} \\
y_2 &= 1.
\end{aligned}
\tag{7.9}
$$

Thus, if $x_2 = 1$ then $a_2 = -n$ and $\Delta_n = -\frac{n^*}{n^*n}$. Based on the normalization (7.9), it is simple to solve for the noise signal. Referring to Figure 7.3, $z_2 = \Delta_n^* y_2$. Substituting the normalization in (7.9) yields $z_2 = \Delta_n^* = -\frac{n}{n^*n}$. Solving for $n$ results in

$$n = -\frac{z_2}{z_2^* z_2}. \tag{7.10}$$

The lower bound solution for $n$ in (7.10) will be used to compute the noise signal which achieves the lower bound. Comparing the size of the noise and uncertainty which achieve the lower bound is useful when designing robust models.

Specific equations from Table 7.1 are included in (7.11-7.13). Note that for each type of uncertainty ($\delta^r$, $\delta^c$, and $\Delta^C$) if there is more than one block then the equations presented below will be performed block by block.

1. Compute $z_{1k+1}$ block by block, see (7.11).
2. Assign $y_{2k+1} = 1$.
3. Compute $z_{2k+1}$ as follows:
    - Define $z_2 = z_a + z_b$ where $z_a$ is aligned with $m_{12}^*$ and $z_b$ is aligned with $m_{22}^*$.
    - Assign $z_a = \frac{m_{22}}{m_{22}^* m_{22}}(y_{2k+1} - m_{12}^* z_{1k+1})$.
    - Compute $z_b$ implicitly, see (7.12).
    - Assign $z_{2k+1} = z_a + z_b$.
4. Assign $y_{1k+1} = m_{11}^* z_{1k+1} + m_{21}^* z_{2k+1}$.
5. Compute $x_{1k+1}$ block by block, see (7.13).
6. Assign $a_{2k+1} = \frac{z_{2k+1}}{|z_{2k+1}|}$.
7. Assign $x_{2k+1} = \frac{m_{22}^*}{m_{22}^* m_{22}}(a_{2k+1} - m_{21} x_{1k+1})$.
8. Assign $a_{1k+1} = m_{11} x_{1k+1} + m_{12} x_{2k+1}$.

Table 7.1: An iteration of the model validation lower bound power algorithm.

The steps for the $k^{th}$ iteration of the power algorithm for computing $z_1$ are given by

$$\begin{aligned}
\tilde{\alpha}_{k+1} &= \mathrm{Sgn}(\hat{q}_k)\frac{|x^r_{1k}|}{|a^r_{1k}|} + \mathrm{Real}(a^{r*}_{1k}y^r_{1k}) \\
\text{If } |\tilde{\alpha}_{k+1}| \geq 1 &\text{ then } \tilde{q}_{k+1} = \mathrm{Sgn}(\tilde{\alpha}_{k+1}) \text{ else } \tilde{q}_{k+1} = \tilde{\alpha}_{k+1} \\
z^r_{1k+1} &= \tilde{q}_{k+1}y^r_{1k} \\
z^c_{1k+1} &= \frac{y^{c*}_{1k}a^c_{1k}}{|y^{c*}_{1k}a^c_{1k}|}y^c_{1k} \\
z^C_{1\,k+1} &= \frac{|y^C_{1\,k}|}{|a^C_{1\,k}|}a^C_{1\,k} \\
z_{1k+1} &= -\frac{z_{1k+1}}{|z_{2k}|\mathrm{Sgn}(x_{2k})}.
\end{aligned} \tag{7.11}$$

The steps for the $i^{th}$ iteration of the implicit loop, used to compute $z_b$ are outlined in (7.12). This implicit loop is performed for each iteration in the power algorithm. For those vectors which do not change in the implicit loop the power algorithm iteration subscript, $k$, is dropped. The iteration is defined by

$$\begin{aligned}
\alpha_{i+1} &= \mathrm{Sgn}(\tilde{q})\frac{|x^r_1|}{|a^r_1|} + \mathrm{Real}(a^{r*}_1 y^r_{1i}) \\
\text{If } |\alpha_{i+1}| \geq 1 &\text{ then } q_{i+1} = \mathrm{Sgn}(\alpha_{i+1}) \text{ else } q_{i+1} = \alpha_{i+1} \\
v^r_i &= -q_{i+1}\frac{|z_{2i}|}{|x_2|}a^r_1 \\
I^r_i &= 0 \\
v^c_i &= -\frac{a^{c*}_1 y^c_1}{|a^{c*}_1 y^c_1|}a^c_1 \\
I^c_i &= 0 \\
\xi &= -\frac{|a^C_1||z^C_{2\,i}|}{|y^C_1||x^C_2|} \\
v^C_i &= \xi(m^*_{11}z_1 + m^*_{21}z_a) \\
I^C_i &= -\xi I_{\dim(\Delta^C)} \\
z_{bi+1} &= (I_{n_m} + Pm_{21}I_i m^*_{21})/(Pm_{21}v_i),
\end{aligned} \tag{7.12}$$

where $v_i = [v^{rT}_i, v^{cT}_i, v^{CT}_i]^T$, $I_i = \mathrm{diag}(I^r_i, I^c_i, I^C_i)$, $P = I_{n_m} - \frac{m_{22}m^*_{22}}{m^*_{22}m_{22}}$, and $y = A/x$ denotes the least squares solution such that $\|Ay - x\|_2$ is minimized.

The steps for the $k^{th}$ iteration of the power algorithm for computing $x_1$ are given by

$$\begin{aligned}
\hat{\alpha}_{k+1} &= \mathrm{Sgn}(\tilde{q}_{k+1})\frac{|x^r_{1k}|}{|a^r_{1k}|} + \mathrm{Real}(a^{r*}_{1k}y^r_{1k}) \\
\text{If } |\hat{\alpha}_{k+1}| \geq 1 &\text{ then } \hat{q}_{k+1} = \mathrm{Sgn}(\hat{\alpha}_{k+1}) \text{ else } \hat{q}_{k+1} = \hat{\alpha}_{k+1} \\
x^r_{1k+1} &= \hat{q}_{k+1}a_{1k} \\
x^c_{1k+1} &= \frac{a^{c*}_{1k}y^c_{1k}}{|a^{c*}_{1k}y^c_{1k}|}a^c_{1k} \\
x^C_{1\;k+1} &= \frac{|a^C_{1\;k}|}{|y^C_{1\;k}|}y^C_{1\;k} \\
x_{1k+1} &= -\frac{x_{1k+1}}{|x_{2k}|}.
\end{aligned} \tag{7.13}$$

## Convergence properties of the lower bound

The convergence properties of the lower bound are based on empirical observations having used a variety of different interconnections, including complex full blocks, real and complex repeated scalars, disturbances, and noises. For systems with one measurement the lower bound always converges. It appears that the lower bound generally converges when the uncertainty is restricted to complex full blocks. Adding complex repeated scalars complicates the lower bound somewhat, but has little effect on convergence. Real repeated scalars pose the greatest difficulty for lower bound convergence. Aside from systems with only one measurement, convergence of the lower bound appears to be insensitive to the number of disturbance and noise signals.

| **Blocks** | | **Lower Bound** | | **Gap** | | |
|---|---|---|---|---|---|---|
| Full | Real | N | Converge | mean | $\sigma$ | max |
| 1 | 0 | 1000 | 88% | 0.48% | 0.17% | 0.78% |
| 3 | 0 | 1000 | 91% | 0.45% | 0.79% | 14.38% |
| 1 | 2 | 1050 | 71% | 2.49% | 3.71% | 21.95% |

Table 7.2: Convergence results for the model validation lower bound.

A summary of results obtained testing the lower bound are shown in Table 7.2. N is the number of times the bounds were computed, the data in the Converge column corresponds to how many times the lower bound converged, and for the problems where the lower bound converged, the data in the Gap column shows statistics on the difference ("gap") between the lower and upper bounds, where $\sigma$ is the standard deviation. Each time the bounds were computed, a randomly constructed system was converted to a constant matrix problem at a frequency within its bandwidth. The uncertainty description consisted of a specific number of complex full blocks and real repeated scalars shown in the Blocks column.

In all cases where the lower bound has failed to converge, the problem lies within the implicit inner loop. With more work convergence can be greatly improved; however, as the standard mixed $\mu$ power algorithm does not always converge, there will always be limitations to the lower bound.

### 7.1.3 Constant matrix model validation upper bound

Unlike the standard $\mu$ problem of [76], it cannot be assumed that each block in the uncertainty set, $\tilde{\mathbf{\Delta}}$ defined in (7.2), is square. Let the structure of the model validation set $\tilde{\mathbf{\Delta}}$ be defined by the two $m$-tuples $\mathcal{K}_{row}$ and $\mathcal{K}_{col}$, where $\mathcal{K}_{row}$ denotes the row dimensions of $\tilde{\mathbf{\Delta}}$ and $\mathcal{K}_{col}$ the column dimensions. Then define $D_X$ and $D_Z$ as in the $\boldsymbol{\mathcal{D}}$ scaling sets for the standard $\mu$ problem (5.8) with $\mathcal{K}_{row}$ and $\mathcal{K}_{col}$, respectively. Note that the only difference between $D_X$ and $D_Z$ is their dimension, resulting from non-square complex full blocks in the uncertainty set. $G$ is defined as in the $\boldsymbol{\mathcal{G}}$ scaling sets for the standard $\mu$ problem (5.9) with $\mathcal{K}_{col}$; however, the "0" blocks in (5.9) are omitted. Because of the simplified definition of $G$, it is necessary to define permutation matrices $X$ and $Y$ which map $G$ into the appropriate rows and columns of $\tilde{P}$. Let $n_r$ be the row dimension of $\Delta$. Define $I(\gamma)$ as

$$I(\gamma) = \begin{bmatrix} \gamma I_{n_r+n_d} & 0 \\ 0 & \frac{1}{\gamma} I_{n_m} \end{bmatrix}. \tag{7.14}$$

**Theorem 7.1.2** *Let $\tilde{P}$ be defined as in (7.1), with $\tilde{\Delta}$ a norm-bounded structured uncertainty set, $D_X$, $D_Z$, $G$, $X$, $Y$, and $I(\gamma)$ defined as above, for $\gamma > 0$. If there exist feasible solutions, $D_X$, $D_Z$, and $G$, for the complex-valued hermitian LMI given by*

$$\tilde{P}^* D_Z \tilde{P} + j(XGY\tilde{P} - \tilde{P}^* Y^* G X^*) - I(\gamma)^2 D_X < 0$$
$$\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} D_X \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} < \begin{bmatrix} I_{n_r+n_d} & 0 \\ 0 & 0 \end{bmatrix} D_X \begin{bmatrix} I_{n_r+n_d} & 0 \\ 0 & 0 \end{bmatrix},$$

*then $\mu_g(\tilde{P}) < \gamma$, which implies that the robust model and data are not $\gamma$-consistent.*

Newlin and Smith provide a proof for Theorem 7.1.2 [52]. Note that if $\mu_g < \gamma$, or if any upper bound of $\mu_g$ is less than $\gamma$, then the robust model and data are not $\gamma$-consistent.

The quality of the upper bound given in Theorem 7.1.2 is demonstrated by observing that it is exact for two or fewer complex full blocks in $\mathbf{\Delta}$, when there are no exogenous disturbances, $d$, or one complex full block in $\mathbf{\Delta}$ otherwise [52].

#### Model validation upper bound LMI algorithm

The software for solving the upper bound LMI was written with MATLAB using the LMI Control Toolbox [26, 25]. The LMI Control Toolbox is only capable of solving real-valued LMIs, so the LMI in Theorem 7.1.2 must be reformulated in terms of real matrices.

**Theorem 7.1.3** *Let $\tilde{P}$, $D_X$, $D_Z$, $G$, $X$, $Y$, and $I(\gamma)$ be defined as in Theorem 7.1.2. $D_X$, $D_Z$, and $G$ are feasible solutions to the complex-valued hermitian LMI*

$$\tilde{P}^* D_Z \tilde{P} + j(XGY\tilde{P} - \tilde{P}^* Y^* G X^*) - I(\gamma)^2 D_X < 0$$
$$\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} D_X \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} < \begin{bmatrix} I_{n_r+n_d} & 0 \\ 0 & 0 \end{bmatrix} D_X \begin{bmatrix} I_{n_r+n_d} & 0 \\ 0 & 0 \end{bmatrix}$$

*if and only if $\hat{D}_X$, $\hat{D}_Z$, and $\hat{G}$ are feasible solutions for the real-valued symmetric LMI*

$$\hat{\tilde{P}}^T \hat{D}_Z \hat{\tilde{P}} + J\left(\hat{X}\hat{G}\hat{Y}\hat{\tilde{P}} - \hat{\tilde{P}}^T \hat{Y}^T \hat{G} \hat{X}^T\right) - \widehat{I(\gamma)}^2 \hat{D}_X < 0$$
$$\widehat{\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}} \hat{D}_X \widehat{\begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}} < \widehat{\begin{bmatrix} I_{n_r+n_d} & 0 \\ 0 & 0 \end{bmatrix}} \hat{D}_X \widehat{\begin{bmatrix} I_{n_r+n_d} & 0 \\ 0 & 0 \end{bmatrix}},$$

*where $J$ is defined in (5.12) and $\hat{\cdot}$ is defined in (5.11).*

**Proof:** *Equivalence is proven by applying Theorem 5.4.1 and distributing the $\hat{\cdot}$ operator through the LMIs using Lemmas 5.4.1 and 5.4.2.* ■

Note that $\hat{X} = \text{diag}(X, X)$, $\hat{Y} = \text{diag}(Y, Y)$, and $\widehat{I(\gamma)} = \text{diag}(I(\gamma), I(\gamma))$. The constraints on $D_X$ (and implicitly on $D_Z$) force the blocks which align with the uncertainty set, $\mathbf{\Delta}$, and the disturbance, $d$, to satisfy maximum norm constraints (positive definite), as with the standard $\mu$ problem; the blocks which align with the noise, $\Delta_n$, are forced to satisfy minimum norm constraints (negative definite), see Definition 7.1.2.

## 7.2 Model validation in the frequency-domain

Now that we have reasonable and computable results for the constant matrix model validation problem, how do we use them? In general, model validation of an LTI system with norm-bounded structured uncertainty and experimental data can be performed by considering the model validation problem in the frequency-domain.

Because $\tilde{P}$ is dependent on $u$ and $y$, which are generally discrete-time experimental data, the application of Theorems 7.1.1 and 7.1.2 must be done on a point by point basis. That is, given discrete-time experimental data $u(kT)$ and $y(kT)$, transform them to the frequency-domain using the discrete Fourier transform (DFT), obtaining $u(\omega)$ and $y(\omega)$. Let $\mathbf{\Omega}$ be a subset of the frequency set resulting from the transformation of $u$ and $y$ into the frequency-domain. For each $\omega \in \mathbf{\Omega}$, compute

$\tilde{P}(\omega)$, using $u(\omega)$ and $y(\omega)$. Then define frequency-domain bounds in terms of the constant matrix lower and upper bounds of $\mu_g(\tilde{P}(\omega))$, denoted by $\gamma_{lb}(\omega)$ and $\gamma_{ub}(\omega)$, respectively, as follows:

$$\begin{aligned}\gamma^\star(\tilde{P}) &= \min_{\omega\in\boldsymbol{\Omega}}\{\mu_g(\tilde{P}(\omega))\}\\ \gamma^\star_{lb}(\tilde{P}) &= \min_{\omega\in\boldsymbol{\Omega}}\{\gamma_{lb}(\omega)\}\\ \gamma^\star_{ub}(\tilde{P}) &= \min_{\omega\in\boldsymbol{\Omega}}\{\gamma_{ub}(\omega)\},\end{aligned} \tag{7.15}$$

where the bounds satisfy the relationship

$$\gamma^\star_{lb}(\tilde{P}) \leq \gamma^\star(\tilde{P}) \leq \gamma^\star_{ub}(\tilde{P}). \tag{7.16}$$

We can now relate $\gamma^\star_{lb}$ and $\gamma^\star_{ub}$ to the model validation problem in the frequency-domain. Note that although $\gamma^\star_{lb}$ is a lower bound for $\gamma^\star$ it plays the roll of a sufficient condition in Theorem 7.2.1, and vice-versa for $\gamma^\star_{ub}$ in Theorem 7.2.2. This is because the sizes of noise, disturbances and uncertainty are related to $1/\gamma^\star_{lb}$ and $1/\gamma^\star_{ub}$, i.e., $\frac{1}{\gamma^\star_{ub}} \leq \|\cdot\| \leq \frac{1}{\gamma^\star_{lb}}$, where $\|\cdot\|$ represents the size of noise, disturbances, or uncertainty.

**Theorem 7.2.1** *Let $\gamma^\star_{lb}$ and $\boldsymbol{\Omega}$ be defined as in (7.15). If $\gamma \leq \gamma^\star_{lb}$ then the robust model and data are $\gamma$-consistent on the frequency set $\boldsymbol{\Omega}$.*

**Proof:**

$$\begin{aligned}\gamma \leq \gamma^\star_{lb} &\implies \gamma \leq \gamma_{lb}(\omega),\ \forall\omega\in\boldsymbol{\Omega}\\ &\implies \gamma \leq \mu_g(\omega),\ \forall\omega\in\boldsymbol{\Omega}.\end{aligned}$$

*Therefore, by application of Theorem 7.1.1 at each frequency $\omega \in \boldsymbol{\Omega}$, the robust model and data are $\gamma$-consistent on the frequency set $\boldsymbol{\Omega}$.* ∎

**Theorem 7.2.2** *Let $\gamma^\star_{ub}$ and $\boldsymbol{\Omega}$ be defined as in (7.15). If $\gamma > \gamma^\star_{ub}$ then the robust model and data are not $\gamma$-consistent on the frequency set $\boldsymbol{\Omega}$.*

**Proof:**

$$\begin{aligned}\gamma > \gamma^\star_{ub} &\implies \exists\omega_0\in\boldsymbol{\Omega}\ \textit{such that}\ \gamma > \gamma_{ub}(\omega_0)\\ &\implies \exists\omega_0\in\boldsymbol{\Omega}\ \textit{such that}\ \gamma > \mu_g(\omega_0).\end{aligned}$$

*Therefore, by applying Theorem 7.1.1 at the frequency $\omega_0 \in \boldsymbol{\Omega}$, the robust model and data are not $\gamma$-consistent on the frequency set $\boldsymbol{\Omega}$.* ∎

As an aside, note that in the synthesis of a robust controller, the $\mu$-analysis results for the same block structure as used in model validation should be smaller than $\gamma^\star_{lb}$. When this occurs, the controller will be robust to the disturbances, noises and uncertainties necessary for the model to be consistent with the measured data. Simply put, if $\mu(\omega) < \mu_g(\omega)\ \forall\,\omega\in\boldsymbol{\Omega}$ then the model is consistent with the data, and the controller is consistent with the model.

### 7.2.1 Frequency sweep algorithm

A frequency sweep algorithm for computing the frequency dependent upper and lower bounds for the model validation problem was developed using MATLAB. The syntax for a software implementation of this algorithm is

$$(\gamma_{lb}^{\star}, \gamma_{lb}(\omega), \gamma_{ub}(\omega), \Delta(\omega), n(\omega)) = \text{val}(P, W_n, blk, \\ u(\omega), y(\omega), \gamma_0, tol), \tag{7.17}$$

where $P$, $W_n$, $u$ and $y$ are defined as in Figure 7.1, $blk$ defines the structure of the uncertainty set $\mathbf{\Delta}$. Recall that at each frequency computation is reduced to a constant matrix problem where the lower bound is computed using a modified power algorithm and the upper bound is computed through $\gamma$ bisection on an LMI. $\gamma_0$ is the starting value of $\gamma$ and $tol$ is used to determine when to stop the $\gamma$ bisection. $P$ is a system representing the robust model and $u$ and $y$ are frequency varying vectors. At each frequency $\omega$, $\tilde{P}$ can be constructed from $P(\omega)$, $W_n(\omega)$, $u(\omega)$, and $y(\omega)$.

The algorithm returns the lower and upper bounds, $\gamma_{lb}(\omega)$ and $\gamma_{ub}(\omega)$, respectively, and the perturbation, $\Delta(\omega)$, and noise, $n(\omega)$, achieving the lower bound.

#### Lower bound frequency sweep

At each frequency point, $\omega$, the lower bound, $\gamma_{lb}(\omega)$, is computed using the modified power algorithm discussed in Section 7.1.2. Additionally, the perturbation, $\Delta(\omega)$, and noise, $n(\omega)$, achieving the lower bound are computed.

#### Upper bound frequency sweep

The upper bound, $\gamma_{ub}(\omega)$, is similarly computed at each frequency by looking for feasible solutions of the upper bound LMI. The feasibility tests are computed iteratively by bisecting on a value of $\gamma$. Therefore, in the frequency-domain the upper bound LMI computation results in bounds on $\gamma_{ub}(\omega)$. These bounds can be made arbitrarily close through bisection, for an appropriately chosen value of $tol$, and will in general be ignored when presented.

There are two types of computations which are relevant to the frequency-domain upper bound for $\mu_g$, $(i)$ the value of $\gamma_{ub}^{\star}$ and $(ii)$ the value of $\gamma_{ub}$ for each $\omega$ in $\mathbf{\Omega}$. It requires much less computation to compute only $\gamma_{ub}^{\star}$. The upper bound frequency sweep algorithm consists of two main parts: a $\gamma$ bisection and a frequency sweep. In the case of $(i)$, the $\gamma$ bisection is in the outer loop and the frequency sweep is in the inner loop, with the opposite for $(ii)$. $\gamma_0$ is used as a starting value for the $\gamma$-bisection.

In the case of computing only $\gamma_{ub}^{\star}$, for each $\omega$ two items are maintained: the largest $\gamma$, $\gamma_{min}(\omega)$, for which the LMI had no feasible solution and the smallest $\gamma$, $\gamma_{max}(\omega)$, for which the LMI had a feasible solution. If the first $x\%$, where $x$ is a parameter for the algorithm, of the frequency points tested have no feasible solution then the frequency sweep for this value of $\gamma$ is aborted and $\gamma$ is updated; similarly, if $x\%$ of the frequency points tested are feasible then the sweep is aborted and $\gamma$ is updated. This speeds up location of the correct $\gamma$-level to begin the frequency sweep, and minimizes the number of times the LMI must be checked for feasibility. For most problems this results in a threefold reduction in the number of LMI feasibility tests. If any of the frequency points were feasible then $\gamma$ is decreased, otherwise $\gamma$ is increased. After each adjustment of $\gamma$, the frequency points remaining to be tested are those for which $\gamma_{min}(\omega) < \gamma$. The iteration completes when $\min_\omega \gamma_{max}(\omega) - \min_\omega \gamma_{min}(\omega) < tol$.

In the case of computing $\gamma_{ub}(\omega)$, for each $\omega$ a $\gamma$ bisection on the LMI in Theorem 7.1.2 is performed. $\gamma_0$ is used as the initial value of the $\gamma$-bisection. For each value of $\gamma$, feasibility of the LMI is tested. If the LMI is feasible then $\gamma$ is reduced, otherwise $\gamma$ is increased. This is done iteratively until $\gamma_{k+1} - \gamma_k < tol$.

### 7.2.2 Frequency-domain considerations

Since model validation tests are performed in the frequency-domain, the finite length discrete-time data are transformed to the frequency-domain, with the DFT. Recall that an implicit assumption in the use of the DFT is that the time-domain data repeats periodically forever. Thus high quality frequency-domain data requires the use of time-domain data that looks as though it could repeat periodically.

In addition, drifts that result from nonlinearities may corrupt the data at other, more interesting, frequencies. This corruption is due to the frequency content in the step transition from the end of the data record to the beginning of the record as it repeats periodically. This effect can be mitigated by high pass filtering those signals which exhibit such drift.

As much as possible, experiments should be designed to excite all unmodelled dynamics. Note that because at each frequency the robust model and data are converted into a constant matrix problem, the uncertainty may be uncorrelated and complex at all frequencies.

## 7.3 Tuning the Robust Model

The previous sections presented a method for testing the consistency between experimental data and a robust model. This method can be further extended to iteratively refine both the nominal model and the uncertainty description so that the level of noise, disturbances, and uncertainty can be minimized without sacrificing the ability of the robust model to cover the experimental data.

Recall that Figure 7.1 corresponds to a general interconnection for a robust model which is suitable for model validation. Measurement noise corresponds to $n$, and there are additional exogenous disturbances, $d$, acting directly on the plant $P$. The model uncertainty is defined by $\mathbf{\Delta}$, which is a norm-bounded structured perturbation, consisting of possibly many different blocks.

Each signal and uncertainty block will generally have associated with it a stable, rational, minimum phase, and bi-proper weighting function which scales its size and frequency shape. $W_n$ corresponds to the weighting function for the measurement noise $n$ and the weights for $d$ and $\Delta \in \mathbf{\Delta}$ are included in the plant $P$. This section describes a methodology for tuning the weighting function by making use of the frequency varying upper and lower bounds for $\mu_g$. It is assumed that the robust model will be used for the synthesis of some type of robust controller using either $\mathcal{H}_\infty$ or $\mu$-synthesis. In this context, it is desirable to achieve $\mu(\omega) < \mu_g(\omega)$ or, more specifically, $\mu(\omega) < \gamma_{lb}(\omega) < \mu_g(\omega)$. Recall that with $\mathcal{H}_\infty$ or $\mu$-synthesis the optimal is often reached when $\mu(\omega)$ is flat across frequency. Therefore, it is likely that "flattening" $\mu_g(\omega)$ $(\gamma_{lb}(\omega))$ is a reasonable approach to take when refining the robust model.

For simplicity, throughout this section it is assumed that there are no disturbances, $d$, and a single block in $\Delta$. It is straightforward to extend the techniques to include disturbances and additional blocks in $\Delta$.

Let $P$ be the plant, with $W_n$ the measurement noise weight and $W_\Delta$ the uncertainty weight as in Figure 7.1. In most cases, the nominal model will be obtained from standard identification techniques and the structure of the uncertainty description arrived at through *a priori* knowledge of the system.

Assume open-loop experimental data are available which adequately captures the system dynamics in regions which need to be controlled. If these data are not already in the frequency-domain, it will be transformed. The frequency-domain data and the robust model will then be used with the model validation tools to iteratively "tune" the noise and uncertainty weights in the interconnection.

The initial value of the noise weight $W_n$ can be obtained by computing $\gamma_{lb}(\omega)$ (or $\gamma_{ub}(\omega)$ in those cases where $\mu_g = \gamma_{ub}^\star$) for the nominal model with $W_n = 1$. The frequency range where noise is most significant can be obtained by plotting the frequency response of the nominal model and comparing it with experimental data as follows. $W_n$ should be set equal to the inverse of $\min_{\mathbf{\Omega}_n} \gamma_{lb}(\omega)$, where $\mathbf{\Omega}_n$ is the frequency range where the noise is most significant. This provides an initial value for the magnitude of the noise weight, and a frequency shape for $W_n$ corresponding to the frequency "selection" set $\mathbf{\Omega}_n$.

Then, $\gamma_{lb}(\omega)$ $(\gamma_{ub}(\omega))$ is computed for the robust model, using the initial value of $W_n$ and $W_\Delta = 1$. The initial value for $W_\Delta$ should be set equal to the inverse of $\min_{\mathbf{\Omega}_\Delta} \gamma_{lb}(\omega)$, where $\mathbf{\Omega}_\Delta$ is the frequency range where the uncertainty is considered to be dominant. If there is no *a priori* knowledge of $\mathbf{\Omega}_\Delta$, then it can be taken to be the complement of $\mathbf{\Omega}_n$. This provides an initial value for the magnitude of the uncertainty weight, and a frequency shape defined by $\mathbf{\Omega}_\Delta$.

Once these initial values have been chosen, iterative computation of $\gamma_{lb}(\omega)$ (or $\gamma_{ub}(\omega)$) will provide information which can be used to adjust both the magnitude and frequency shape of $W_n$ and $W_\Delta$. The inverse of $\min_{\mathbf{\Omega}} \gamma_{lb}(\omega)$ can be used to adjust both the magnitude and frequency shape of the weights in the region of $\mathbf{\Omega}$, where $\mathbf{\Omega}$ is the frequency region corresponding to either dominant noise or dominant uncertainty. The choices in each iteration would be made to decrease the larger of noise and uncertainty by trading off between the magnitudes of the noise and uncertainty weights $W_n$ and $W_\Delta$, respectively. Tradeoffs should be made based on any *a priori* knowledge of the system or physically motivated intuition.

Because the lower bound involves solving for the signals in Figure 7.1, both $n$ and the perturbation $\Delta$ achieving the lower bound can be computed. By computing this information for each frequency, a perturbation, $\Delta(\omega)$, can be be constructed. For each block of $\Delta$ corresponding to a real parameter which is believed to be relatively static, the perturbation can be used, at those frequencies where the lower bound is a (local) minimum, to adjust the associated parameter in the nominal model so that the magnitude of the scaling of the parametric uncertainty is decreased. This can be done iteratively, until no additional improvement is obtained. In this way, there is a direct connection between identification of parameters in the nominal model and validation of a robust model.

When iterating between robust model refinement and controller synthesis, the stopping criterion is simply $\mu(\omega) < \gamma_{lb}(\omega) < \mu_g(\omega)$, $\forall \omega \in \mathbf{\Omega}$; otherwise the controller is not robust with respect to a noise, disturbance or uncertainty perturbation which is required to obtain consistency between the robust model and data.

## 7.4 Utility of the model validation method

The model validation lower bound shares most properties of the lower bound for the standard $\mu$ problem, with the exception of the implicit inner loop. The major difficulty is with the convergence of the inner loop when several real or complex repeated scalars are present. Referring to Table 7.2, it appears that the lower bound converges approximately 70% of the time when real repeated scalars are used, and approximately 90% of the time when only complex blocks are used. In general, the lower bound is extremely fast and comparable to the standard $\mu$ software.

The model validation upper bound is an LMI and inherits the limitations of the LMI solver employed. The LMI solver used in this work was developed by Gahinet *et al.* [26, 25]. This LMI solver has performance limitations with respect to the size of the problem. The size is determined by several quantities: the number of LMI constraints and the number of LMI decision variables. This is the first commercially available LMI solver, improvements can be expected.

## 7.5 Spring-mass example

To test the capabilities of these model validation methods, we consider the simple example of a spring-mass system with parametric uncertainty in the damping and stiffness coefficients. Data will be generated through simulation of a perturbed and noisy system. The upper and lower bounds are then used to "tune" the noise and uncertainty weights so that $\mu_g$ is minimized. The weights are then compared with the actual weights used in the mathematical model. Finally, the lower bound is used to make a robust estimation of the value of the perturbed parameters.

### 7.5.1 Description of the spring-mass model

A mathematical model of the spring-mass system is given by

$$m\ddot{x} + \beta\dot{x} + kx = u, \tag{7.18}$$

where $m$ is the mass, $\beta$ and $k$ are the damping and stiffness coefficients, respectively, $x$ is the spring displacement, and $u$ is the input force. This system can be realized in state-space form as

$$\begin{aligned} \dot{z}_1 &= z_2 \\ \dot{z}_2 &= \frac{1}{m}\left(-(k_0 + w_k\delta_k)z_1 - (\beta_0 + w_\beta\delta_\beta)z_2 + u\right) \\ y &= z + w_n n, \end{aligned} \tag{7.19}$$

where $z_1$ corresponds to the spring displacement $x$ from (7.18) and $z = [z_1^T, z_2^T]^T$. The parametric uncertainty in the stiffness and damping coefficients is defined by $k = k_0 + w_k\delta_k$ and $\beta = \beta_0 + w_\beta\delta_\beta$, respectively. Measurement noise is modelled by $w_n n$, where $w_n$ weights the noise appropriately. Refer to Figure 7.4 for a block diagram of the model validation interconnection, with $w_p = \text{diag}(w_k, w_\beta)$ and $\Delta = \text{diag}(\delta_k, \delta_\beta)$, where $\delta_k$ and $\delta_\beta$ are real parameters.

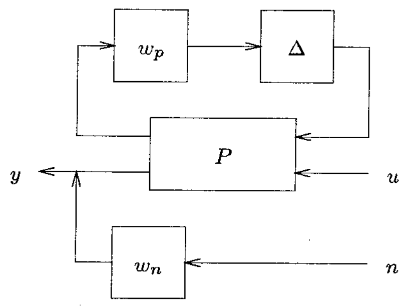

Figure 7.4: Spring-mass model validation interconnection.

Referring to (7.19), the parameters used for the nominal system are given by

$$\begin{aligned} m &= 1 \\ k_0 &= 1 \\ \beta_0 &= 1 \end{aligned} \tag{7.20}$$

and the perturbed parameters used in the simulation are

$$\begin{aligned} \delta_k &= 0.8 \\ \delta_\beta &= -0.3. \end{aligned} \tag{7.21}$$

The norm of the true uncertainty perturbation is $\|\Delta\|_\infty = \max(|\delta k|, |\delta\beta|) = 0.8$.

In this example, model validation is used to determine the validity of the robust model in Figure 7.4. To use model validation, the input/output data, $u$ and $y$, must be obtained. In this case, $y$ represents spring-mass data obtained through simulation of the perturbed and noisy system, (7.19). An exponentially decaying sine sweep was used as an input $u$, and initially $y$ was obtained through simulation of (7.19) with the perturbed parameter values, (7.21). It was found that using exponentially decaying sine sweeps as inputs generated much better frequency-domain data than when steps were used. After simulation, noise with a level of $\pm 0.1$ was added to the output in the time-domain, yielding the noisy $y$ used by the model validation software. The norm of the additive random noise is $\|n\| = \max_\omega \|n(\omega)\|_2 = 8.10$. The input and noise-free output data are shown in Figure 7.5. The input/output data was then

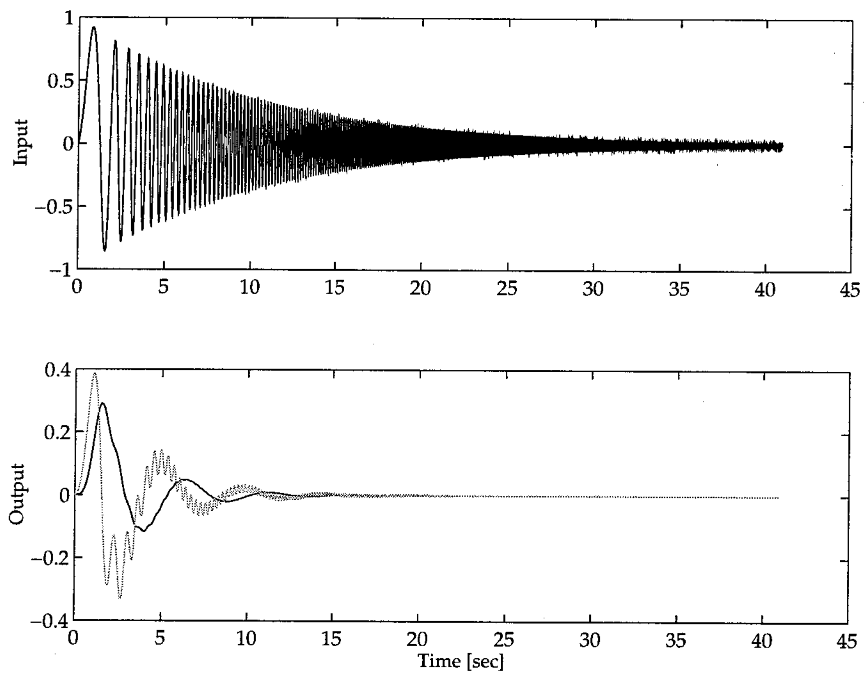

Figure 7.5: Spring-mass input and noise free simulated output. For the output, solid corresponds to spring displacement and shaded spring velocity.

transformed into the frequency-domain with an FFT. Figures 7.6 and 7.7 include plots of the nominal transfer functions (solid) and noisy and perturbed transfer functions (shaded, with data points indicated by "x").

### 7.5.2 Analysis of the actual spring-mass model

Since this is a simulation, we have access to the norm of the noise, $\|n\|$, and the sizes of the parametric uncertainty, $\delta_k$ and $\delta_\beta$ which were used in the simulation. These values can be compared with the model validation results to verify its applicability.

When we choose $w_n = \|n\|I_2$, $w_k = \text{abs}(\delta_k)$ and $w_\beta = \text{abs}(\delta_\beta)$ the bounds resulting from the model validation software are $1.003 \leq \mu_g \leq 1.010$. A graph of the

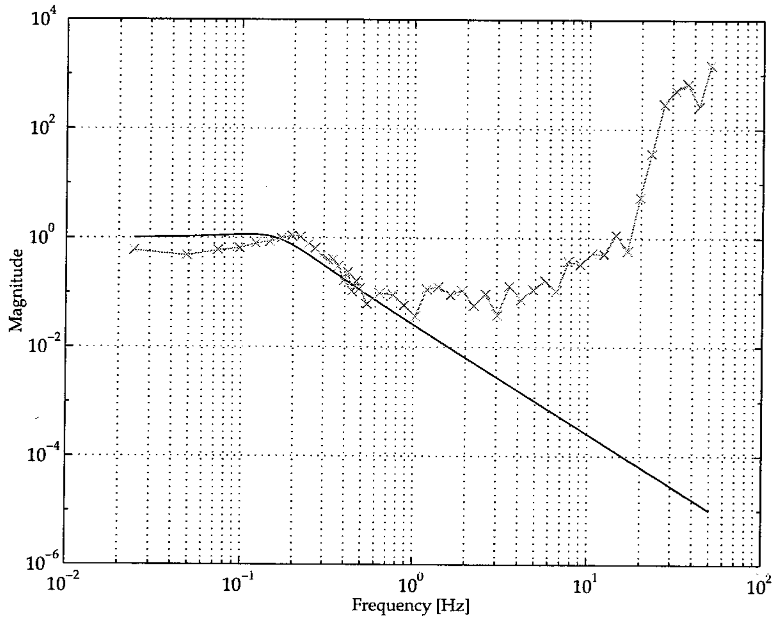

Figure 7.6: Spring-mass transfer function, $u \rightarrow y_1$: nominal (solid), noisy and perturbed (shaded). The noisy and perturbed data points used for model validation analysis are indicated by "x".

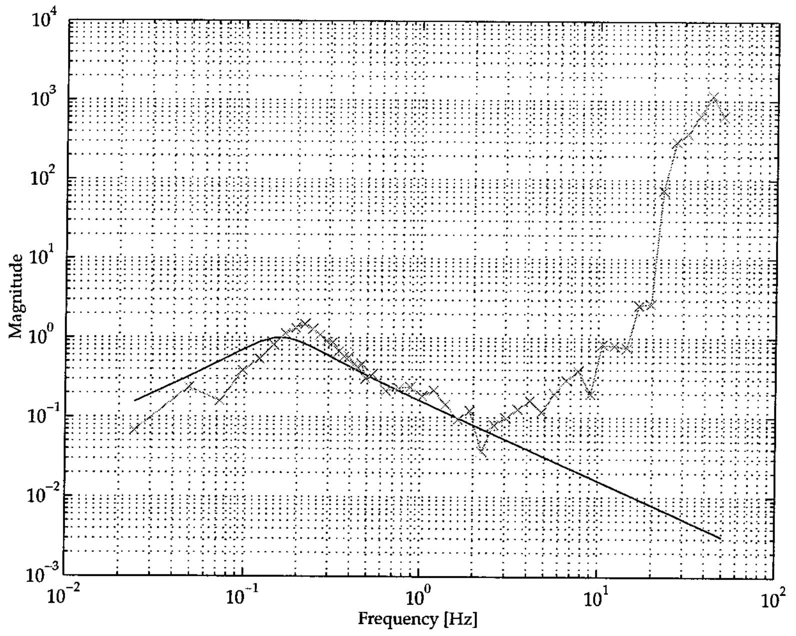

Figure 7.7: Spring-mass transfer function, $u \rightarrow y_2$: nominal (solid), noisy and perturbed (shaded). The noisy and perturbed data points used for model validation analysis are indicated by "x".

resulting lower and upper bounds is shown in Figure 7.8. Note that the lower bound did not converge for some data points. This is indicated on the graph by points for which there is an "o" but not an "x".

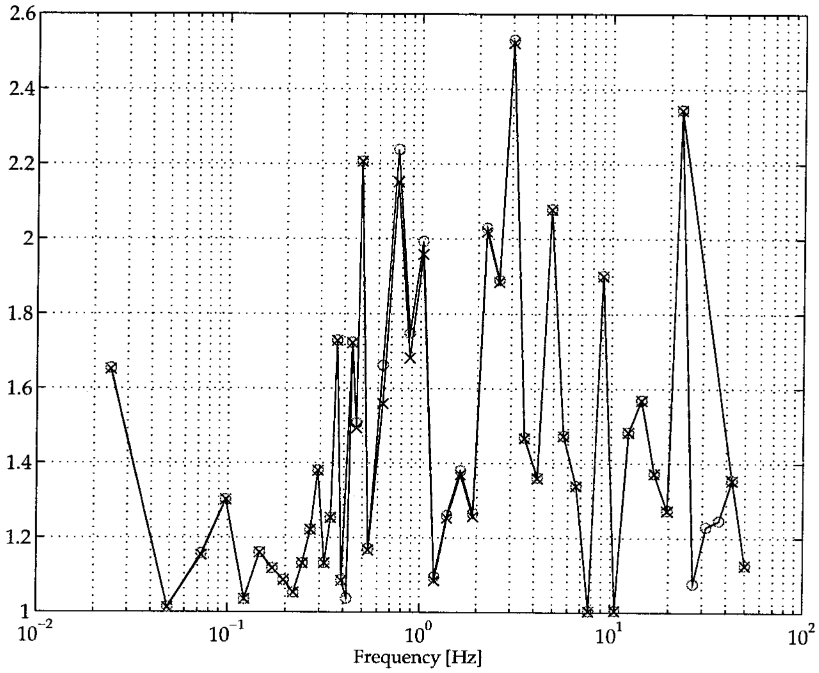

Figure 7.8: Model validation bounds of actual spring-mass model. At each frequency, the lower bound is indicated by "x" data points, the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

The scaled real parameter perturbation (containing the two parameters, $\delta_k$ and $\delta_\beta$) resulting from the lower bound is shown in Figure 7.9. Note that at low frequency, where the lower bound was close to the minimum, the values of the real parameters were nearly equal to the actual parameter values, with the exception of two data points for $\delta_\beta$. Also, for the two frequency points where the perturbation was not close to the actual value of $\delta_\beta$, the lower bound was not near its minimum. This indicates that when the robust model is appropriately weighted, the lower bound can be used to determine the correct value of static real parameters. The value of the static real parameters can be taken from the value of the corresponding perturbation, resulting from the lower bound, at those frequencies where the lower bound is close to its minimum value. These static values can then be wrapped back into the nominal model, with an appropriate reduction in the magnitude of the weight for the parameter.

In this particular problem, the lower bound was achieved at high frequency, resulting more from noise than parametric uncertainty; however, at low frequency, the lower bound was quite close to its minimum value, implying that the value of the perturbation can be meaningfully related to the static real parameters.

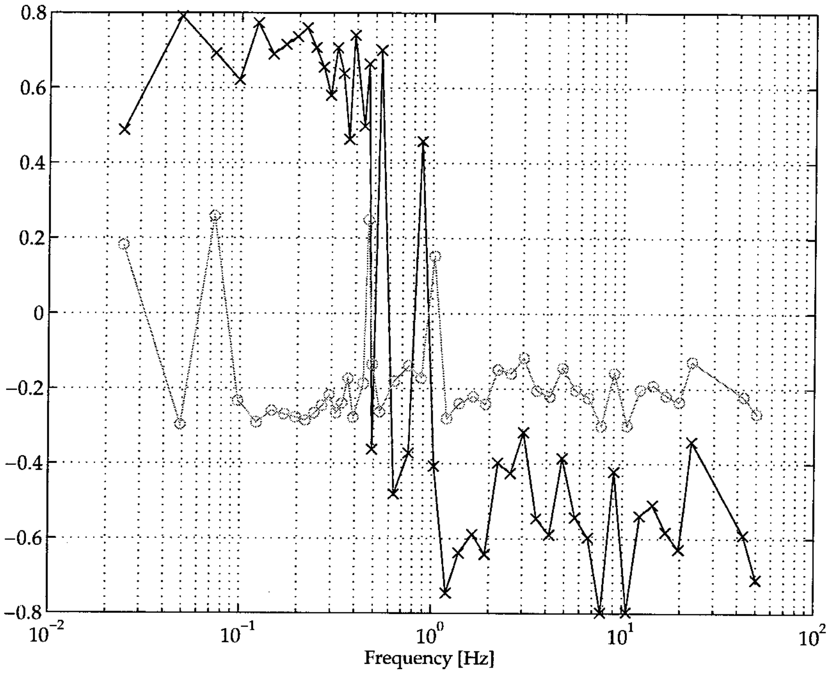

Figure 7.9: Model validation lower bound perturbation of actual spring-mass model. At each frequency, the $\delta_k$ and $\delta_\beta$ perturbations are indicated by "x" and "o" data points, respectively.

### 7.5.3 Development of a robust spring-mass model

Using the noisy and perturbed simulated data, the interconnection given in Figure 7.4, and frequency-domain model validation software, we can iteratively "tune" the frequency weights $w_p$ and $w_n$, without prior information on the noise and uncertainty. The objective is to end up with the largest $\gamma_{lb}^*$ possible. To this end, we will attempt to find a good starting magnitude for $w_p$ and $w_n$ using ideas presented in Section 7.3.

First, we consider model validation for a model without uncertainty, i.e., the nominal model where $\delta_k = \delta_\beta = 0$, and choose $w_n = 1$. Referring to Figures 7.6 and 7.7, there is noticeable mismatch between the nominal model and the noisy and perturbed data up to about 1.0 Hz, where it appears that noise takes over. So the bounds above 1.0 Hz will be considered when determining the value of the noise weight, $w_n$. A graph of the upper bound at each frequency is shown in Figure 7.10, where $\gamma_{ub}(\omega)$ is bounded by the curves in the figure. Data points for the upper bound are indicated by "o". In subsequent upper bound figures it is implicit that $\gamma_{ub}(\omega)$ lies between the curves containing "o" data points, although, in most cases they are overlapping. Referring to the frequency range above 1.0 Hz in Figure 7.10, $0.1187 < \min_{\Omega_n} \gamma_{ub} \leq 0.1238$, where $\Omega_n$ is the frequency range above 1.0 Hz. The average of this bound on $\min_{\Omega_n} \gamma_{ub}$ can be taken as an initial value for the peak magnitude of $1/w_n$, resulting in $w_n \approx 8.25$. This sets the size of allowable noise to about 8.25. Compare this with $\|n\| \approx 8.1$, the actual norm of the noise in the noisy and perturbed data.

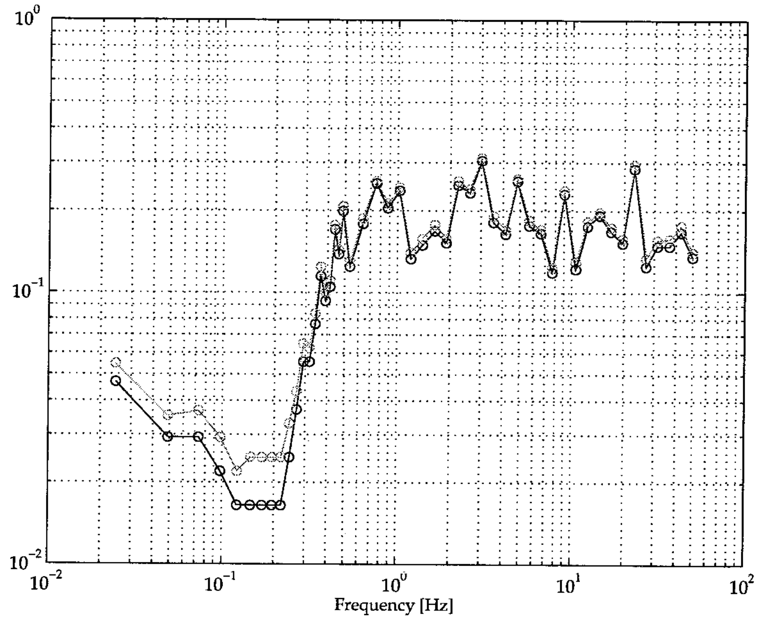

Figure 7.10: Model validation upper bound of nominal spring-mass model. At each frequency the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

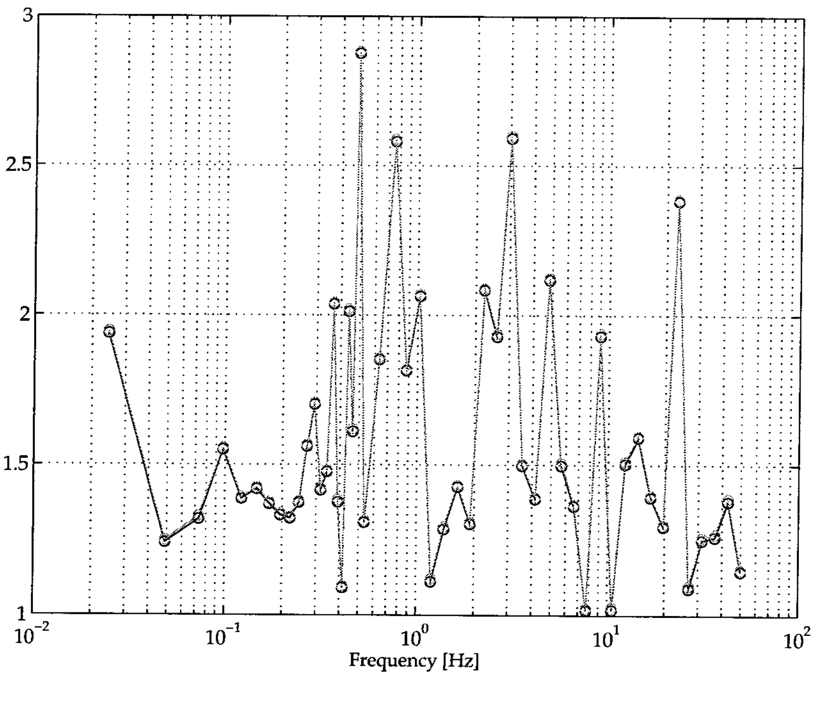

Figure 7.11: Model validation upper bound of unweighted spring-mass model. At each frequency the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

Second, using the computed noise level above, the uncertainty level can be computed. We define $w_p = I_2$ and $w_n$ as above. A graph of the resulting upper bound at each frequency is shown in Figure 7.11. As before, we assume that noise dominates above 1.0 Hz. Referring to Figures 7.6 and 7.7, the range below 0.4 Hz appears to be relatively noise free. This range will be used to determine the uncertainty level from Figure 7.11, and will be denoted by $\boldsymbol{\Omega}_{\boldsymbol{\Delta}}$. Referring to Figure 7.11, $1.2412 < \min_{\boldsymbol{\Omega}_{\boldsymbol{\Delta}}} \gamma_{ub} \leq 1.2488$. The average can be taken as an initial value for the peak magnitude of $1/w_p$. This results in the choice of $w_p \approx 0.80 I_2$. Note that the actual size of uncertainty is 0.8.

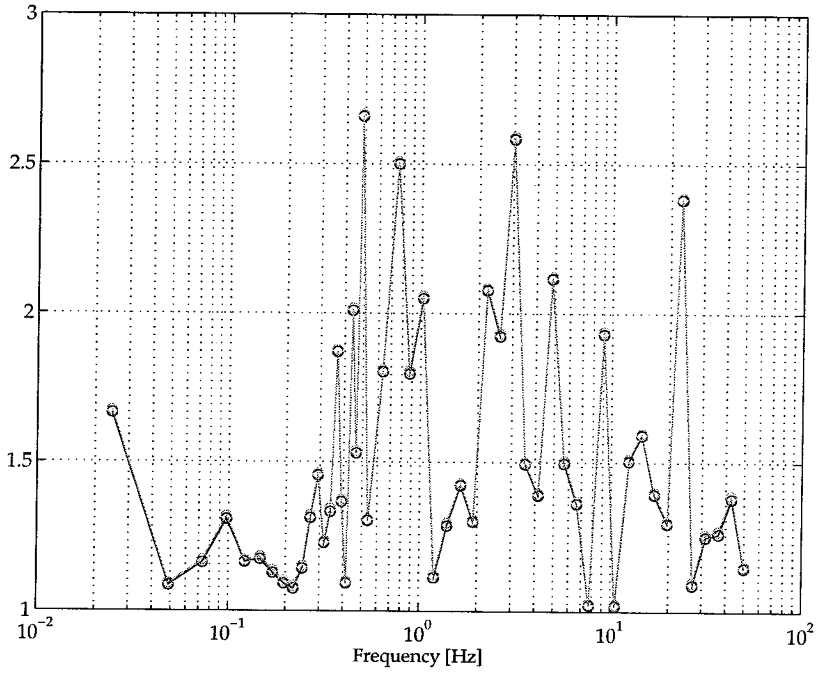

Figure 7.12: Model validation upper bound of robust spring-mass model. At each frequency the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

Finally, using the above values for the noise weight, $w_n$, and uncertainty weight, $w_p$, the bounds are computed at each frequency, and shown in Figure 7.12. Referring to Figure 7.12, $1.0180 < \gamma_{ub}^{\star} \leq 1.026$, which implies that the robust model corresponding to $w_n \approx 8.25 I_2$, $w_k = 0.80$ and $w_\beta = 0.80$ is consistent with the data. Because the upper bound is greater than 1, further iterative adjustments of $w_n$, $w_k$ and $w_\beta$ could be made. In particular, we could adjust the frequency shape of $w_n$ to take into account the observation that the noise enters above 1.0 Hz, and explore a tradeoff between $\delta_k$ and $\delta_\beta$, but that is not pursued here.

## 7.6 Ducted fan example

In this section we discuss an application of model validation to a ducted fan apparatus. We will see that model validation provides valuable information for the refinement of an identified model of the ducted fan. Based on this information, we propose a way to develop future uncertainty models for the ducted fan, and critique where the identified model should be improved. This work arose out of a collaborative effort with Bobby Bodenheimer and Matt Newlin.

### 7.6.1 Description of the ducted fan model

In this section, a brief review of the experimental ducted fan setup is given. A picture of the experimental system, a thrust-vectored ducted fan engine, is shown in Figure 7.13. It consists of a high-efficiency electric motor with a 6-inch diameter blade, capable of generating up to 9 Newtons of thrust. A detailed description of the performance of the fan was given by Choi *et al.* including models for the thrust as a function of flap angle and fan speed [14]. Overall, the experimental setup consists of

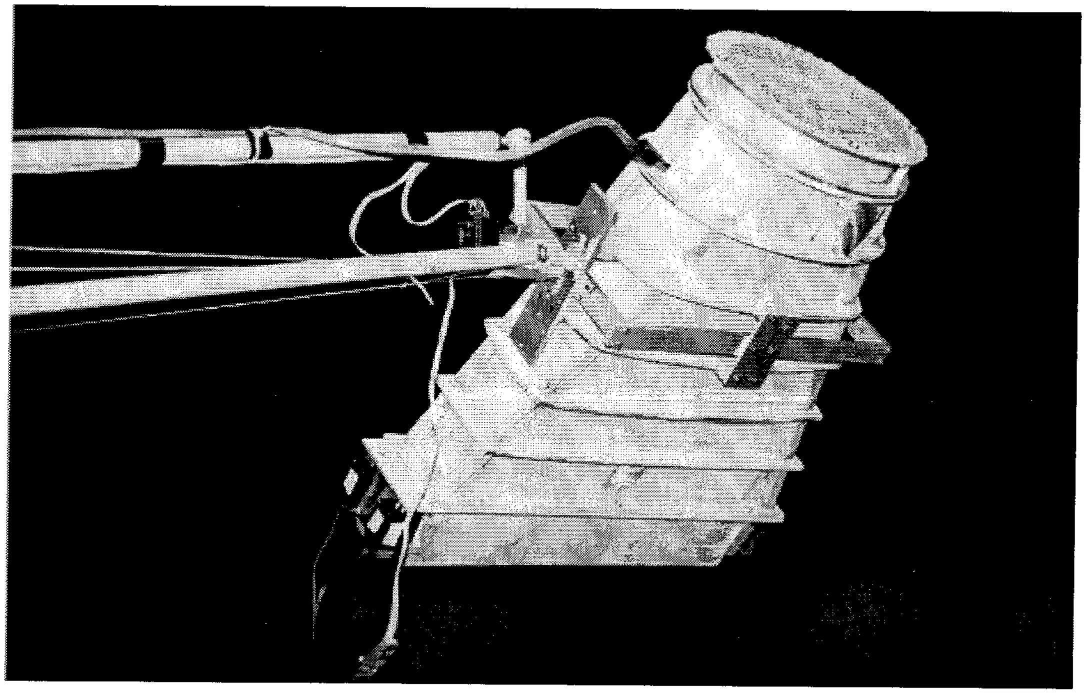

Figure 7.13: Ducted fan apparatus.

the ducted fan attached to a three DOF stand, as shown in Figure 7.14. The ducted fan is bolted to a rotating arm, which limits its motion to one rotational and two translational DOFs. The motor and propeller assembly are housed inside a wooden duct which has two flaps attached to its end. For these experiments, the ducted fan was configured in a stable pitch axis mode. Each axis is measured with optical shaft encoders.

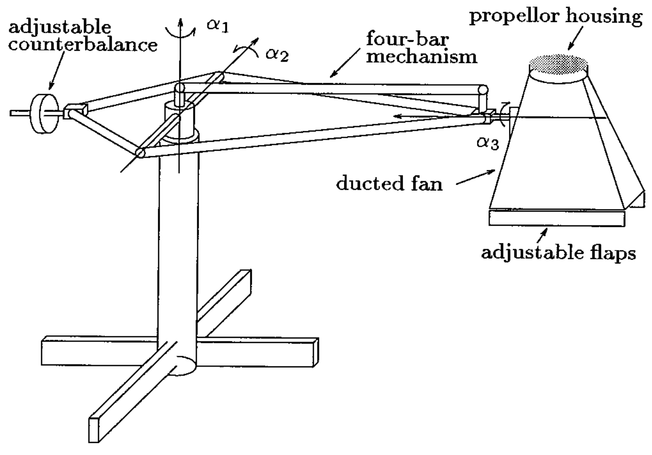

Figure 7.14: Ducted fan attached to stand.

This section concerns itself exclusively with a linear model for the ducted fan around hover. The linear model used in this section was developed using identification techniques and is presented in [9].

The uncertainty weight to be used for model validation is the weight $W_m$ depicted in Figure 7.15. Note that the uncertainty weight was developed in conjunction with a separate linearized model, and may not coincide with the weight a good controller designed from the identified model would have. To generate a data set to employ for model validation, a zero mean random signal was constructed around the equilibrium hover point caused by a force pair of $(2.65, 0)$N. The input data set is shown in Figure 7.16. The output of the ducted fan for the $\alpha_1$, $\alpha_2$, and $\alpha_3$ channels is shown in Figure 7.17.

The test input signals were designed to end with a quiescent period, so that the system might return to the initial rest state. This strategy works well for the states associated with $\alpha_2$, $\alpha_3$, and $\dot{\alpha}_1$, but not with $\alpha_1$, which exhibits a slow drift, caused by nonlinearities in the model.

As we aren't concerned with the system behavior at very low frequencies, this $\alpha_1$ drift is of little concern except that it might corrupt the data at other, more interesting, frequencies. This corruption is due to the frequency content in the step transition from the end of the data record to the beginning of the record as it repeats periodically.

To minimize this effect, the $\alpha_1$ data set is filtered with a fourth order acausal high pass filter with a cut-off frequency of 0.25 rad/s. The filter was chosen to cause minimal phase distortion over the frequency range of interest while making the processed time history appear suitable for the DFT. Note that the filtering corrupts the data below 0.25 rad/s.

Model validation acts on frequency-domain data, and hence representative plots of the transfer function data are now presented. The particular example we show is for the identified model. Shown are the transfer functions from $(u_1, u_2) \longrightarrow (\alpha_1, \alpha_2, \alpha_3)$.

In Figures 7.18–7.23, the solid line shows the magnitude of the FFT of the

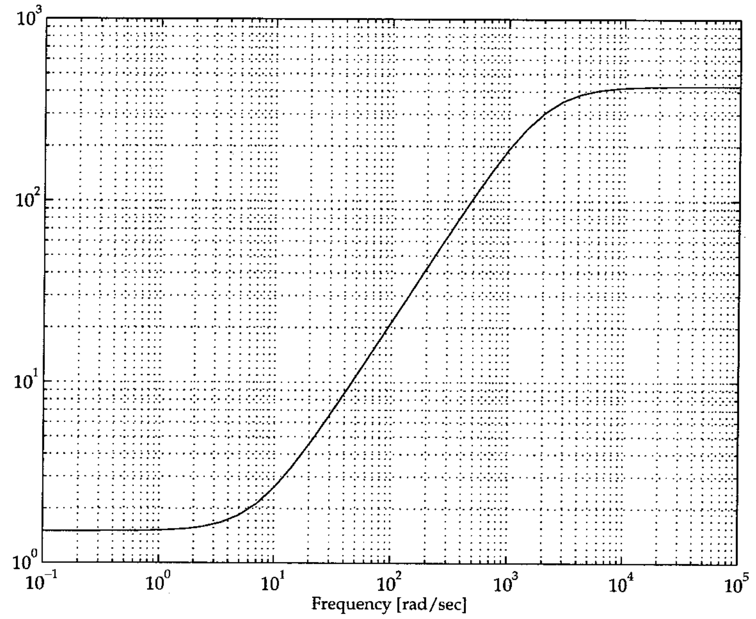

Figure 7.15: Magnitude of the ducted fan uncertainty weight, $W_m$, used for model validation.

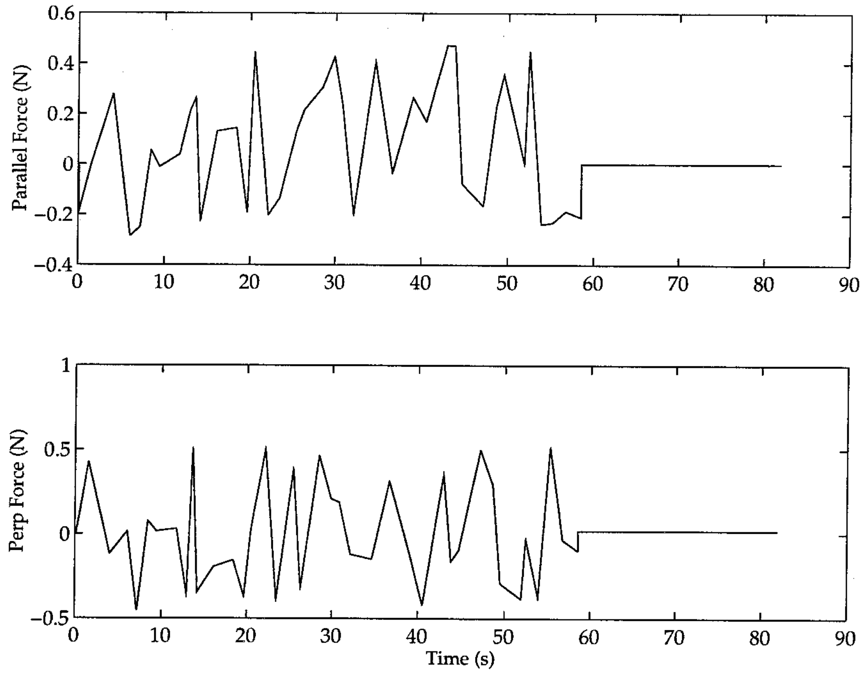

Figure 7.16: A set of random inputs to the ducted fan. Top: $u_1$. Bottom: $u_2$.

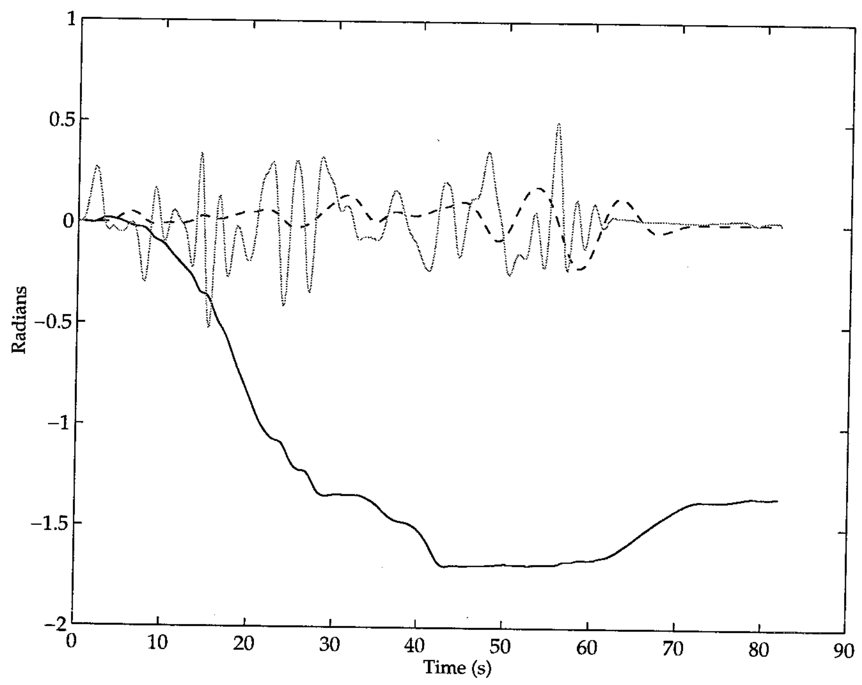

Figure 7.17: Outputs from the ducted fan. The solid line is $\alpha_1$, the dashed is $\alpha_2$ and the shaded is $\alpha_3$.

measured output data divided by the FFT of the input data. An 'x' in the figure shows a point at which the model validation bounds are calculated. The shaded line shows the magnitude of the FFT of the output data generated by simulating the identified model with the input data used to generate the real output data.

Note that these plots are not sufficient to predict the results of a model validation analysis, as phase information is important to the model validation calculations.

### 7.6.2 Analysis of the ducted fan model

In this section, the ducted fan model validation results are presented. A very important point to note when interpreting the results below is that we are *not* using the results to verify the robustness guarantees of a closed-loop system. This is the standard way of considering model validation, so readers familiar with the subject should be wary of falling into this habit of thinking. Instead, we are using model validation to determine if a model and uncertainty description can capture the dynamic structure of the true system.

Because $\mathcal{H}_\infty$-synthesis, when posed as an output tracking problem, makes no distinction between a command input and a noise input, we have limited empirical intuition for the noise weight required by the validation procedure. Preliminary model validation data was used to iterate on the magnitude of $W_n$; the final choice for $W_n$ was $W_n = 0.05I$. $W_m$ was unchanged. These preliminary results seemed to indicate that the problem is not particularly sensitive to the choice of $W_n$, so it was set to a

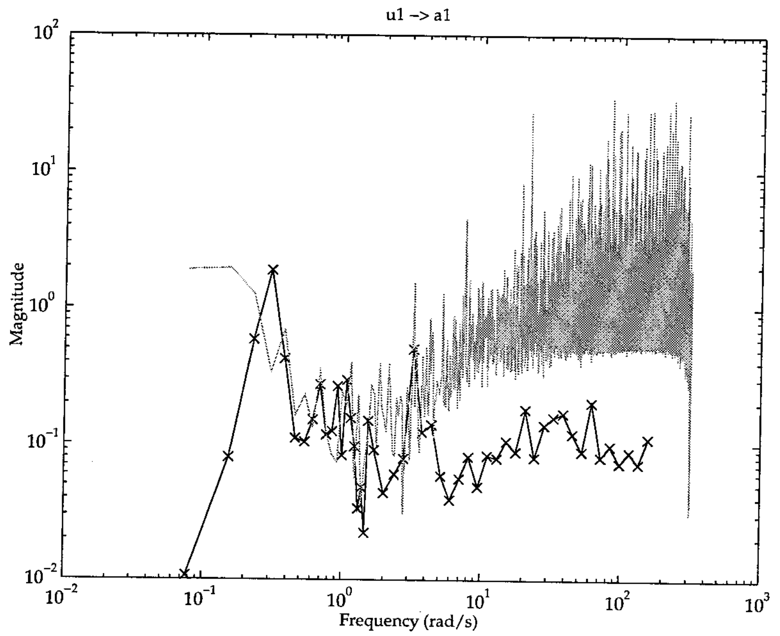

Figure 7.18: Ducted fan transfer function, $u_1 \rightarrow \alpha_1$, from measured data (solid with an 'x' at the actual data point) and simulated with the ducted fan model (shaded).

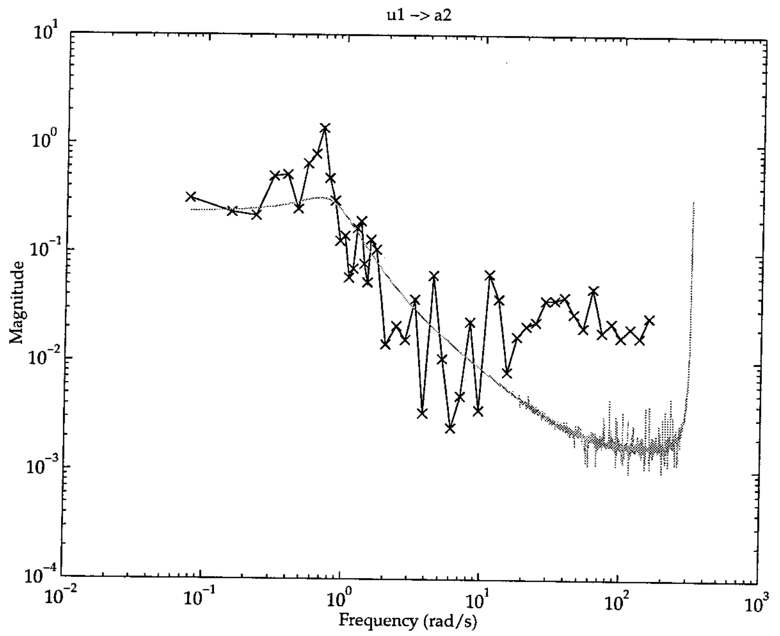

Figure 7.19: Ducted fan transfer function, $u_1 \rightarrow \alpha_2$, from measured data (solid with an 'x' at the actual data point) and simulated with the ducted fan model (shaded).

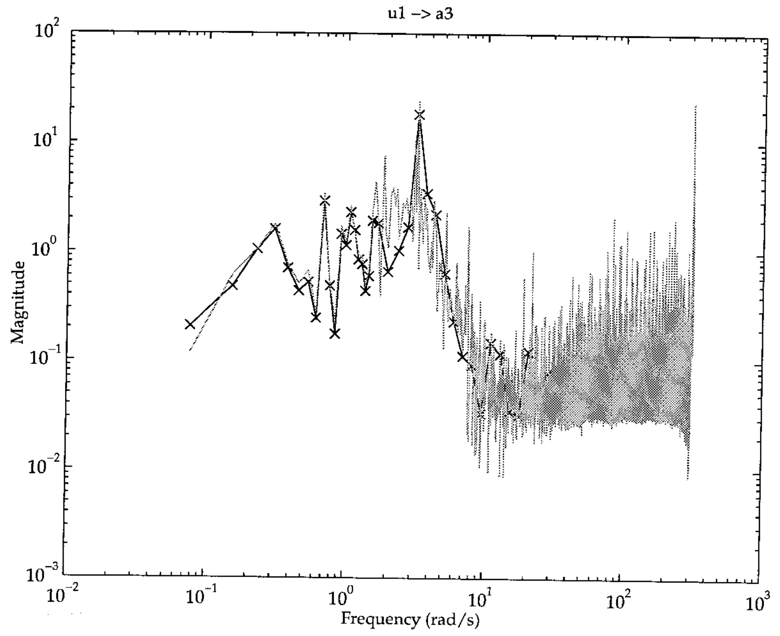

Figure 7.20: Ducted fan transfer function, $u_1 \rightarrow \alpha_3$, from measured data (solid with an 'x' at the actual data point) and simulated with the ducted fan model (shaded).

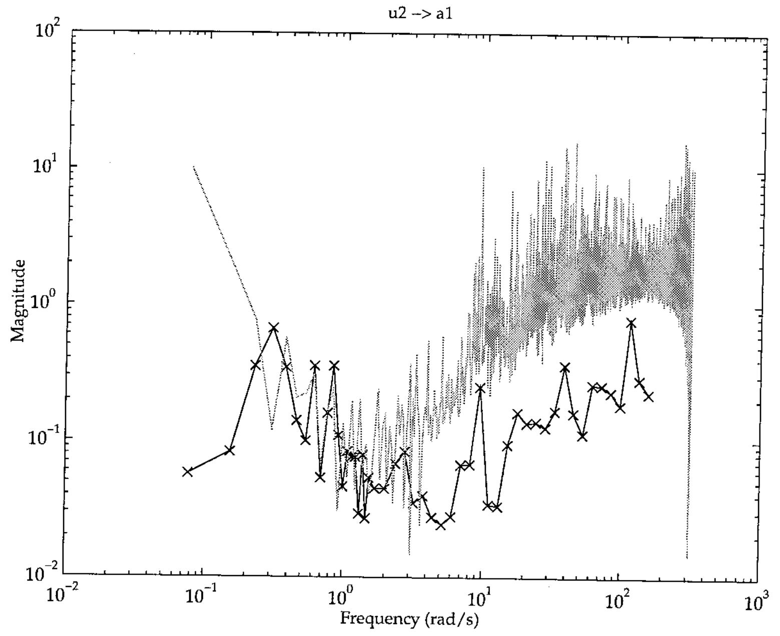

Figure 7.21: Ducted fan transfer function, $u_2 \rightarrow \alpha_1$, from measured data (solid with an 'x' at the actual data point) and simulated with the ducted fan model (shaded).

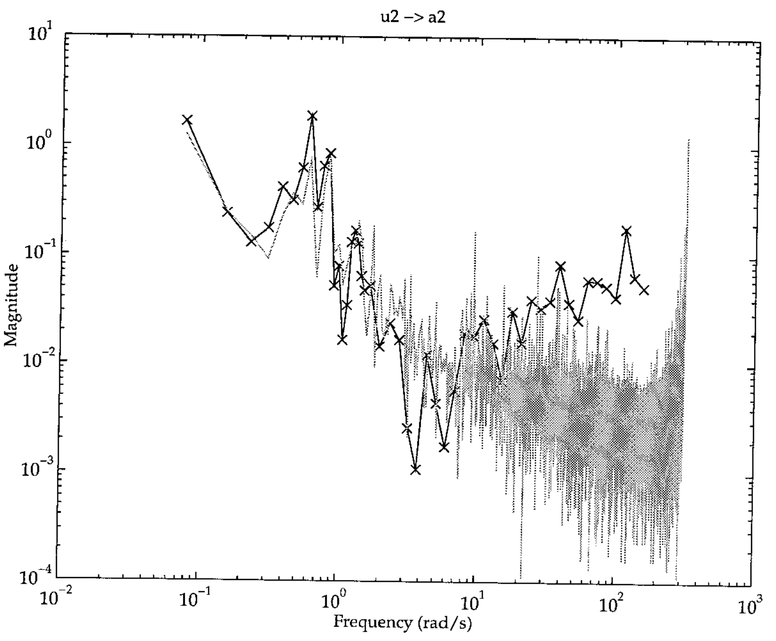

Figure 7.22: Ducted fan transfer function, $u_2 \to \alpha_2$, from measured data (solid with an 'x' at the actual data point) and simulated with the ducted fan model (shaded).

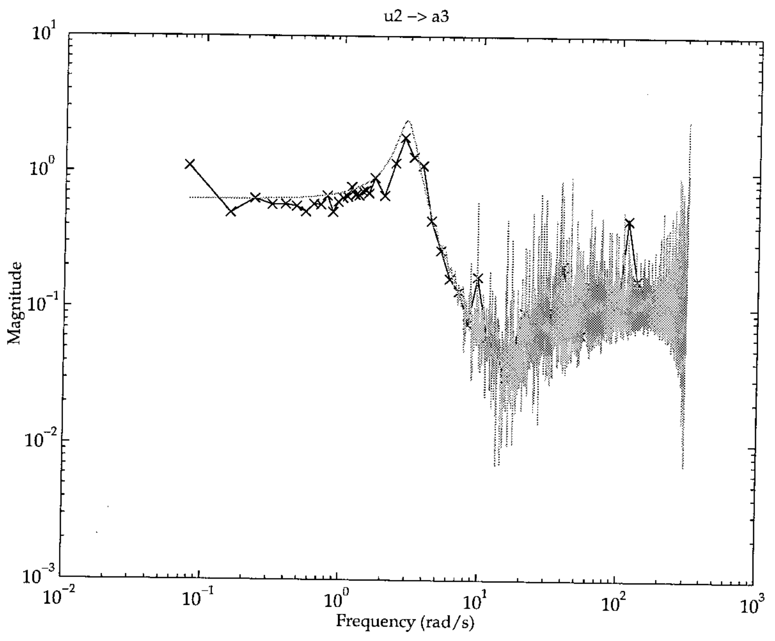

Figure 7.23: Ducted fan transfer function, $u_2 \to \alpha_3$, from measured data (solid with an 'x' at the actual data point) and simulated with the ducted fan model (shaded).

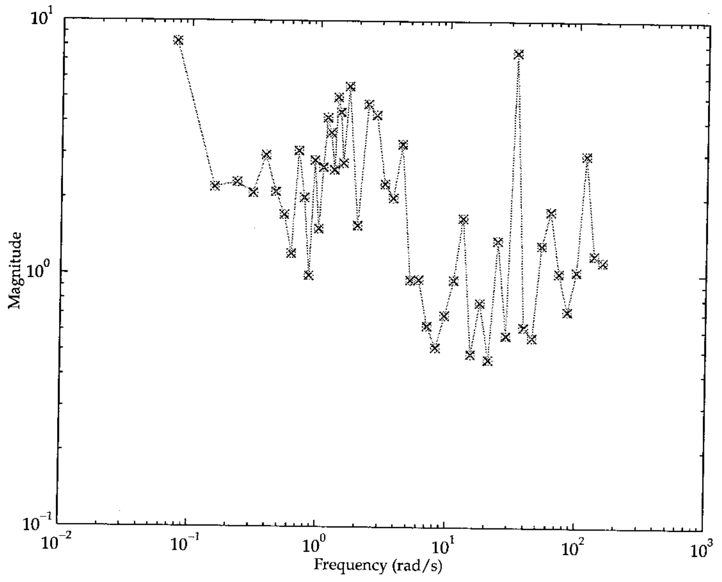

Figure 7.24: Model validation bounds for the ducted fan: $(u_1, u_2) \rightarrow \alpha_1$. At each frequency, the lower bound is indicated by "x" data points, the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

low value.

Figure 7.24 shows that the MISO transfer function, $(u_1, u_2) \longrightarrow \alpha_1$, is consistent for $\gamma \approx 0.5$. It is much better over most of the frequency range but drops around 10 rad/s. This is interesting, as it is after the mode shown in Figure 7.15. Figure 7.25 also shows that the MIMO transfer function $(u_1, u_2) \longrightarrow (\alpha_2, \alpha_3)$ is consistent for $\gamma \approx 1$.

The previous two results might lead us to believe that the complete MIMO model is $\gamma$-consistent for $\gamma \approx 0.5$. Figure 7.26 shows the bounds for the MIMO transfer function $(u_1, u_2) \longrightarrow (\alpha_1, \alpha_2, \alpha_3)$, and demonstrates clearly that this is not the case. In fact, the model and data only become $\gamma$-consistent at high frequency, where the uncertainty weight becomes very large. This implies that either there are unmodelled dynamics in the ducted fan that have a significant effect on the $\alpha_1$ channel, or that the specific uncertainty required to obtain model and data consistency is different for the $\alpha_1$ channel than for the $(\alpha_2, \alpha_3)$ channels. We address this question further below.

Additional model validation analyses on sets of different channels are shown in Figures 7.27–7.30.

The model validation results are tabulated in Table 7.3. The "transfer function" column shows the output channels which were selected for particular validation run; the inputs were always $(u_1, u_2)$. The second column shows the lower and upper bounds, $\gamma_{lb}^{\star}$ and $\gamma_{ub}^{\star}$, for the identified ducted fan model. For the single channel case,

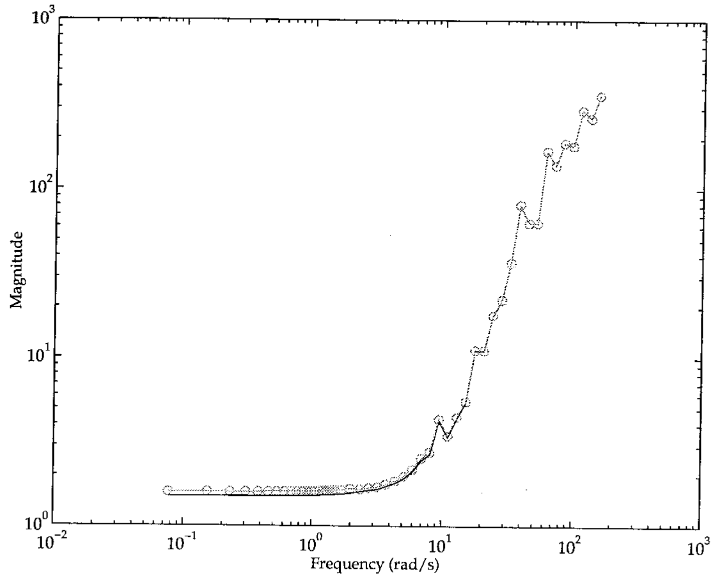

Figure 7.25: Model validation upper bound for the ducted fan: $(u_1, u_2) \to (\alpha_2, \alpha_3)$. At each frequency, the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

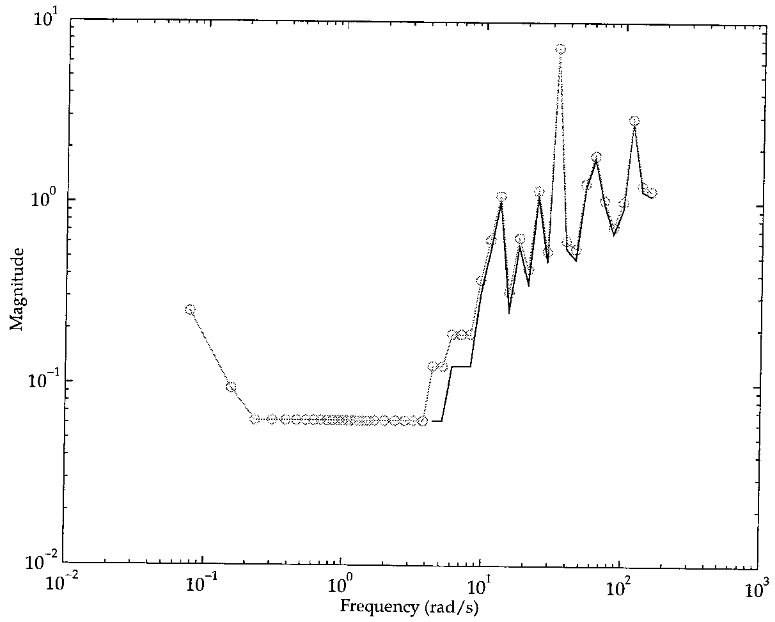

Figure 7.26: Model validation upper bound for the ducted fan: $(u_1, u_2) \to (\alpha_1, \alpha_2, \alpha_3)$. At each frequency, the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

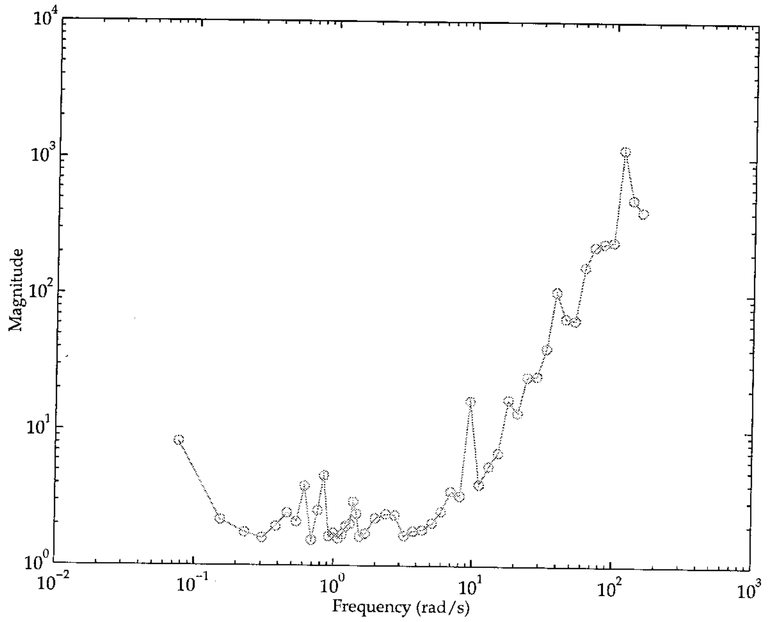

Figure 7.27: Model validation upper bound for the ducted fan: $(u_1, u_2) \rightarrow \dot{\alpha}_1$. At each frequency, the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

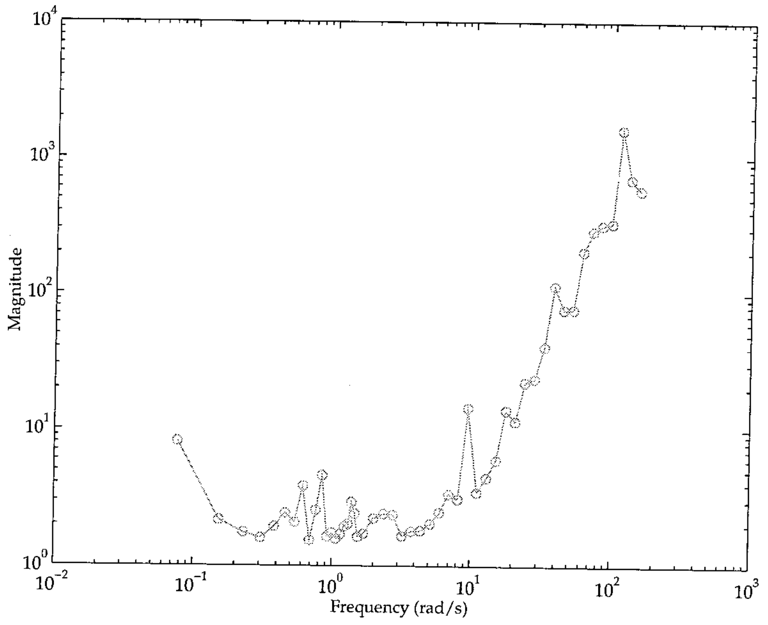

Figure 7.28: Model validation upper bound for the ducted fan: $(u_1, u_2) \rightarrow \alpha_3$. At each frequency, the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

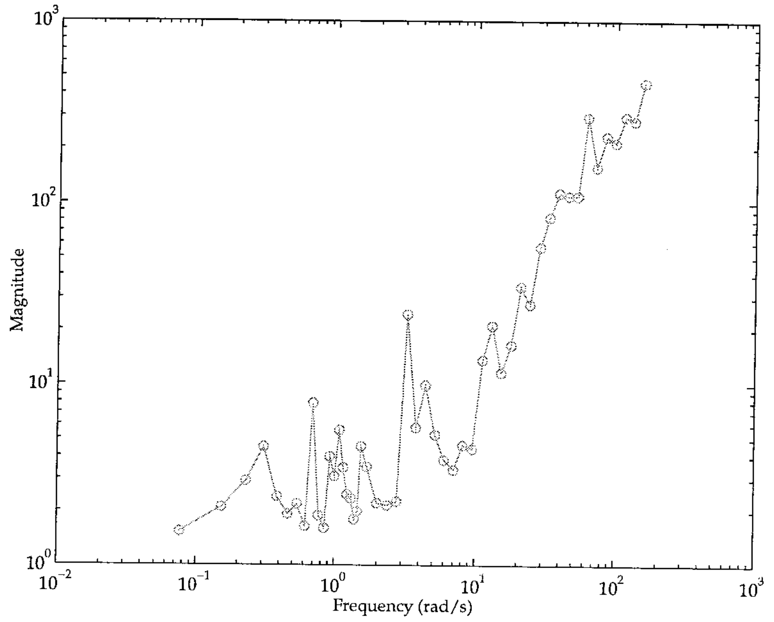

Figure 7.29: Model validation bounds for the ducted fan: $(u_1, u_2) \rightarrow \alpha_2$. At each frequency, the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

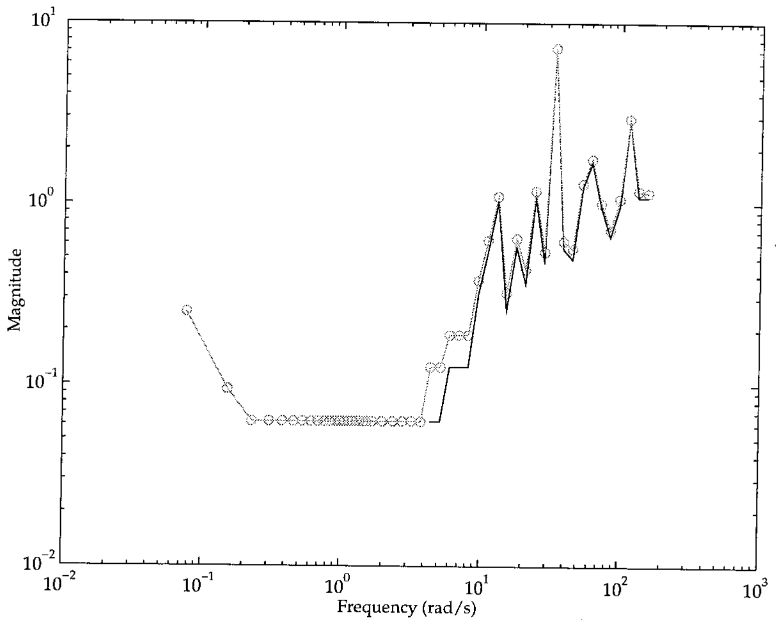

Figure 7.30: Model validation bounds for the ducted fan: $(u_1, u_2) \rightarrow (\alpha_1, \alpha_3)$. At each frequency, the the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection.

| Transfer Function | Identified Model | | Nonlinear Simulation | |
|---|---|---|---|---|
| | $\gamma^\star_{lb}$ | $\gamma^\star_{ub}$ | $\gamma^\star_{lb}$ | $\gamma^\star_{ub}$ |
| $\alpha_1$ | 0.5139 | 0.5139 | 0.2241 | 0.2241 |
| $(\alpha_2, \alpha_3)$ | 1.0520 | 1.1177 | 0.4022 | 0.4525 |
| $(\alpha_1, \alpha_2, \alpha_3)$ | 0 | 0.0625 | 0 | 0.0625 |
| $\dot{\alpha}_1$ | 0.1195 | 0.1195 | 0.2352 | 0.2352 |
| $\alpha_3$ | 2.0466 | 2.0466 | 0.7401 | 0.7401 |
| $\alpha_2$ | 1.0583 | 1.0583 | 0.4025 | 0.4025 |
| $(\alpha_1, \alpha_3)$ | 0 | 0.0625 | 0 | 0.0625 |

Table 7.3: Ducted fan model validation results. Where the upper and lower bounds are theoretically equal, the value of $\mu_g$ is given.

$\mu_g$ can be transformed into a standard $\mu$ problem, and the bounds for the transformed problem are given. Finally, the third column shows model validation results where the data $y$ was not obtained from an actual experiment but from applying the input $u$, shown in Figure 7.16, to a nonlinear simulation. Since the nonlinear simulation is undamped, the resulting output was then windowed in the time-domain before being transformed into the frequency-domain. Examining them shows that the results are very similar to those for the identified model with real data, which allows us to conclude that the discrepancy between the model validation results for $(\alpha_2, \alpha_3)$ and $(\alpha_1, \alpha_2, \alpha_3)$ are not due to unmodelled dynamics (as there are none in the nonlinear simulation), but that it must be due to the uncertainty structure not being representative.

From the results shown, it is clear that the present uncertainty description, that of a lumped multiplicative uncertainty at the input, is insufficient to account for the data in all three channels simultaneously. The problem channel appears to be $\alpha_1$, since a model validation on $\alpha_2$ and $\alpha_3$ shows those channels work well together, but neither works well with $\alpha_1$. Note, though, that a model validation on $\alpha_1$ alone shows that the model is good. This is a striking example, in a practical application, of why thinking about MIMO systems in a "loop-at-a-time" framework is incorrect. To improve the model, a different uncertainty description is needed to account for this difference.

The model validation results indicate that the $\dot{\alpha}_1$ channel needs to be improved. This probably indicates that better rate filters are required, but it might be poorly predicted because the effects of friction and stiction are particularly significant here.

The model validation has proven to be a capable tool for indicating improvements in models of the ducted fan which would have been difficult to ascertain otherwise. In particular, better models should incorporate either a different type of uncertainty or more structure in the uncertainty. Additionally, some modelling of the gyroscopic effect in the fan blades is clearly needed.

## 7.7 Summary

This work was motivated by the need to unify the previously disparate methods used for system identification and the construction of robust models. Previously there was no way to verify whether a robust model used for the synthesis of robust controllers covered the experimental data, or even if the uncertainty description was meaningful. With the development of the tools discussed in this work there is now a direct and computable connection between experimental data and robust models. This relationship is reduced to the simple question: "Is $\mu(\omega) < \mu_g(\omega)$, $\forall\omega \in \mathbf{\Omega}$?" If the answer is yes then the controller is robust over the model set and the model set is guaranteed to cover all previously observed experimental data. Although model validation provides a powerful analytical tool, it is an unfortunate misconception that model validation can "validate" a robust model. In fact, because model validation is data specific, it can only invalidate a model, or at most, verify that a model set covers all observed data.

A methodology was presented which makes use of lower and upper bounds for $\mu_g$ to iteratively tune the magnitude and frequency shape of the weights in the robust model. Additionally, it was shown how the lower bound could be used to iteratively adjust parameters in the nominal model. This aspect of the lower bound is an important contribution to the application of robust control theory to practical problems. This methodology was demonstrated with an example consisting of simulated data from a spring-mass system. An example of the use of model validation to analyze a robust model for an experimental ducted fan system was also presented. The development of a lower bound for $\mu_g$ with better convergence properties will be the focus of future efforts.

For linear systems, the frequency-domain provides a great simplification of dynamical system representation, as convolution in the time-domain becomes multiplication in the frequency-domain. Thus, data which is coupled across time is decoupled in frequency, and the associated robustness analysis and model validation problems are greatly simplified by this decoupling. While the frequency-domain is natural for continuous-time infinite horizon data, it is unnatural for discrete-time finite horizon data, where an implicit assumption in the transformation is that the time-domain data repeats periodically forever. Frequently, through careful experiment design, one may collect data that appear fairly consistent with this periodicity assumption, and the errors induced by going to the frequency-domain are reasonably small. Often, however, it is not possible to collect data that seem suitable for transformation. In these cases, it would be a great benefit to perform the model validation computation in the time-domain. Although this is no harder conceptually, the coupling of the problems from one time to the next makes the computation much more expensive. Effective computation of such problems is an area of current research.

# Chapter 8

# Robust Hover Model and Controller

Traditionally, modelling and controller analysis and synthesis have been treated as two separate and unrelated problems. As demonstrated in Chapter 7, the model validation methodology can be used effectively to both analyze and iteratively design better robust models, by forming a bridge between modelling and controller analysis and synthesis.

System identification tools, such as the prediction error method, have gained widespread acceptance, primarily because they provide an easy way to estimate parameters in a nominal model [44, 64]. Nonetheless, they do not provide the control engineer with a systematic framework for constructing a robust model. Although many systems are easily controlled without resorting to robust control techniques, e.g., a simple inverted pendulum, it is true that robust control techniques offer a powerful framework for studying complex systems such as helicopters.

Unfortunately, while techniques in robust control have been successfully applied to many complex systems, such as the space shuttle [20], they remain at present a very *ad hoc* procedure. This rests on the fact that until recently there have been no computable measures which could directly relate a robust model to the physical system it represents. The control engineer was forced to rely on intuition to develop a suitable robust model.

In this chapter a new iterative design process is developed using model validation analysis, $\mu$-analysis and $\mu$-synthesis, simulation, and implementation. This design process iteratively links the previously disparate areas of system identification, robust model construction, and $\mu$-synthesis through the use of recent results in and new software for model validation of general linear uncertain systems. Using this design process, uncertainty and noise descriptions can be iteratively adjusted so that a controller designed with $\mu$-synthesis satisfies $\mu(\omega) < \mu_g(\omega)\,\forall\,\omega$: a robustness criterion which takes into account a robust model and experimental data. This design process is important: for the first time the control engineer can place more confidence in and rely upon simulation, because model validation guarantees that the robust model is directly related to and able to generate all observed data collected from the physical system. Furthermore, the controller is guaranteed to be robust with respect to this model. In other words, the principal design work can be more reliably based on simulation. The result is that fewer implementations of the control system are necessary during the design cycle, limiting the chance of catastrophic failure.

This design process is presented in Section 8.1 and used to iteratively design a robust model and robust controller for the helicopter. Details on the construction of the new robust model are contained in Section 8.1. In Section 8.2 the model validation software is utilized to analyze and compare the helicopter hover model presented in Chapter 6 with the new robust model. An evaluation of the new robust hover controller is presented in Section 8.3.

## 8.1 Development of a robust hover model

In this section a new iterative design process is presented which addresses the construction of a robust model and the synthesis of a robust controller while taking into account all observed experimental data. The bridge which forms the link between robustness and data is model validation. Heavy use of the concepts developed in Chapter 7 will be made in this chapter. The design process is presented through its application to the design of a robust model and robust controller for the helicopter.

The interconnection used for model validation of the helicopter operating near hover is shown in Figure 8.1. $P$ is obtained from the nominal discrete-time helicopter hover model, $\hat{G}_3$, developed in Chapter 4, through a norm-preserving bilinear transformation, $u$ is the control input, $y$ is the measurement, $n$ is the measurement noise, $W_n$ is the weight on the measurement noise, $W_m$ is the actuator uncertainty weight, and $\Delta$ is the lumped multiplicative input uncertainty. As discussed in Chapter 6, the nominal model was augmented with multiplicative input uncertainty to account for unmodelled dynamics.

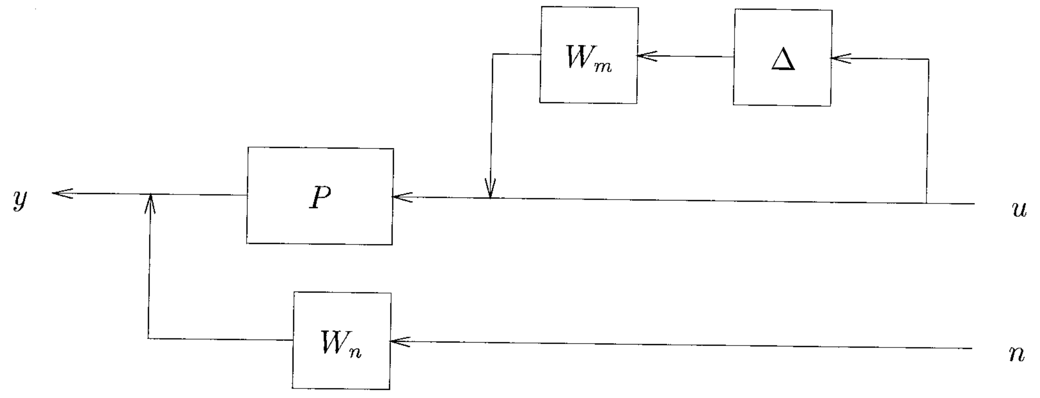

Figure 8.1: Helicopter model validation interconnection: nominal model $P$, uncertainty $\Delta$, uncertainty weight $W_m$, control input $u$, measurement $y$, measurement noise $n$, and measurement noise weight $W_n$.

In order to fit the model validation framework, the disturbance, *dist*, (Figure 6.2) is interpreted as playing the role of measurement noise, $n$, in Figure 8.1. Recall that when $\mathcal{H}_\infty$ and $\mu$-synthesis are posed as output tracking problems, there is no distinction made between output commands and noise on the plant output. In the synthesis interconnection in Figure 6.2, the restriction on the output command, *dist*, is simply $\|dist\|_2 \leq 1$. The model validation interconnection in Figure 8.1 is compatible with the synthesis interconnection in Figure 6.2 so long as for each frequency $\omega$ $\|W_n(\omega)n(\omega)\|_2 \leq \frac{1}{\mu}$. Therefore, for model validation, $W_n$ can be freely chosen subject to the restriction that $\max_\omega \|W_n(\omega)n(\omega)\|_2 \leq \frac{1}{\mu}$. This is easily verified, as $n$ is directly computable using the model validation lower bound.

Transfer functions of the nominal model of the helicopter operating near hover (dashed), developed in Chapter 4, and constructed directly from the experimental data (solid) are shown in Figures 8.2–8.10. The "x" indicates which points were used in the model validation analysis.

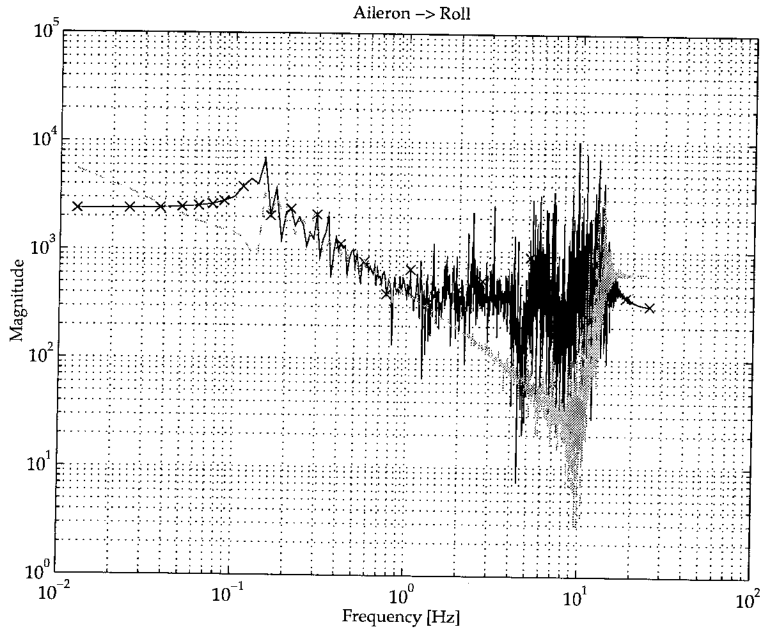

Figure 8.2: Helicopter hover transfer function: aileron $\rightarrow \phi$. Experimental (solid), nominal simulation (dashed).

Recall that in Chapter 6, the uncertainty weight $W_m$ (Figure 6.2) was tweaked iteratively, based on the behavior of the closed-loop system: first through simulation and then, for those controllers which simulated well, through implementation. The properties of interest were disturbance rejection and output tracking. At each step in the design cycle, the weights were adjusted and a new controller was synthesized and evaluated. This process continued for many iterations before satisfactory closed-loop performance and robustness were achieved in implementation. The iterative tweaking of the uncertainty and performance weights and the subsequent synthesis and evaluation of the LQG and robust $\mathcal{H}_\infty$ controllers presented in Chapter 6 was an

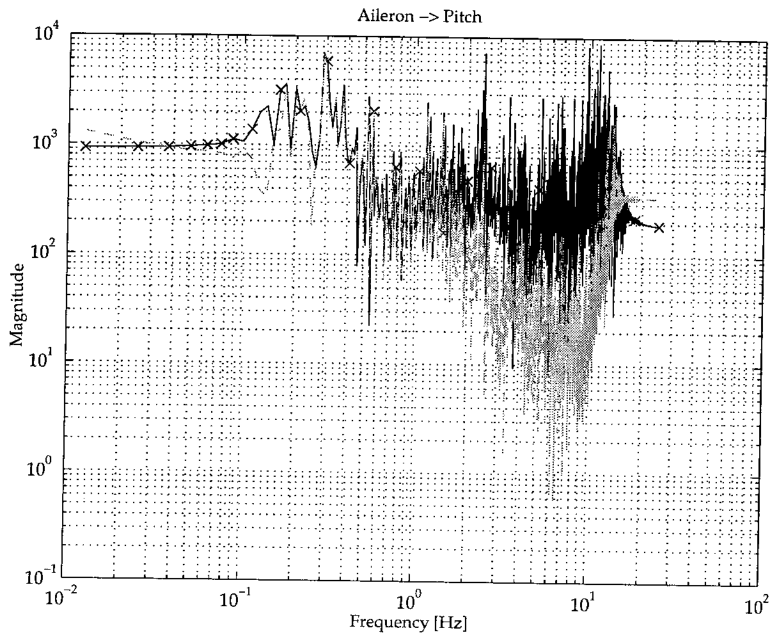

Figure 8.3: Helicopter hover transfer function: aileron $\rightarrow$ $\theta$. Experimental (solid), nominal simulation (dashed).

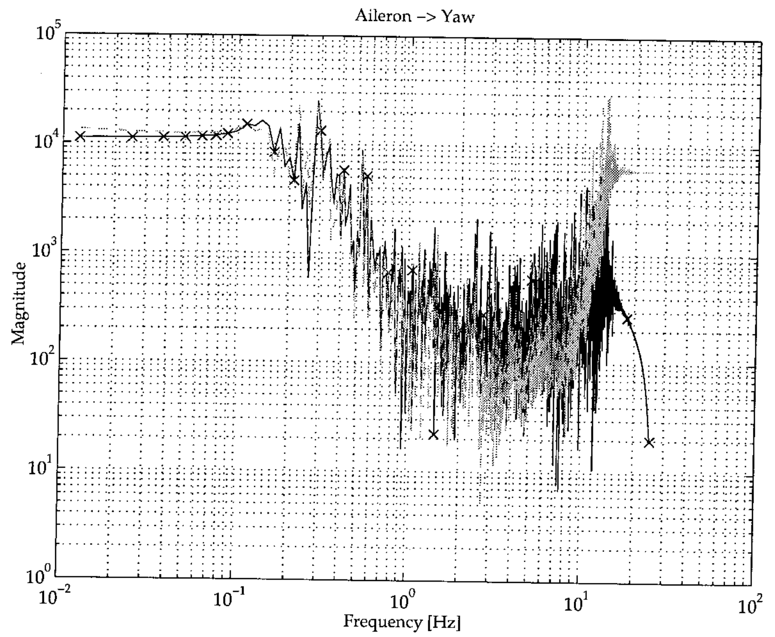

Figure 8.4: Helicopter hover transfer function: aileron $\rightarrow$ $\psi$. Experimental (solid), nominal simulation (dashed).

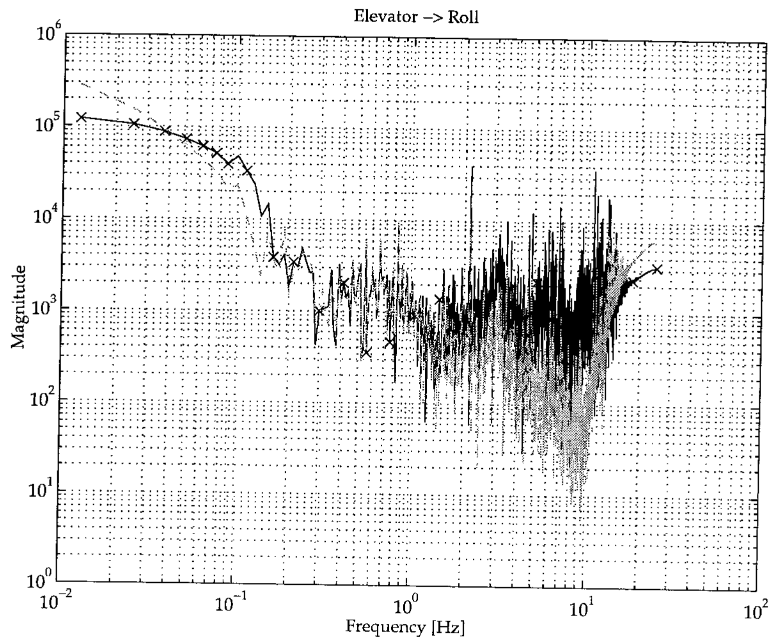

Figure 8.5: Helicopter hover transfer function: elevator $\rightarrow \phi$. Experimental (solid), nominal simulation (dashed).

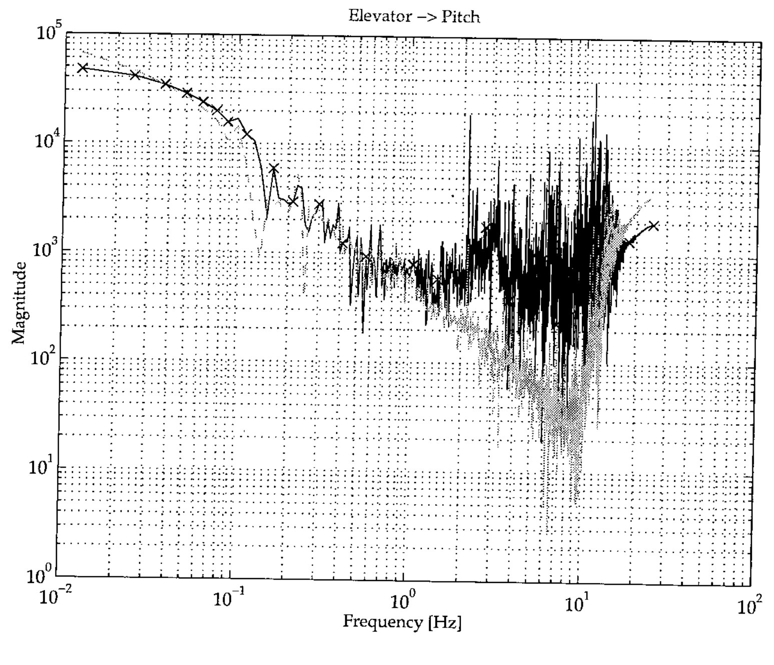

Figure 8.6: Helicopter hover transfer function: elevator $\rightarrow \theta$. Experimental (solid), nominal simulation (dashed).

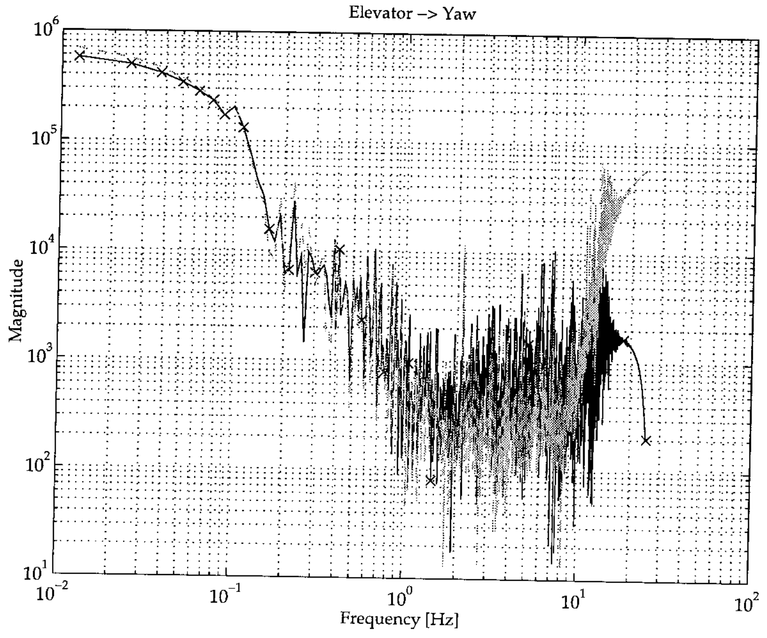

Figure 8.7: Helicopter hover transfer function: elevator $\rightarrow \psi$. Experimental (solid), nominal simulation (dashed).

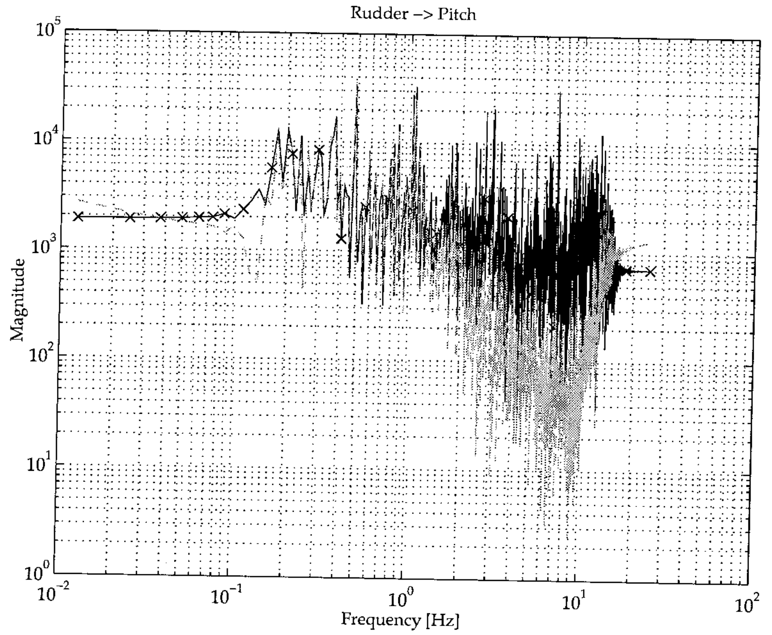

Figure 8.8: Helicopter hover transfer function: rudder $\rightarrow \phi$. Experimental (solid), nominal simulation (dashed).

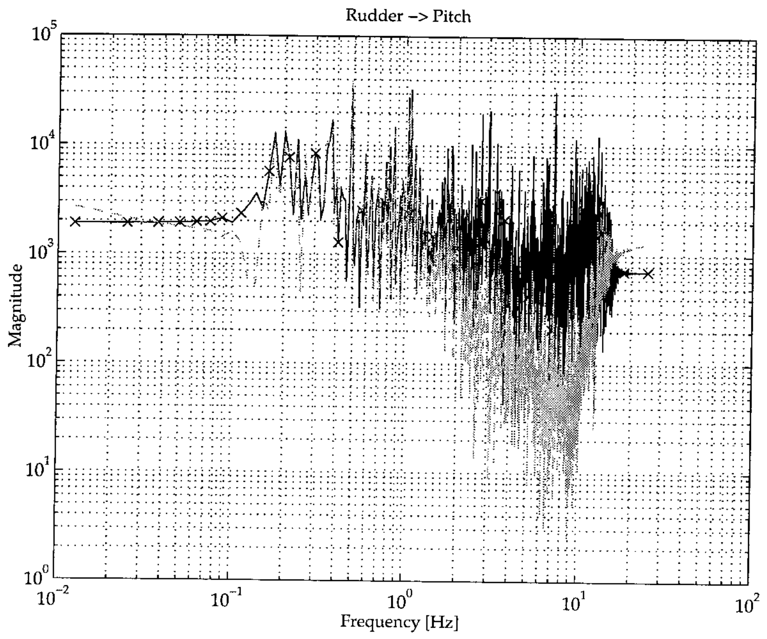

Figure 8.9: Helicopter hover transfer function: rudder $\rightarrow \theta$. Experimental (solid), nominal simulation (dashed).

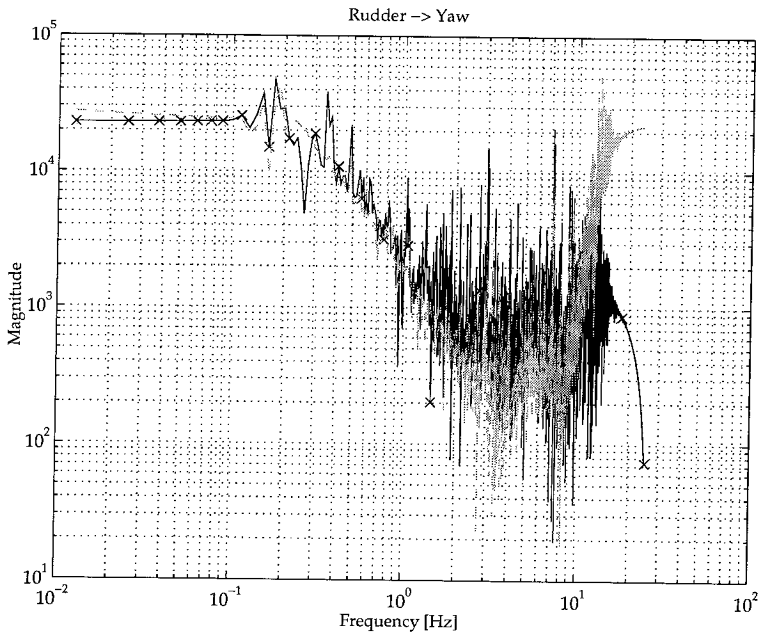

Figure 8.10: Helicopter hover transfer function: rudder $\rightarrow \psi$. Experimental (solid), nominal simulation (dashed).

*ad hoc* process. The difficulties with this method provided the motivation to develop a new process for designing robust models using model validation techniques.

In this section the performance, uncertainty, and noise weighting functions denoted $W_p$, $W_m$, and $W_n$, respectively, will be redesigned by carefully iterating between model validation and $\mu$-synthesis using a new design process. The values of the weights from the model used to design the $\mathcal{H}_\infty$ controller in Chapter 6 will be used as a starting point. In this chapter, this baseline model will be referred to as "$M_1$." The object of the redesign is to achieve $\mu_g \approx 1$ and $\mu \approx 1$, with both relatively flat across frequency, and such that $\mu(\omega) < \mu_g(\omega)$. Based on the model validation results presented in Chapter 7, if these conditions are satisfied this should produce a robust controller. The newly designed robust model for the helicopter will be denoted "$M_2$."

The interconnection used for $\mathcal{H}_\infty$ and $\mu$-synthesis, as shown in Figure 6.2, is reproduced in a slightly different form in Figure 8.11.

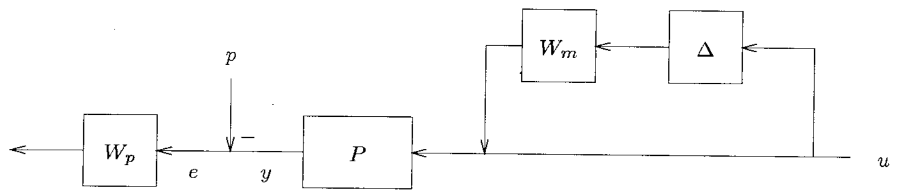

Figure 8.11: Helicopter synthesis interconnection: nominal model $P$, uncertainty $\Delta$, uncertainty weight $W_m$, control input $u$, measurement $y$, reference command $p$, error signal $e$, performance weight $W_p$.

$P$ is a continuous-time representation of the discrete-time nominal model $\hat{G}_3$ which was identified in Chapter 4. A bilinear transformation was used to convert $\hat{G}_3$ to continuous-time. A bilinear transformation was used so that norms would be preserved, at the expense of eliminating the structure of the state-space realization. $W_m$ weights $\Delta$, the input multiplicative uncertainty, and $W_p$ weights the output tracking error. $W_p$ will be designed to provide good step response tracking with zero steady-state error by incorporating integral action.

Note that the noise signal, $n$, and the associated weight, $W_n$, which were used in model validation (Figure 8.1) are not present in Figure 8.11. For the synthesis interconnection in Figure 8.11 this is acceptable so long as for the model validation problem $\max_\omega \|W_n(\omega)n(\omega)\|_2 \leq \frac{1}{\mu}$, where $\omega$ represents the frequency variable and $\mu$ results from a $\mu$-analysis of the robust model.

Before proceeding, recall that the general shape of the uncertainty and noise weights can be determined from Figures 8.2–8.10. It appears that plant mismatch (unmodelled dynamics or uncertainty) dominates below 0.2 Hz. There is significant noise above 1 Hz. The cross-over between dominant uncertainty and dominant noise appears to occur at about 0.5 Hz; however, it is likely that at high frequency noise, unmodelled dynamics, and uncertainty are equally important. This information can be used as a rule of thumb when designing the uncertainty and noise weights. In other words, we can assume a high-pass shape for $W_m$ and $W_n$, the weights on the noise and uncertainty, respectively. The zeroes will be chosen such that three conditions are satisfied: $(i)$ $\|W_n(\omega)n(\omega)\|_2 \leq \frac{1}{\mu}$ for all frequencies $\omega$, $(ii)$ $\|W_n(\omega)n(\omega)\|_2 < \overline{\sigma}(W_m(\omega)\Delta(\omega))$ at low frequencies, and $(iii)$ $\|W_n(\omega)n(\omega)\|_2 \approx \overline{\sigma}(W_m(\omega)\Delta(\omega))$ at high frequencies. The latter constraint is important, otherwise the noise can become unrealistically large relative to the uncertainty. For the helicopter experiment, which is dominated by uncertainty rather than noise, this is justified. To compare the relative sizes of noise and uncertainty we will plot $\|W_n(\omega)n(\omega)\|_2$ and $\overline{\sigma}(W_m(\omega)\Delta(\omega))$ at each frequency $\omega$ using the value of $n(\omega)$ and $\Delta(\omega)$ resulting from the model validation lower bound.

Based on these constraints, $W_n$ was designed to be a high-pass filter with a zero near 0.5 Hz, and the pole is used to keep the magnitude of $\|W_n(\omega)n(\omega)\|_2 \leq \frac{1}{\mu}$ above 0.5 Hz.

1. Adjust (or initialize) uncertainty weight $W_m$ and noise weight $W_n$. If initializing choose $W_m = 0$ and $W_n = I$.

2. Compute model validation bounds and uncertainty and noise signal achieving lower bound.

3. Verify uncertainty and noise satisfy any constraints or rules of thumb, e.g., $\|W_n(\omega)n(\omega)\|_2 << \|W_m(\omega)\Delta(\omega)\|_\infty$ at low frequencies. If necessary return to Step 1.

4. Adjust (or initialize) $W_p$. If initializing choose $W_p$ to reflect (nominal) performance specifications.

5. Design controller using $\mu$-synthesis.

6. Simulate the closed-loop system. If necessary return to Step 4.

7. Analyze the closed-loop system with $\mu$.

8. Verify $\mu(\omega) < \mu_g(\omega)$ for each relevant $\omega$. If necessary, return to Step 1 or 4.

9. Implement the controller. If necessary return to Step 1 or 4.

Table 8.1: Model validation design process.

The process used to design the weights $W_m$, $W_n$, and $W_p$ is summarized in Table 8.1. The goal of the design process is to choose $W_m$, $W_n$, and $W_p$ so that for each $\omega$ in the relevant frequency range $\mu(\omega) < \mu_g(\omega)$, with both relatively flat, and so that when simulated (or implemented) the closed-loop has good performance. Refer to Section 8.3 for a discussion of the performance specifications and details on how evaluation of the simulated responses was made. Because the model validation tools are defined for general linear uncertain systems, the process in Table 8.1 is generically applicable.

The process outlined in Table 8.1 was applied to iteratively design a new robust model and robust controller for the helicopter. After approximately three iterations between model validation and simulation, suitable weights were obtained and the final robust controller designed using $\mu$-synthesis performed acceptably when implemented. This was strikingly different from the *ad hoc* process employed in Chapter 6, where several controllers were implemented which simulated well but were destabilizing when implemented. A brief description of the first three iterations using this design process follows. Note that in the iterations, only the model validation upper bound is shown. As a practical matter we use the upper bound for guidance during intermediate iterations, as computation of the lower bound generally only converges 90% of the time.

## 8.1.1 Zeroth iteration designing a robust helicopter model

For the zeroth iteration it is assumed that $W_m = 0$: in other words analyze the nominal model in a robust framework. To do this we choose $W_n = I$ and compute the model validation lower and upper bounds. The resulting upper bound is shown in Figure 8.12.

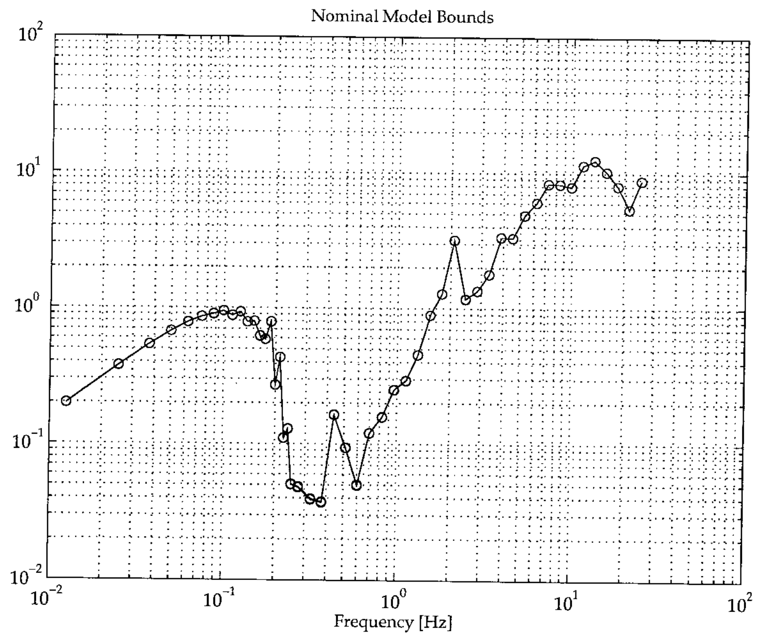

Figure 8.12: Zeroth iteration model validation upper bound for nominal helicopter model using filtered data. At each frequency, the upper bound lies between the two "o" data points corresponding to the final values of the upper bound LMI bisection.

From Figure 8.12 it is clear that above 2 Hz either noise or uncertainty appears.

In this frequency range $\mu_g \gtrapprox 2$ so we will choose $\|W_n\| \approx 0.5$ and proceed to the next iteration in the design process.

### 8.1.2 First iteration designing a robust helicopter model

For the first iteration we will choose $W_n = 0.5I$ based on the results from the previous iteration. As an initial guess we will choose $W_m = I$ and compute the model validation lower and upper bounds. The resulting upper bound is shown in Figure 8.13.

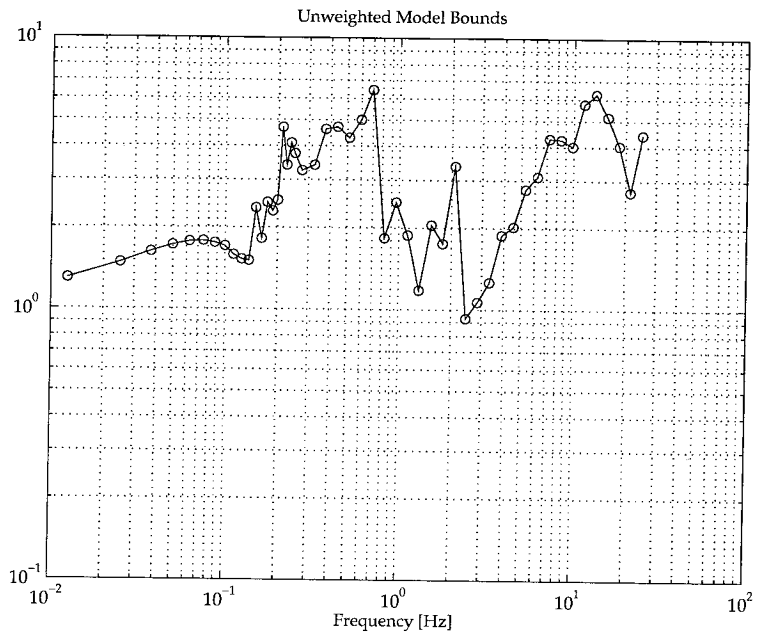

Figure 8.13: First iteration model validation upper bound for nominal helicopter model using filtered data. At each frequency, the upper bound lies between the two "o" data points corresponding to the final values of the upper bound LMI bisection.

From Figure 8.13 it is clear that below 0.1 Hz $\mu_g \gtrapprox 1.3$. Therefore, based on our rule of thumb that uncertainty is the dominant factor at low frequency, we will reduce the uncertainty weight by the factor $1/\mu_g \approx 0.75$.

### 8.1.3 Second iteration designing a robust helicopter model

For the second iteration we will choose $W_m = 0.75I$ and leave $W_n$ unchanged. The model validation lower and upper bounds are then computed. The resulting upper bound is shown in Figure 8.14. From Figure 8.14 it is clear that $\mu_g \gtrapprox 1$ with the exception of the narrow frequency range near 1.1 Hz. The solution to this is to increase either uncertainty or noise above 1 Hz with a zero.

Further iterations designing the robust helicopter model involved model validation analysis, controller synthesis, robustness analysis, and simulation, resulting in a

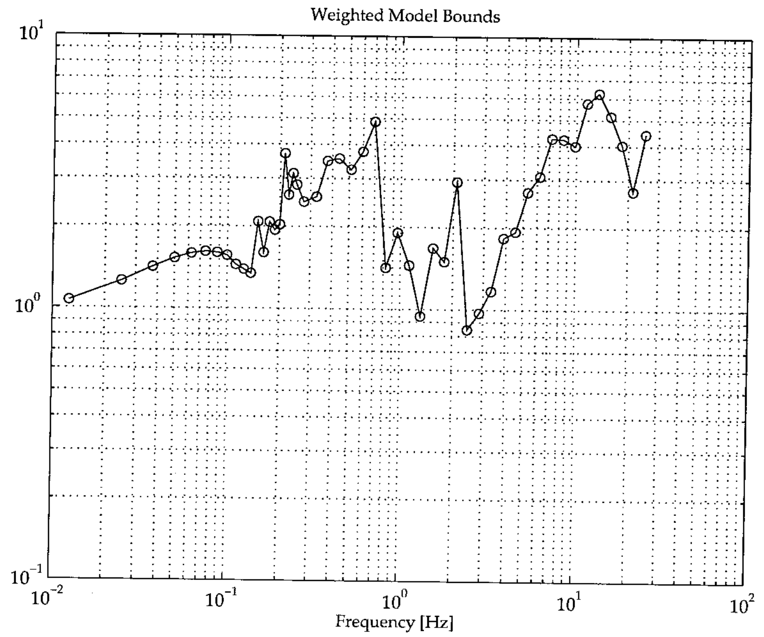

Figure 8.14: Second iteration model validation upper bound for nominal helicopter model using filtered data. At each frequency, the upper bound lies between the two "o" data points corresponding to the final values of the upper bound LMI bisection.

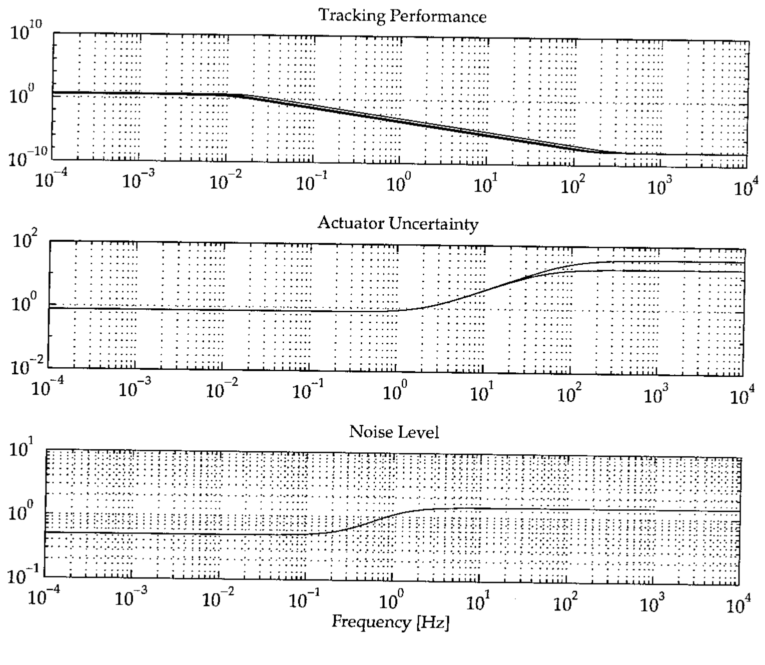

Figure 8.15: Weights used for helicopter model validation of $M_2$. Top: performance weight $W_p$. Middle: uncertainty weight $W_m$. Bottom: noise weight $W_n$.

selection of the zeros and poles for $W_m$ and $W_n$. The final choice for the weights, $W_m$, $W_n$, and $W_p$ are shown in Figure 8.15.

The model validation bounds computed for $M_2$ using filtered experimental data are shown in Figure 8.16, along with $\mu(M_2)$. The model validation lower and upper bounds are indicated by "x" and "o" data points, respectively. For each frequency $\omega$, $\mu(\omega) < \mu_g(\omega)$, hence the closed-loop system will be robust with respect to the uncertainty perturbations and noise which are necessary to guarantee that the model and data are consistent.

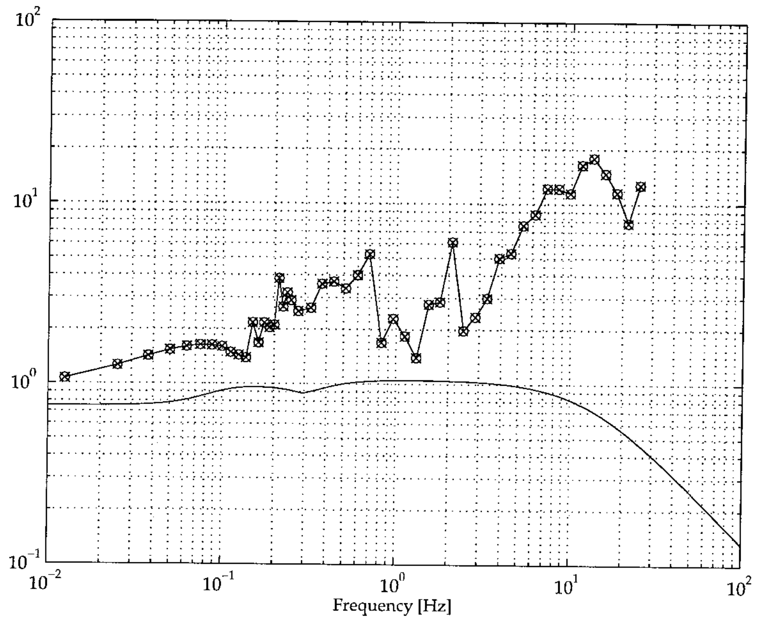

Figure 8.16: Bounds from helicopter model validation of $M_2$ using filtered data. At each frequency, the lower bound is indicated by "x" data points, the upper bound lies between the two "o" data points corresponding to the final values of the upper bound LMI bisection. $\mu_g(M_2)$ lies between the solid lines joining the "x" and "o" data points. $\mu(M_2)$ is indicated by the shaded line lying below $\mu_g(M_2)$.

The weighted noise and uncertainty computed by the lower bound are shown in Figure 8.17. At each frequency $\omega = 2\pi f$ the noise ($\|W_n(\omega)n(\omega)\|_2$) and uncertainty ($\overline{\sigma}(W_m(\omega)\Delta(\omega))$) are indicated by "x" and "o" data points, respectively. Notice that $\max_\omega \|W_n(\omega)n(\omega)\|_2 \approx 0.85$. As desired, the uncertainty dominates at low frequency (below 0.2 Hz), noise and uncertainty contribute roughly the same at mid frequency (0.5 Hz to 4 Hz), and uncertainty dominates at high frequency.

It might be possible to further decrease the amount of uncertainty in the robust model $M_2$ by adding real parametric uncertainty in the $A$ and $B$ matrices of the nominal model $P$ where the most significant parameter estimation error occurred when identifying the nominal model $\hat{G}_3$ (Table 4.1). After adding up to 25% uncertainty

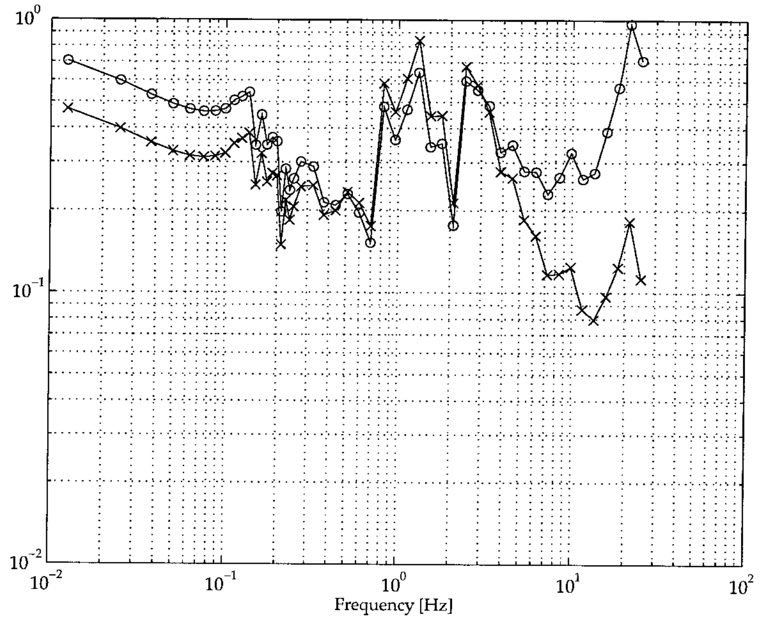

Figure 8.17: Model validation lower bound perturbation for $M_2$ using filtered data. At each frequency $f$, noise $(\|W_n(2\pi f)n(2\pi f)\|_2)$ and uncertainty $(\overline{\sigma}\,(W_m(2\pi f)\Delta(2\pi f)))$ are indicated by "x" and "o" data points, respectively.

in the principal axis parameters ($a_{11}$, $a_{22}$, and $a_{33}$, see Chapter 4) there was little change in the model validation bounds. For this problem there appears to be little significance to real parametric uncertainty.

## 8.2 Analysis of the helicopter hover models

As discussed in Chapter 7, model validation is concerned with determining if a robust model adequately captures the system dynamics implicitly contained in experimental data. With this in mind, the model validation tools were used to analyze the helicopter models to determine if the empirical observations of the robustness of each model and its corresponding controller were in any way connected to the data consistency predictions of the lower and upper bounds of $\mu_g$. In other words, is there a correlation between the models for which apparently robustly stabilizing controllers could be synthesized and models for which the model validations tools predict $\mu(\omega) \leq \mu_g(\omega)$? The negation of this question is equally important, and both ideas are further considered herein.

There are two open-loop interconnections which will be evaluated in this section: $M_1$ and $M_2$. Each of these models shares the same uncertainty interconnection with the only difference lying in the specific weighting functions $W_p$, $W_m$, and $W_n$ (see Figures 6.2 and 8.11 for synthesis and Figure 8.1 for model validation).

Based on empirical results obtained after implementing controllers which were designed with preliminary models, the performance and uncertainty weights, $W_p$ and $W_m$, respectively, were designed for $M_1$. $M_1$ is the same robust model discussed in Chapter 6, and was used to design the robust $\mathcal{H}_\infty$ hover controller first presented by Bendotti and Morris [8]. Controllers synthesized with $M_1$ performed well when simulated and more importantly, when implemented on the real helicopter, the performance compared well with the simulated predictions.

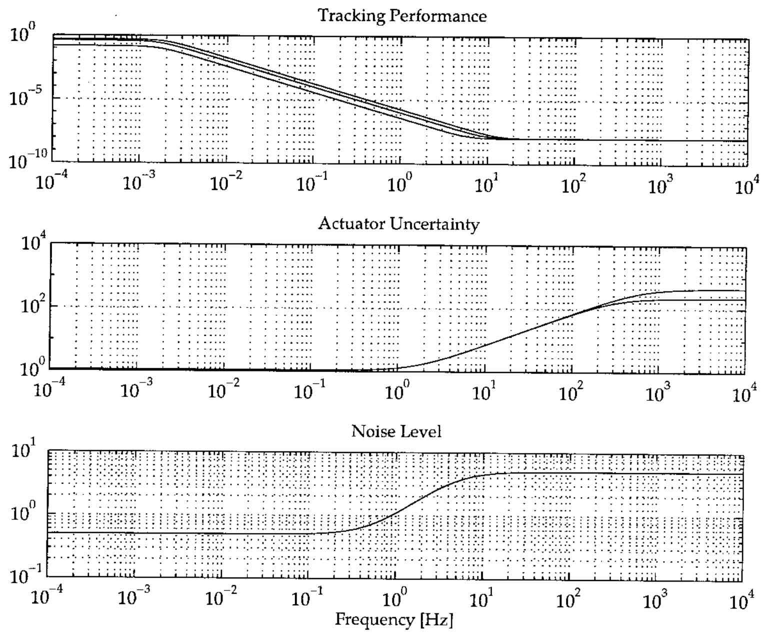

Figure 8.18: Weights used for helicopter model validation of $M_1$. Upper: performance weight $W_p$. Middle: uncertainty weight $W_m$. Bottom: noise weight $W_n$.

The specific form of $W_m$ and $W_p$ used in $M_1$ are given in (6.1) and (6.2), respectively (Figures 6.2 and 8.11). The noise weight, $W_n$, was chosen so that at high frequencies the noise and uncertainty were of comparable size and is given by

$$W_n = 0.50 \begin{pmatrix} w & 0 & 0 \\ 0 & w & 0 \\ 0 & 0 & w \end{pmatrix}, \quad \text{with } w = 10\frac{s+\pi}{s+10\pi}.$$

The weights used in $M_1$ are shown in Figure 8.18.

The model validation evaluation of $M_1$, as with $M_2$, is done using raw experimental data and filtered data. The filtered data corresponds to the same high-pass filtered data that was used to estimate the nominal model, $\hat{G}_3$, in Chapter 4. Plots of the model validation bounds and uncertainty are shown for the filtered data. Recall that filtered data tend to minimize corruption that results from drift in the experimental data. Because the helicopter is nonlinear, it is expected that the open-loop data will contain some drift as a result of varying trim. This implies that the filtered data should generate a more reliable model validation analysis. Nevertheless, results from the raw data are presented for comparison.

Figure 8.19 contains a plot of the resulting model validation bounds computed for the model $M_1$ using filtered experimental data; superimposed on this figure is a plot of $\mu(M_1)$. For each set of data 50 points are used, spaced logarithmically. In each of the following plots of the model validation lower and upper bounds an "x" corresponds to a data point for the lower bound, $\gamma_{lb}(\omega)$. An "o" corresponds to data points for the bounds on the upper bound LMI, i.e., at each frequency, $\omega$, there will be two "o" data points, and $\gamma_{ub}(\omega)$ lies between them. The gap between the "o" data points can be made arbitrarily small by appropriate choice of tolerance for the $\gamma$ bisection in the upper bound LMI.

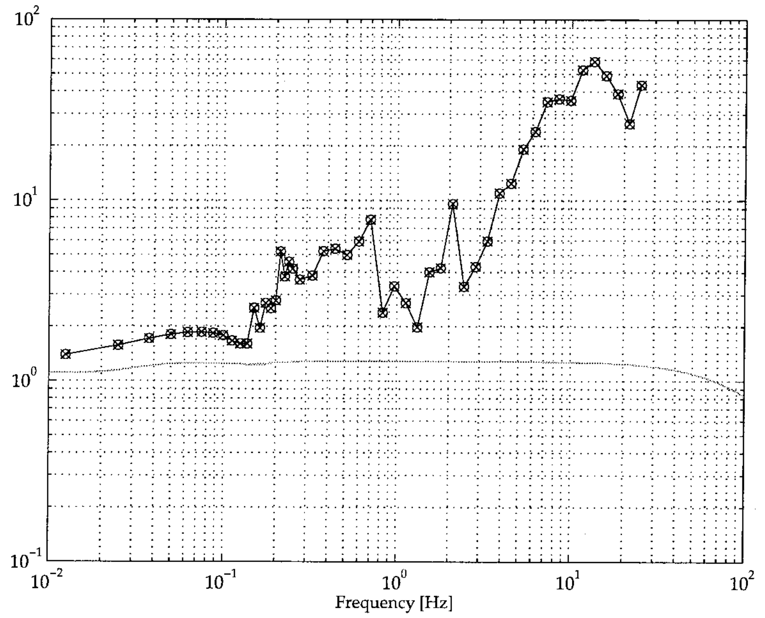

Figure 8.19: Bounds from helicopter model validation of $M_1$ using filtered data. At each frequency, the lower bound is indicated by "x" data points, the upper bound lies between the two "o" data points, corresponding to the final values of the upper bound LMI bisection. $\mu_g(M_1)$ lies between the solid lines joining the "x" and "o" data points. $\mu(M_1)$ is indicated by the shaded line lying below $\mu_g(M_1)$.

For $M_1$, which was used to design the $\mathcal{H}_\infty$hover controller presented in Chapter 6, $\mu_g \approx 1.39$ and $\mu \approx 1.28$. Because at each frequency $\mu(\omega) < \mu_g(\omega)$ (Figure 8.19), it is clear that the controller will be robust with respect to the noise and uncertainty necessary to achieve consistency between the model and the experimental data, i.e., the controller is robust. In fact, when implemented, the controller was very stable, with acceptable performance.

The weighted noise and uncertainty computed from the model validation lower bound are shown in Figure 8.20. The noise and uncertainty are indicated by "x" and "o" data points, respectively. Notice that $\max_\omega \|W_n(\omega)n(\omega)\|_2 \approx 0.68$, and that uncertainty dominates at low frequency, noise and uncertainty contribute roughly the same at mid frequency, and uncertainty dominates at high frequency.

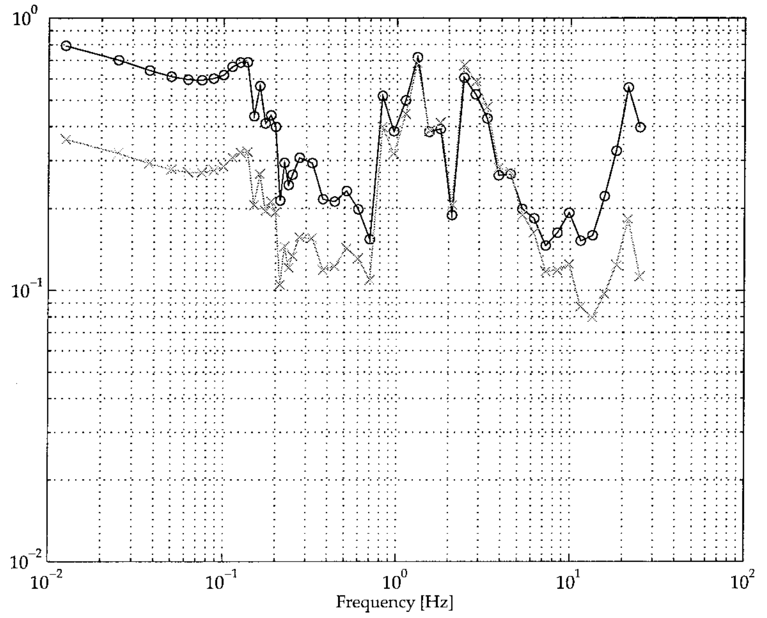

Figure 8.20: Model validation lower bound perturbation for $M_1$ using filtered data. At each frequency $f$, noise ($\|W_n(2\pi f)n(2\pi f)\|_2$) and uncertainty ($\overline{\sigma}\left(W_m(2\pi f)\Delta(2\pi f)\right)$) are indicated by "x" and "o" data points, respectively.

Notice in Figure 8.19 that $\mu_g$ is not flat across frequency. It appears that there is far too much uncertainty or noise at high frequency. A first step to reduce the amount of uncertainty in the robust model would be to reduce the high frequency content in the uncertainty and noise weights. This is the sort of tradeoff that becomes clear using model validation.

For the robust model, $M_2$, the structure of the interconnection was preserved, but the weighting functions were redesigned to minimize the overall uncertainty without sacrificing the fidelity of the model. The model validation bounds for $M_2$ are shown in Figure 8.16 and the corresponding weighted noise and uncertainty perturbations are shown in Figure 8.17. Synthesis of controllers based on $M_2$ is explored in Section 8.3.

The model validation results for each of the robust models are collected in Table 8.2. A model is considered to be robust if for each frequency $\omega$ $\mu(\omega) < \mu_g(\omega)$. It is not necessary that $\mu < \mu_g$, because the frequency at which $\mu$ is maximum may be different than the frequency at which $\mu_g$ is minimum.

It is important to recall that the helicopter model $M_1$ was arrived at by adjusting uncertainty and performance weights until satisfactory closed-loop behavior was

| Model | Model validation $\mu_g$ |  | $\mu$ | $\|\|W_n n\|\|$ | Robust |
|---|---|---|---|---|---|
|  | Raw Data | Filtered Data |  |  |  |
| $M_1$ | 1.08 | 1.39 | 1.28 | 0.68 | Yes |
| $M_2$ | 0.74 | 1.06 | 1.06 | 0.85 | Yes |

Table 8.2: Helicopter model validation bounds. For each model the minimum value of the lower bound is shown for raw and filtered data along with the value of $\mu$ and the maximum scaled noise signal $||W_n n|| = \max_\omega ||W_n(\omega)n(\omega)||_2$ resulting from the lower bound. If $\mu(\omega) < \gamma_{lb}(\omega)$, $\forall \omega$ then the model is considered to be robust.

obtained when the controller was implemented. The choice of weighting functions relied primarily on an empirical and qualitative assessment of the closed-loop behavior of the controller when implemented on the model helicopter. Before obtaining $M_1$, many other models were evaluated and controllers designed and implemented which were unable to stabilize the actual helicopter, even though simulations predicted great closed-loop behavior for both the nominal system and a perturbed system using worst-case perturbations from a $\mu$-analysis. It is exactly this inability of standard robust control techniques to produce measures which qualify the robust model and robust controller with respect to actual experimental data that motivated the development of tools which bridge this gap.

## 8.3 Robust hover controller

The robust hover controller was designed through $\mu$-synthesis using the process outlined in Section 8.1 (Table 8.1). The main control objective is to reject disturbances, track computer generated reference commands on the helicopter attitude, and decouple the modes at hover.

The performance weights for the helicopter were chosen such that it responded to commands using a bank-to-turn technique, with yaw used to reduce the couplings between pitch and roll. Thus, the controller is designed such that the dynamics of yaw are fast relative to those of pitch and roll.

When designing the controller, a key factor prior to implementation was to guarantee that the actuators were not over utilized. For a 5 degree step on the pilot commands, the rule of thumb employed was to allow the aileron and elevator to use no more than 50% and the rudder 100% of the maximum servo command.

The controller was designed using $\mu$-synthesis. For the final value of the weights, presented in Section 8.1, the initial $\mathcal{H}_\infty$ controller achieved $\gamma = 1.93$. One *D-K* iteration (Table 5.1) was completed. The $D$ scales were fit with a second order transfer function and the resulting controller had 27 states.

Figure 8.21 shows the step-responses of the closed-loop control system, where the controller was synthesized using the model designed in Section 8.1. Input–output signals corresponding to simulations of $\hat{G}_3$ and $\hat{G}_6$ are plotted with solid lines and mixed lines, respectively. The reference signals are plotted with dashed lines.

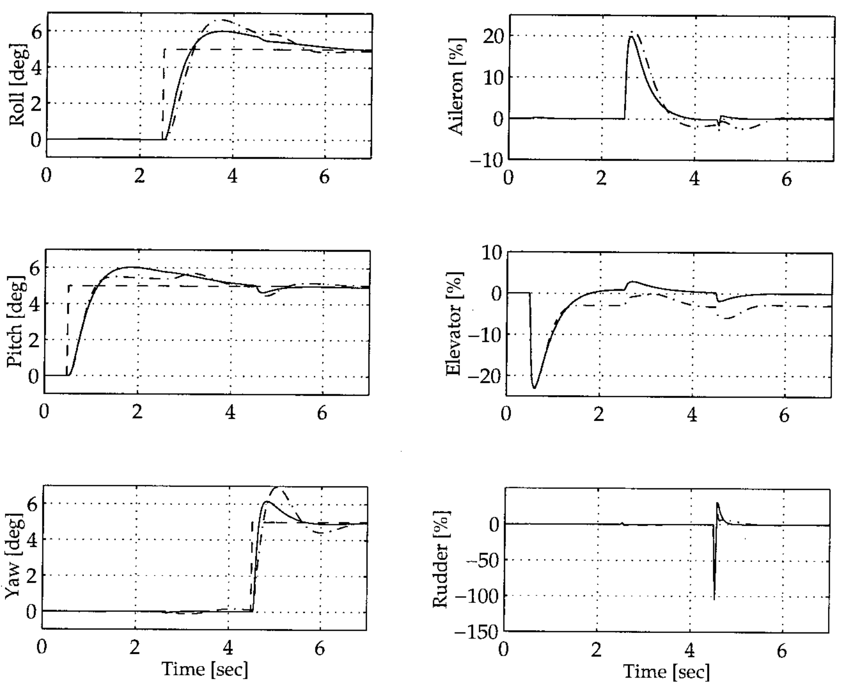

Figure 8.21: Simulated step responses: nominal (solid), perturbed (mixed), command (dashed).

The frequency response of the $\mu$ controller is shown in Figure 8.22. Like the $\mathcal{H}_\infty$ controller in Figure 6.5, the controller rolls off rapidly above the mid frequency range.

The closed-loop system was analyzed with $\mu$, using the same uncertainty structure as in (6.4). The bounds for $\mu$ are plotted in Figure 8.23. The peak value of $\mu$ is approximately 1.06.

Prior to implementation, the continuous-time $\mu$ controller was converted to discrete-time with a bilinear transform. As with the $\mathcal{H}_\infty$ and LQG controllers, the $\mu$ controller was implemented on the model helicopter in the three DOF configuration. The helicopter was first piloted to hover. Then the controller was switched on and trimmed. Exogenous reference commands on roll, pitch, and yaw were provided by the real-time computer, using look-up tables. The commands consisted of one positive and one negative pulse of 5 degrees on each axis independently.

Figure 8.24 shows the experimental step response of the helicopter; superimposed on this figure are the simulated responses of the three and six degree-of-freedom models. The step response performance of the $\mu$ controller in implementation is relatively good, except that there is a very low amplitude high frequency ringing. This could be reduced by further iteration on the performance weight $W_p$.

The experimental control signals generated by the $\mu$ controller are shown in Figure 8.25. Both implementation and simulated responses are shown with the same

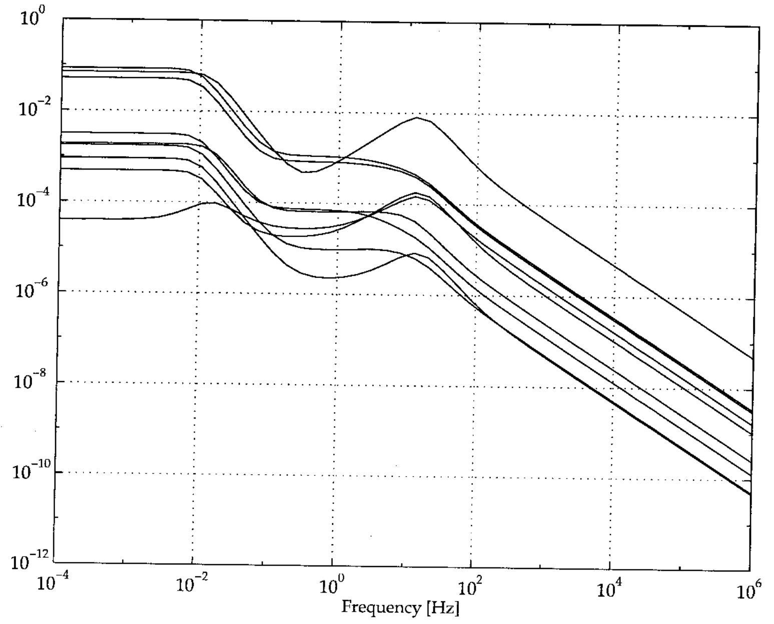

Figure 8.22: Magnitude of the frequency response of the $\mu$ controller.

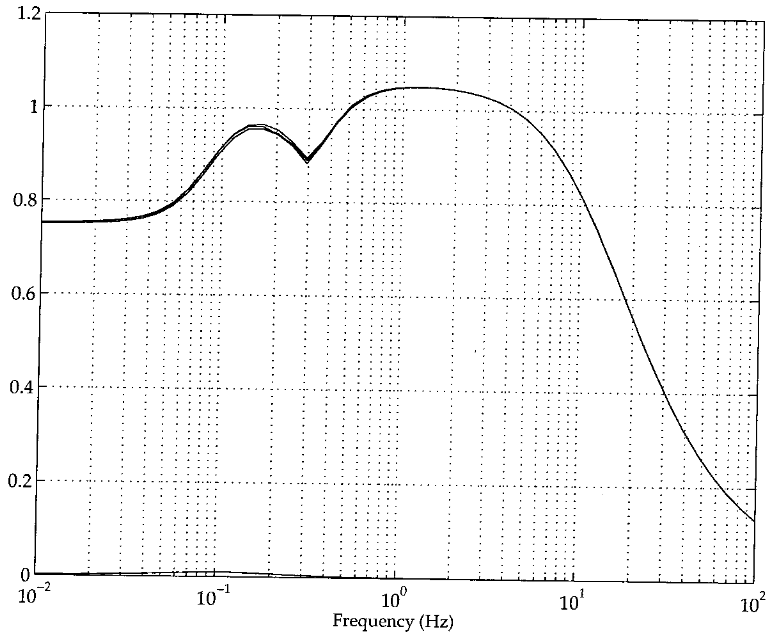

Figure 8.23: Closed-loop $\mu$-analysis of helicopter with $\mu$ controller.

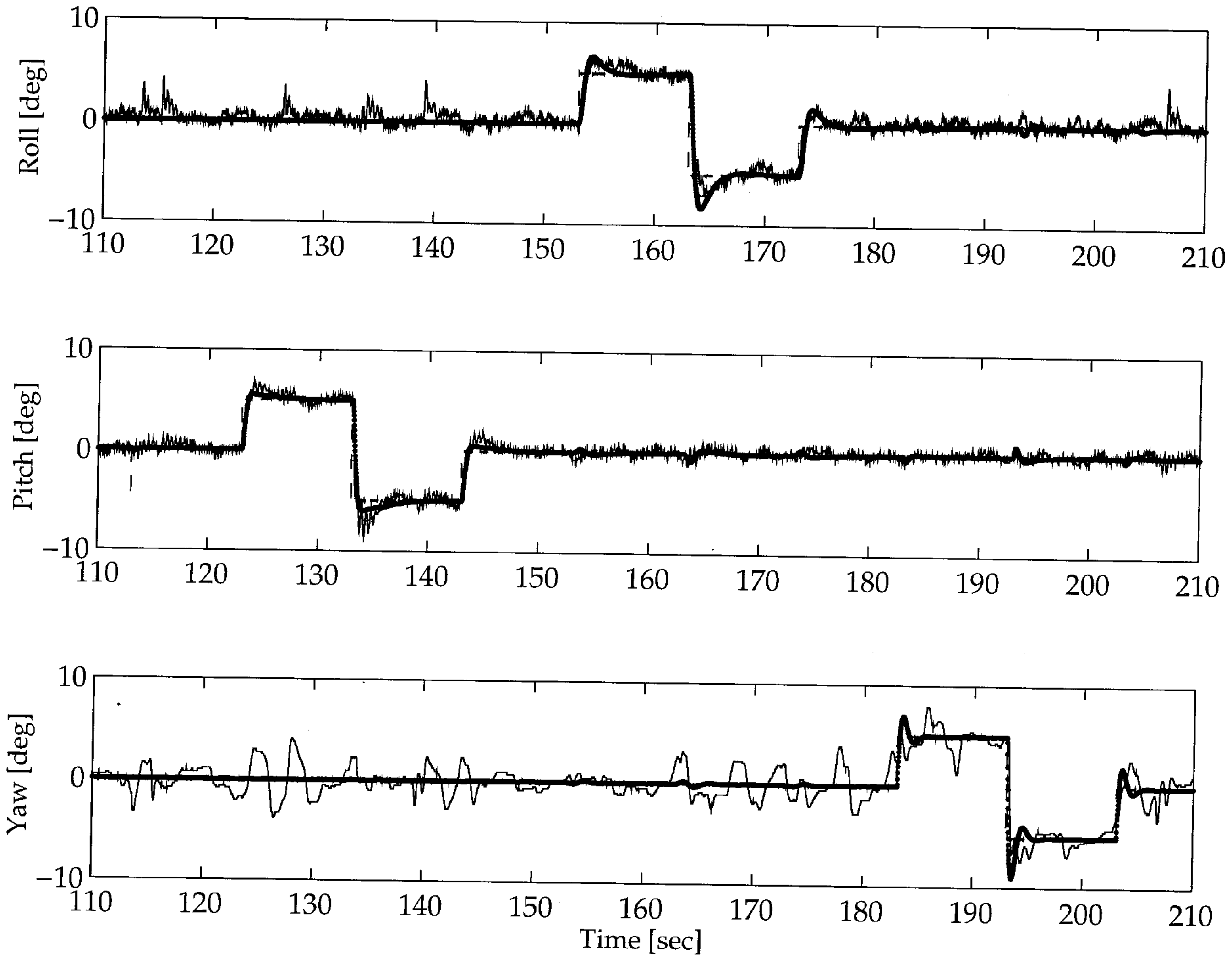

Figure 8.24: Experimental step responses using the $\mu$ controller: experimental (solid), commanded (dashed), $\hat{G}_3$ simulation (mixed), $\hat{G}_6$ simulation (dotted).

conventions as in Figure 8.24.

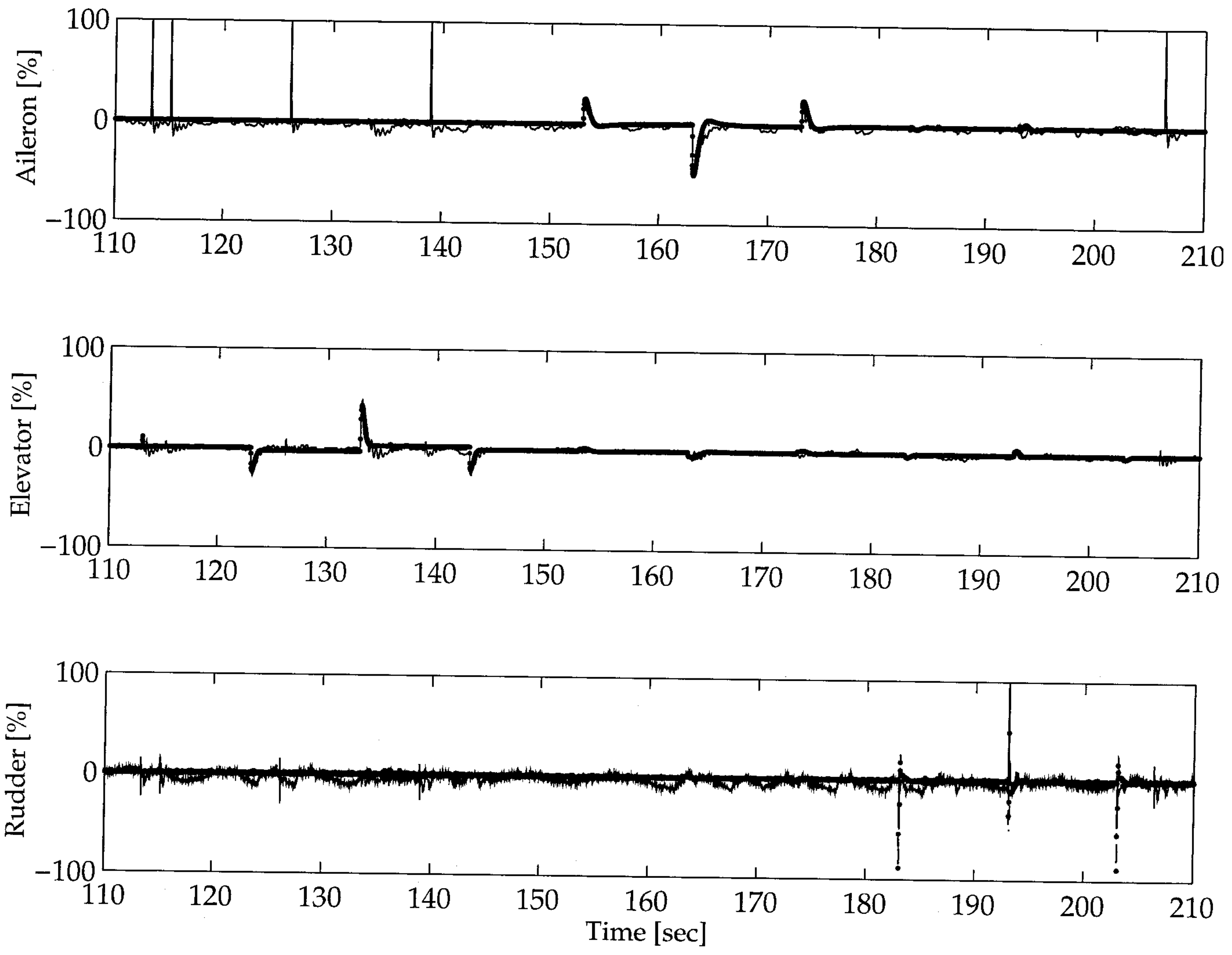

Figure 8.25: Experimental control signals for the $\mu$ controller resulting from step commands: experimental (solid), $\hat{G}_3$ simulation (mixed), $\hat{G}_6$ simulation (dotted).

As with the LQG and $\mathcal{H}_\infty$ controllers, disturbances were simulated by tapping on the helicopter. The $\mu$ controller rejected disturbances as well as the $\mathcal{H}_\infty$ controller. The disturbance rejection results are not shown.

Performance measures for the $\mu$, $\mathcal{H}_\infty$ , and LQG controllers are summarized in Table 8.3. The performance of each controller is compared by measuring the rise time, overshoot, and settling time on each axis. The rise time is computed from 10% to 90%, overshoot is computed in the standard fashion, and settling time is defined as the time it takes to fall below 15% ripple on the steady-state value.

| | $\mu$ | $\mathcal{H}_\infty$ | **LQG** |
|---|---|---|---|
| $\phi$ rise time, seconds | 0.38 | 0.54 | 0.54 |
| $\phi$ settling time, seconds | 4.74 | 7.8 | 7.7 |
| $\phi$ overshoot, % | 26 | 22 | 30 |
| $\theta$ rise time, seconds | 0.44 | 0.52 | 1.34 |
| $\theta$ settling time, seconds | 2.82 | 3.70 | 5.0 |
| $\theta$ overshoot, % | 38 | 20 | 23 |
| $\psi$ rise time, seconds | 0.30 | N/A | N/A |
| $\psi$ settling time, seconds | N/A | N/A | N/A |
| $\psi$ overshoot, % | 63 | N/A | N/A |

Table 8.3: Summary of controller performance measures for step commands.

## 8.4 Summary

A more complete example of the importance of using model validation to verify the applicability of robust models was shown with an experimental model helicopter. It was found that when designing controllers using robust models based solely on engineering judgement and qualitative adjustment, many iterations of implementation may be needed to arrive at a suitable robust model.

The design process outlined in this chapter, incorporating model validation for the first time, can be used to more effectively design robust controllers by guaranteeing that the robust model is consistent with all of the observed experimental data before implementing any controllers. Thus, more iterations in the design cycle can be performed on the computer. This is of particular relevance when the system is unstable; as in the case of the experimental model helicopter where it took several iterations just to produce a stabilizing controller. Some controllers were implemented which simulated well, but resulted in the helicopter crashing; in some cases the helicopter was actually damaged.

Using methods like model validation early in the design cycle should minimize problems like this by allowing one to start the design cycle with better models. The use of model validation will not necessarily eliminate hand tweaking of model parameters to optimize the performance and robustness characteristics of the resulting closed-loop system. However, it brings one to a better starting point, speeding up the overall design process. This is the advantage of the model validation technique: it brings the "art" of weighting function selection closer to "science".

The design process, outlined in Table 8.1, was successfully applied to the helicopter and reduced the size of uncertainty in the robust model, without sacrificing performance and stability robustness when implemented. The first controller designed with this process worked when implemented on the helicopter, and was, in fact, the best controller overall. This was strikingly different from the results obtained when using standard *ad hoc* robust control techniques, where several controllers destabilized the helicopter when implemented, even though they performed well under simulation. It is interesting to note that when the model validation tools were used to assess single axis reductions of the MIMO helicopter model, e.g., the pitch axis transfer function $\Theta_e \rightarrow \Theta$, the single-axis model was often invalidated. This implies that trivializing the dynamics to SISO is not feasible for the helicopter. This illustrates again that thinking of MIMO systems a "loop-at-a-time" is often not adequate.

The methods discussed in this chapter were concerned primarily with using model validation to develop a robust model for an open-loop system. These techniques can only provide a good model for robustness, not performance, i.e., there may be the need to use the model validation tools to iterate on the closed-loop data to design better performance weights.

An important aspect of this design process is the data set. The robustness test, $\mu(\omega) < \mu_g(\omega) \forall \omega$, is only as good as the data. If the data are not representative of the system then the model validation robustness results will misleading. This may be exploited in flight test programs: the model validation tools may be able to provide a much needed method for assessing the relevance of flight test data.

# Chapter 9 Conclusions

Why, upon my word, that I am very far from supposing that I know the explanation of any of these things. I cannot even convince myself that when you add one to one either the first or the second one becomes two, or they both become two by the addition of the one to the other ... Nor can I now persuade myself that I understand how it is that things become one, nor, in short, why anything else comes or ceases or continues to be, according to this method of inquiry. So I reject it altogether, and muddle out a haphazard method of my own.

—Plato, *Phaedo*, translated by Hugh Tredennick

The inability of standard system identification techniques to determine if a robust model has any relevance to the actual system under control has limited the widespread adoption of robust synthesis techniques by the control engineering community. Control engineers have not been able to make effective use of the knobs (weighting functions) when designing robust models and synthesizing controllers using robust control techniques because there were never any tools which showed them how adjusting the knobs affected the ability of the model to match the experimental data. Thus, the control engineer was forced to use two disconnected techniques in control: system identification and robust control, leading to *ad hoc* solutions.

The clear advantage of using model validation methods lies in an *a priori* understanding of whether the robust model adequately captures the system dynamics implicitly contained in the experimental data. In this way, when standard robust control synthesis techniques are applied to the robust model, which has been evaluated with model validation techniques, the robustness measures are directly comparable to the measures generated by the model validation tools. This connection builds a bridge between nominal and uncertain system modelling and robust control theory with the simple test: "Is $\mu(\omega) < \mu_g(\omega)$ for all relevant frequencies?"

A design process was developed which incorporated frequency-domain model validation analysis, $\mu$-analysis and $\mu$-synthesis, simulation, and implementation. This design process proved to be a valuable new tool for designing control systems for the helicopter. In particular, by applying this design process, the size of uncertainty in the robust model for the helicopter was substantially reduced over several iterations. After the design cycle was completed, the first controller implemented on the helicopter worked and was the best controller overall. This was substantially different from the results obtained when using standard robust control techniques, where several controllers destabilized the helicopter when implemented, even though they performed well under simulation. The model validation design process was also successfully applied to other experimental systems, including the ducted fan apparatus by Bodenheimer *et al.*, described in Chapter 7 and a two-mass oscillator by Eich [22].

The advantage of the design process developed in this dissertation is that, for the first time, the control engineer can place more confidence in and rely upon simulation; model validation guarantees that the robust model is able to generate all observed data on the physical system. In other words, the principal design work can be more reliably based on simulation. The result is that fewer implementations of the control system are necessary during the design cycle, limiting the chance of catastrophic failure. For the control engineer this process brings the art of weight selection closer to a science. However, because these measures are data dependent it becomes increasingly important to collect data that adequately capture the system dynamics, otherwise the robustness measures may be misleading.

To further increase the utility of this design process, the model validation lower bound power algorithm must be improved. When the uncertainty description incorporates real or complex repeated scalars there are problems within the implicit loop. Although the algorithm appears to readily converge in many cases, there were cases for which convergence was a significant problem. Further improvement in the speed of the LMI solver for the model validation upper bound would also be helpful. For the helicopter, the upper bound generally took several minutes of computation on standard workstations with only 50 data points. This is just long enough to make it an unpleasant tool in an iterative design cycle.

A future goal of this research program is to develop control methodologies and synthesis techniques which allow fully autonomous rotorcraft flight, including hover, take-off and landing, tracking, and obstacle detection and avoidance. The first step will be developing a new helicopter testbed with adequate sensing, instrumentation, embedded real-time flight control systems, and telemetry to allow free flight. Once such a testbed is available, new models could be developed for the helicopter in the major flight modes using techniques similar to those employed herein.

# Bibliography

[1] Advanced Micro Devices. *Am9513A/Am9513 Technical Manual.* Revision 03402E.

[2] Pierre Apkarian and Pascal Gahinet. A convex characterization of parameter dependent $\mathcal{H}_\infty$ controllers. *IEEE Transactions on Automatic Control*, 1995. To appear.

[3] Pierre Apkarian, Pascal Gahinet, and Greg Becker. $\mathcal{H}_\infty$ control of linear parameter-varying systems: A design example. *Automatica*, 1995. Submitted.

[4] Pierre Apkarian, Pascal Gahinet, and J. Biannic. Self-scheduled $\mathcal{H}_\infty$ control of a missile via LMIs. In *Proceedings of the IEEE Control and Decision Conference*, pages 3312–3317, 1994.

[5] Gary Balas, John C. Doyle, Keith Glover, Andy K. Packard, and Roy Smith. *$\mu$-Analysis and Synthesis Toolbox*, April 1991.

[6] David S. Bayard. Statistical plant set estimation using schroeder-phased multisinusoidal input design. In *Proceedings of the American Control Conference*, pages 2988–2995, June 1992.

[7] Carolyn Beck. Computational issues in solving LMIs. In *Proceedings of the IEEE Control and Decision Conference*, pages 1259–1260, 1991.

[8] Pascale Bendotti and John C. Morris. Robust hover control for a model helicopter. In *Proceedings of the American Control Conference*, 1995.

[9] Robert E. Bodenheimer, Jr. *The Whirling Blade and the Steaming Cauldron.* PhD thesis, California Institute of Technology, 1995.

[10] Robert E. Bodenheimer, Jr., Pascale Bendotti, and Michael Kantner. Linear parameter-varying control of a ducted fan engine. Technical Report CIT-CDS 95-004, California Institute of Technology, January 1995.

[11] Victor H. L. Cheng. Obstacle-avoidance automatic guidance: A concept-development study. In *AIAA Guidance, Navigation, and Control Conference*, August 1988.

[12] Victor H. L. Cheng and Banavar Sridhar. Considerations for automated nap-of-the-earth rotorcraft flight. In *Proceedings of the American Control Conference*, pages 967–976, June 1988.

[13] Victor H. L. Cheng and Banavar Sridhar. Integration of active and passive sensors for obstacle avoidance. *IEEE Control Systems Magazine*, 10(4):43–50, June 1990.

[14] Henry O. Choi, Peter Sturdza, and Richard M. Murray. Design and construction of a small ducted fan engine for nonlinear control experiments. In *Proceedings of the American Control Conference*, pages 2618–2622, 1994. Available electronically from `http://avalon.caltech.edu/~dfan`.

[15] Urs Christen, Martin F. Weilenmann, and Hans P. Geering. Design of $\mathcal{H}_2$ and $\mathcal{H}_\infty$ controllers with two degrees of freedom. In *Proceedings of the American Control Conference*, pages 2391–2395, 1994.

[16] John C. Doyle. Analysis of feedback systems with structured uncertainties. *IEE Proceedings*, 129, Part D(6):242–250, November 1982.

[17] John C. Doyle. Structured uncertainty in control system design. In *Proceedings of the IEEE Control and Decision Conference*, December 1985.

[18] John C. Doyle, Bruce A. Francis, and Allen R. Tannenbaum. *Feedback Control Theory*. Macmillan, 1992.

[19] John C. Doyle, Keith Glover, Pramod P. Khargonekar, and Bruce A. Francis. State-space solutions to standard $\mathcal{H}_2$ and $\mathcal{H}_\infty$ control problems. *IEEE Transactions on Automatic Control*, 34(8):831–847, August 1989.

[20] John C. Doyle, Kathryn Lenz, and Andy K. Packard. Design examples using $\mu$-synthesis: Space shuttle lateral axis FCS during reentry. In *Proceedings of the IEEE Control and Decision Conference*, pages 2218–2223, 1986.

[21] John C. Doyle, Andy K. Packard, and Kemin Zhou. Review of LFTs, LMIs, and $\mu$. In *Proceedings of the IEEE Control and Decision Conference*, pages 1227–1232, 1991.

[22] Jürgen Eich. Model validation and robust control for a two-mass oscillator with dry friction. Technical report, Department for Flight Mechanics and Automatic Control, Darmstadt Technical University, February 1996.

[23] Dale Enns. Multivariable flight control for an attack helicopter. In *Proceedings of the American Control Conference*, pages 858–863, 1986.

[24] Kuang-Hua Fu and Juergen Kaletka. Frequency-domain identification of BO 105 derivative models with rotor degrees of freedom. *Journal of the American Helicopter Society*, pages 73–83, January 1993.

[25] Pascal Gahinet, Arkadi Nemirovskii, Alan J. Laub, and Mahmoud Chilali. The LMI control toolbox. In *Proceedings of the IEEE Control and Decision Conference*, pages 2038–2041, 1994.

[26] Pascal Gahinet, Arkadii Nemirovskii, Alan J. Laub, and Mahmoud Chilali. *The LMI Control Toolbox*, November 1994. Beta release.

[27] A. Gessow and G. Myers. *Aerodynamics of the Helicopter.* Frederick Ungar Publishing Company, 1952.

[28] Keith Glover and John C. Doyle. State-space formulae for all stabilizing controllers that satisfy an $\mathcal{H}_\infty$-norm bound and relations to risk sensitivity. *Systems and Control Letters*, 11:167–172, 1988.

[29] N. Gupta. Frequency-shaped cost functionals: Extension of linear-quadratic-gaussian methods. *Journal of Guidance*, 3:529–535, 1980.

[30] John E. Hauser. Nonlinear control via uniform system approximation. *Systems and Control Letters*, 17:145–154, 1991.

[31] John E. Hauser and Richard M. Murray. Nonlinear controllers for non-integrable systems: the acrobot example. In *Proceedings of the American Control Conference*, May 1990.

[32] John E. Hauser, S. Shankar Sastry, and Petar V. Kokotović. Nonlinear control via approximate input-output linearization: the ball and beam example. In *Proceedings of the IEEE Control and Decision Conference*, pages 1987–1993, December 1989.

[33] John E. Hauser, S. Shankar Sastry, and George Meyer. Nonlinear controller design for flight control systems. Technical report, UC Berkeley, 1988.

[34] Hewlett Packard. *Hewlett Packard Optoelectronics Designer's Catalog 1988–1989.* Revision 5954–8450.

[35] Stewart S. Houston and Colin G. Black. Identifiability of helicopter models incorporating higher-order dynamics. *Journal of Guidance, Control, and Dynamics*, 14(4):840–847, 1991.

[36] Louis R. Hunt, Renjeng Su, and George Meyer. Global transformations of nonlinear systems. *IEEE Transactions on Automatic Control*, 28(1):24–32, 1983.

[37] Alberto Isidori. *Nonlinear Control Systems.* Springer-Verlag, 2nd edition, 1989.

[38] IXYS Corporation. *IXSE502/IXSE503 Shaft Encoder.* Revision SJ90536.

[39] Wayne Johnson. *Helicopter Theory.* Princeton University Press, 1980.

[40] Arthur J. Krener. Approximate linearization by state feedback and coordinate change. *Systems and Control Letters*, 5:181–185, 1984.

[41] Arthur J. Krener, S. Karahan, and M. Hubbard. Approximate normal forms of nonlinear systems. In *Proceedings of the IEEE Control and Decision Conference*, December 1989.

[42] Michael S. Lewis and Edwin W. Aiken. Piloted simulation of one-on-one helicopter air combat at NOE flight levels. Technical Report 86686, NASA Ames Research Center, April 1985.

[43] Michael S. Lewis, M. Hossein Mansur, and Robert T. N. Chen. A piloted simulation of helicopter air combat to investigate effects of variations in selected performance and control response characteristics. Technical Report 89438, NASA Ames Research Center, August 1987.

[44] Lennart Ljung. *System Identification, Theory for the User.* Prentice Hall, 1987.

[45] Lennart Ljung. *System Identification Toolbox.* MathWorks, 1991.

[46] Eicher Low and William L. Garrard. Design of flight control systems to meet rotorcraft handling qualities specifications. *Journal of Guidance, Control, and Dynamics*, 16(1):69–78, January–February 1993.

[47] John C. Morris and Matthew P. Newlin. Model validation in the frequency domain. In *Proceedings of the IEEE Control and Decision Conference*, 1995.

[48] John C. Morris and Michiel van Nieuwstadt. Experimental platform for real-time control. In *Proceedings of the ASEE Annual Conference*, 1994.

[49] John C. Morris, Michiel van Nieuwstadt, and Pascale Bendotti. Identification and control of a model helicopter in hover. In *Proceedings of the American Control Conference*, 1994.

[50] Matthew P. Newlin. *Model Validation, Control, and Computation.* PhD thesis, California Institute of Technology, 1996.

[51] Matthew P. Newlin and John C. Morris, February 1995. Private communication.

[52] Matthew P. Newlin and Roy Smith. A generalization of the structured singular value and its application to model validation. *IEEE Transactions on Automatic Control*, 1996. To appear.

[53] Andy K. Packard. *What's new with $\mu$: Structured Uncertainty in Multivariable Control.* PhD thesis, University of California at Berkeley, 1988.

[54] Andy K. Packard. Gain-scheduling via linear fractional transformations. *Systems and Control Letters*, pages 79–92, 1994.

[55] Andy K. Packard and John C. Doyle. The complex structured singular value. *Automatica*, 29(1):71–109, 1993.

[56] Andy K. Packard, Michael K. H. Fan, and John C. Doyle. A power method for the structured singular value. In *Proceedings of the IEEE Control and Decision Conference*, pages 2132–2137, 1988.

[57] Andy K. Packard and Fen Wu. Control of linear fractional transformations. In *Proceedings of the IEEE Control and Decision Conference*, pages 1036–1041, 1993.

[58] Andy K. Packard, Kemin Zhou, Pradeep Pandey, and Greg Becker. A collection of robust control problems leading to LMIs. In *Proceedings of the IEEE Control and Decision Conference*, pages 1245–1250, 1991.

[59] Wilson J. Rugh. Analytical framework for gain scheduling. In *Proceedings of the American Control Conference*, pages 1688–1694, 1990.

[60] Michael G. Safanov and B. S. Chen. Multivariable stability-margin optimisation with decoupling and output regulation. *IEE Proceedings, Part D*, 129(6):276–282, November 1982.

[61] J. A. Schroeder, D. C. Watson, Mark B. Tischler, and M. M. Eshow. Identification and simulation evaluation of an AH-64 helicopter math model. In *AIAA Atmospheric Flight Mechanics Conference*, 1991.

[62] Jeff S. Shamma. Analysis of gain scheduled control for nonlinear plants. *IEEE Transactions on Automatic Control*, 35(12):898–907, 1990.

[63] Roy S. Smith and John C. Doyle. Model validation: A connection between robust control and identification. *IEEE Transactions on Automatic Control*, 37(7):942–952, July 1992.

[64] Torsten Söderström and Petre Stoica. *System Identification*. Prentice Hall, 1989.

[65] Spectrum Signal Processing. *DSP-LINK Prototyping Module: User's Manual*, November 1988. Version 1.3.

[66] Spectrum Signal Processing. *TMS320C30 System Board: User's Manual*, August 1990. Issue 1.01.

[67] Spectrum Signal Processing. *TMS320C30 System Board: Technical Reference Manual*, June 1991. Issue 1.02.

[68] Texas Instruments. *TMS320C30 User's Guide*, 1992. Revision SPRU031C.

[69] Mark B. Tischler. System identification requirements for high-bandwidth rotorcraft flight control systems. *Journal of Guidance, Control, and Dynamics*, 13(5):835–841, 1990.

[70] Michiel van Nieuwstadt and John C. Morris. Control of rotor speed for a model helicopter: a design cycle. In *Proceedings of the American Control Conference*, 1995.

[71] Joseph E. Wall, John C. Doyle, and Gunter Stein. Performance and robustness analysis for structured uncertainty. In *Proceedings of the IEEE Control and Decision Conference*, pages 629–636, 1982.

[72] Martin F. Weilenmann. *Robuste Mehrgrössen-Regelung eines Helikopters.* PhD thesis, Eidgenössische Technische Hochschule Zürich, 1994.

[73] Martin F. Weilenmann, Urs Christen, and Hans P. Geering. Robust helicopter position control at hover. In *Proceedings of the American Control Conference*, pages 2491–2495, 1994.

[74] Martin F. Weilenmann and Hans P. Geering. A test bench for rotorcraft hover control. In *AIAA Guidance, Navigation and Control Conference*, pages 1371–1382, 1993.

[75] Peter M. Young. *Robustness with parametric and dynamic uncertainty.* PhD thesis, California Institute of Technology, 1993.

[76] Peter M. Young, Matthew P. Newlin, and John C. Doyle. *Let's get real*, volume 66 of *IMA Volumes in Mathematics and its Applications*, pages 143–174. Springer-Verlag, 1994.

[77] George Zames. On the input-output stability of time-varying nonlinear feedback systems, Part I: Conditions derived using concepts of loop gain, conicity, and positivity. *IEEE Transactions on Automatic Control*, 11(4):228–238, April 1966.

[78] George Zames. On the input-output stability of time-varying nonlinear feedback systems, Part II: Conditions involving circles in the frequency plane and sector nonlinearities. *IEEE Transactions on Automatic Control*, 11(7):465–476, July 1966.

[79] Kemin Zhou, John C. Doyle, and Keith Glover. *Robust and Optimal Control.* Prentice Hall, 1995.