SAMPLED-DATA FREQUENCY RESPONSE
SYSTEM IDENTIFICATION FOR
LARGE SPACE STRUCTURES

A Thesis Presented to
the Faculty of the College of
Engineering and Technology
Ohio University

In Partial Fulfillment
of the Requirements of the Degree
Master of Science

by
Thomas T. Hammond
June, 1988
ACKNOWLEDGMENTS

First, I would like to extend my gratitude to the Faculty and Staff of the Department of Electrical and Computer Engineering. Their tireless efforts have aided me immensely in my graduate studies at Ohio University.

Special thanks is due to Dr. Jerrel Mitchell, my advisor, for suggesting the thesis topic, guiding my research, and editing the thesis drafts. I appreciate the input of my thesis committee, Dr. G.V.S. Raju, Dr. O.E. Adams, Jr., and Dr. R. Dennis Irwin, to the final draft of this thesis. Also, I would like to thank Bill Maggard for his cooperation in the experimental stage of this work.

Finally, I would like to express my appreciation to my family whose love and encouragement made this endeavor possible. My mother, Harriet, and late father, Claude, taught me the value of education and demonstrated the perseverance required to obtain it; and my fiancee, Judy, steadfastly supported my efforts with her love and patience.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGMENTS</td>
<td>i</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>iv</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>viii</td>
</tr>
<tr>
<td>Chapter 1. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1 Control of Large Space Structures</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Design-To-Performance</td>
<td>3</td>
</tr>
<tr>
<td>1.3 One-Controller-At-a-Time</td>
<td>4</td>
</tr>
<tr>
<td>1.4 Summary</td>
<td>8</td>
</tr>
<tr>
<td>Chapter 2. SYSTEM IDENTIFICATION</td>
<td>9</td>
</tr>
<tr>
<td>2.1 Overview</td>
<td>9</td>
</tr>
<tr>
<td>2.2 Sampled-data Systems</td>
<td>11</td>
</tr>
<tr>
<td>2.3 Fourier Transforms</td>
<td>14</td>
</tr>
<tr>
<td>2.4 Deterministic Analysis</td>
<td>18</td>
</tr>
<tr>
<td>2.5 Stochastic Analysis</td>
<td>22</td>
</tr>
<tr>
<td>2.6 Summary</td>
<td>23</td>
</tr>
<tr>
<td>Chapter 3. EXCITATION ENHANCEMENT</td>
<td>24</td>
</tr>
<tr>
<td>3.1 Signal Properties</td>
<td>24</td>
</tr>
<tr>
<td>3.2 Unit Pulse</td>
<td>26</td>
</tr>
<tr>
<td>3.3 Increasing Magnitude</td>
<td>30</td>
</tr>
<tr>
<td>3.4 Increasing Duration</td>
<td>32</td>
</tr>
<tr>
<td>3.5 Shaping Pulses</td>
<td>37</td>
</tr>
<tr>
<td>3.6 Summary</td>
<td>44</td>
</tr>
</tbody>
</table>
LIST OF FIGURES

1.1 Multiple-input, multiple-output system .............. 5
2.1 Typical sampled-data control system ................. 12
2.2 Sampled-data control system modeled with ideal
    sampler and zero-order-hold .......................... 12
2.3 Sampled-data system transfer function .............. 15
2.4 Sampled-data system frequency response function ... 15
2.5 System with noise modeled at output ................ 20
2.6 Unity negative feedback configuration ............... 20
3.1 z-plane mapping of frequencies up to the half-
    sampling frequency ..................................... 27
3.2 Unit digital pulse's characteristics ................ 28
3.3 Increased magnitude pulse's characteristics ...... 31
3.4 Increased duration pulse's characteristics ...... 36
3.5 Binomial triangle shaped pulse's characteristics .. 40
3.6 Bipolar triangle shaped signal's characteristics .. 43
3.7 Complementary binomial pair's spectral
    characteristic ........................................... 45
4.1 Three disk physical model ............................ 48
4.2 Three disk system sampled-data block diagram ...... 50
4.3 Three disk system sampled-data frequency response
    function .................................................... 55
4.4 Three disk system continuous root locus ............ 56
4.5 Three disk system continuous root locus of 0.1 Hz. mode .................................................. 57
4.6 Three disk system sampled-data root locus for unity gain .............................................. 59
4.7 Three disk system sampled-data frequency response block diagram .................................. 61
4.8 Frequency response function obtained using unit digital pulse ...................................... 64
4.9 Frequency response function obtained using increased magnitude pulse ...................... 65
4.10 Frequency response function obtained using increased duration pulse ......................... 66
4.11 Frequency response function obtained using binomial triangle shaped pulse .............. 68
4.12 Frequency response function obtained using bipolar triangle shaped pulse ................. 69
4.13 Frequency response function obtained using unity white noise ................................ 70
5.1 Hybrid experiment hardware configuration .......... 72
5.2 Analog computer simulation functional block diagram .................................................. 74
5.3 Spectral characteristics of unit digital pulse .... 79
5.4 Frequency response function using unit digital pulse .................................................. 80
5.5 Spectral characteristics of increased magnitude pulse .................................................. 81
5.6 Frequency response function using increased magnitude pulse ............................................. 82
5.7 Spectral characteristics of increased duration pulse .......................................................... 83
5.8 Frequency response function using increased duration pulse ............................................. 84
5.9 Spectral characteristics of binomial triangle shaped pulse ............................................... 85
5.10 Frequency response function using binomial triangle shaped pulse .................................. 86
5.11 Spectral characteristics of bipolar triangle shaped pulse ................................................... 88
5.12 Frequency response function using bipolar triangle shaped pulse ..................................... 89
5.13 Frequency response function using unity white noise ....................................................... 90
5.14 Frequency response function using single increased magnitude record ............................... 92
5.15 Frequency response function using four increased magnitude records ............................... 93
5.16 Frequency response function using sixteen increased magnitude records ............................ 94
5.17 Frequency response function using single binomial triangle record ..................................... 95
5.18 Frequency response function using four binomial triangle records ...................................... 96
5.19 Frequency response function using sixteen binomial
5.20 Frequency response function using single bipolar triangle record ........................................ 99

5.21 Frequency response function using four bipolar triangle records ........................................ 100

5.22 Frequency response function using sixteen bipolar triangle records .................................... 101

5.23 Frequency response function using sixteen spliced triangle records .................................... 102

5.24 Frequency response function using single unity white noise record ..................................... 103

5.25 Frequency response function using four unity white noise records ...................................... 104

5.26 Frequency response function using sixteen unity white noise records .................................. 105
LIST OF TABLES

Page

5.1 Analog computer potentiometer settings ............ 75
1.1 Control of Large Space Structures (LSSs)

As scientists and engineers attempt to utilize the unique properties of the Earth-orbit environment, it is clear that larger structures must be built in space to house these imaginative projects. Currently, there are several projects being planned that will require a physical structure many times larger than any previously placed in orbit. Two notable examples are the National Aeronautics and Space Administration's Space Station and the Department of Defense's Strategic Defense Initiative's Space-Based Lasers. It is the challenge of control systems engineers to develop effective methods for controlling the mechanical dynamics of these huge structures.

The construction of LSSs in orbit dictates the use of a minimal amount of light-weight materials. Typically, this implies the interconnection of long, thin, flexible beams to form the framework. This construction technique produces structures that are very flexible and exhibit very little mechanical damping. Hence, the unconstrained LSS will vibrate for an extremely long time when disturbed. Additionally, due to the complexity of the LSS, these vibrations, or flexible modes, will occur at several
frequencies with significant magnitudes. Effective suppression of these undesirable modes is essential for any LSS control scheme.

The process of assembling LSSs will span several months or years. During this time, the testing and use of modules already completed requires the implementation of structural control systems while the structure is evolving. The implication is that the dynamics of the LSS will change slowly over time. Similar effects will be produced by reorienting subsystems such as solar panels, docking with transportation vehicles, expanding and contracting due to thermal effects, etc. Therefore, the control scheme must be adaptable to these slowly changing dynamics.

Finally, much of the information on which the control system is based must be extracted from the structure itself. Ground-testing of the true structure is not possible, and even the investigation of scaled models is of questionable value due to gravitational effects. The method of finite-element analysis may yield some useful information, but models obtained in this fashion generally lack sufficient fidelity when applied to extremely high order systems. So the successful control scheme must be capable of incorporating empirical information gleaned from the true dynamics of the actual LSS.
1.2 Design-To-Performance (DTP)

A control system design strategy that has been tailored to meet the stringent requirements of LSS control is the Design-To-Performance procedure [1]. The key elements of DTP are the following:

- Determining LSS characteristics on-orbit.
- Designing an appropriate digital controller.

The design process entails gathering structural response data for the LSS; extracting a high-fidelity frequency domain system model from the data; specifying a high-performance control law based on the model; and implementing the control law with the digital controller onboard the LSS. This process may be repeated as necessary to fine-tune the design or to cope with changes in the dynamics of the LSS.

The DTP concept offers some important advantages in the LSS application. Other control methods often require sensors and actuators dedicated to gathering data to identify the dynamics of the LSS. The DTP method uses only those actuators and sensors necessary to control the LSS. Another attractive feature is the use of frequency domain models. Methods that require state variable models are faced with the formidable task of developing these models for a system that may be of order 100 or more. Furthermore,
design procedures for frequency domain models can be used which produce much simpler control laws than state-space methods. A design procedure that is particularly well-suited for the DTP philosophy is considered next.

1.3 One-Controller-At-a-Time (1-CAT)

One-Controller-At-a-Time is a design procedure for multiple-input, multiple-output (MIMO) feedback control systems [2]. A linear MIMO system may be viewed as an array of single-input, single-output (SISO) systems. For a MIMO system with \( n \) inputs and \( m \) outputs, there are \( m \times n \) SISO systems in the model. Each MIMO output is formed by summing the outputs of \( n \) SISO systems, each with a different MIMO system input. The input-output relationships of this model are conveniently represented with a vector equation using an \( m \times n \) coefficient matrix of SISO system transfer functions. This representation of a MIMO system is illustrated in Figure 1.1, where
Figure 1.1. Multiple-input, multiple-output system.
The frequency domain description of this system is formed by replacing the transfer functions with the corresponding frequency response functions. In practice, these may be obtained by exciting the system inputs one at a time, measuring each output due to the current input, and computing the frequency response function from the current input to each output.

Now, the addition of control loops to this system will couple the loops: the control loops cannot be designed independently. However, they can be designed sequentially, or one-at-a-time. This is the underlying theme of I-CAT. The procedure is to close one loop, to evaluate the effects of this closure on the next loop to be closed, and to close the next loop based on the effects of closing the previous loop. This process is repeated until all required loops are closed. With prudent selection of the sequence of loop closures, high-performance control may be achieved with this
step-by-step method. Used with frequency domain design techniques, I-CAT produces low-order compensators for high-order systems.

Another design enhancement that is compatible with the methods outlined above is that of Modal Suppression by Phase Stabilization (MSPS) [1]. This technique is especially attractive for suppressing low-frequency flexible modes encountered in LSS control. "A mode is said to be phase stabilized if its gain, in the compensated open loop frequency response, is greater than 1 (0 dB) and no encirclement of the -1+j0 point on the polar frequency response occurs due to this mode. A mode is perfectly phase stabilized if the phase of the loop frequency response is 0 degrees at its peak frequency" [1]. Modes which are phase stabilized are stabilized throughout the structure: MSPS is a global dynamic effect, not merely a local loop phenomenon.

Phase stabilization occurs inherently in certain hardware configurations. In particular, the collocation of a force or torque actuator with the corresponding rate sensor forms a "complementary pair" that produces modal phase stabilization [1]. Thus, neglecting actuator and sensor dynamics, modes excited and measured in this pair's loop will be phase stabilized by closure of the loop; and,
as noted above, this stabilization is effective throughout the structure. Obviously, this is an extremely nice property to exploit whenever it is possible.

1.4 Summary

As detailed in this introductory chapter, the DTP strategy encompasses all the salient features desirable in LSS control. However, its effectiveness is rooted in the fidelity of the empirical frequency domain model elicited from the LSS. It is the purpose of this thesis to examine some techniques of enhancing the sampled-data frequency response functions that are characteristic of LSS dynamics.

First, in Chapters Two and Three, the theoretical basis of the experimental methods used later is examined. Next, an experiment is designed and implemented to explore the practical application of these methods. Finally, the experimental findings are analyzed, and suggestions are offered for the application and further investigation of these methods.
CHAPTER 2.
SYSTEM IDENTIFICATION

2.1 Overview

The design of control systems requires sufficient knowledge of the mathematical relationships that describe the physical system that is to be controlled. Before a system is constructed, attempts may be made at predicting or "modeling" the system's behavior. For simple systems, the mathematical model may be constructed by applying physical principles known to govern the system's behavior. More complex systems may be viewed as a combination of simpler subsystems, and the same techniques applied. However, as the order of the system grows, its behavior becomes more difficult to satisfactorily predict in this manner.

After a system has been constructed, the analysis of its input and output signals may reveal a wealth of information concerning the dynamics of the system. "System identification" is the process of deriving a mathematical model to describe the system's dynamic behavior from observations of its input and output signals. As one would expect, incorporating the system modeling information into the system identification process often yields a more suitable system model.
There are many questions to be answered before a suitable system identification method can be prescribed. What is to be the mathematical structure of the model (e.g. differential or difference equations, transfer functions, frequency response functions, etc.)? Is the system linear or nonlinear; stationary or nonstationary; discrete, sampled-data, or continuous; single-input or multiple-input; etc.? What input signals are desirable and permissible to be employed to excite the system? Proper consideration must be given to all these questions and more, in order to successfully identify a high-fidelity system model.

For the LSS control design procedures outlined in the introduction, sampled-data frequency response functions are required. It is assumed that the LSS has linear characteristics, and excitation signals will be limited to levels that will not evoke nonlinear behavior of the actuators, sensors, or structure. Also, the system is assumed to be stationary, at least during the identification process. What input signals are desirable? This is the question to be addressed in the sequel. First though, a closer examination of sampled-data frequency response system identification is necessary.
2.2 Sampled-data Systems

Historically, frequency response methods of system identification have enjoyed much success. For continuous systems, the general approach is to excite the system with an input signal whose spectral characteristics are known. The system's output signal due to this excitation is observed, and its spectral characteristics are determined. Now, the system's frequency response function may be estimated from the input and output spectra. The same general procedure may be extended to sampled-data systems.

A typical SISO sampled-data control system is shown in Figure 2.1. The control loop is composed of the continuous system that is to be controlled, an analog-to-digital (A/D) converter, the digital computer/controller, and a digital-to-analog (D/A) converter. The A/D converter samples the system's continuous output at regular intervals of \( T \) seconds and provides the digital computer with a number representative of the system's output at the last sampling time. This sampling function may be represented diagrammatically by an ideal sampler whose input is the system's output, and whose output is equal to the system's output, but only exists at the sampling instants. The D/A converter accepts a number from the computer that represents the input signal to be applied to the system and applies the specified
Figure 2.1. Typical sampled-data control system.

Figure 2.2. Sampled-data control system modeled with ideal sampler and zero-order-hold.
input signal to the system until the next number in the sequence arrives. Usually, D/A converters are used that apply a constant input signal between numbers received from the computer. This manner of constructing a continuous signal from a digital sequence is described by the zero-order-hold (ZOH) function. An equivalent representation of the sampled-data system is shown in Figure 2.2.

Now, the system to be identified is the sampled-data system from the input of the ZOH to the output of the sampler. So the input signal is \( u(kT) \), and the output signal is \( v(kT) \), where \( k \) is the sample index. The relationship between the input and output signals of this model may be described by the discrete-time system transfer function. A sampled-data signal is transformed to the discrete domain by using the z-transform. The z-transform is defined as

\[
F(z) = \sum_{k=0}^{\infty} f(kT)z^{-k},
\]

where the relationship between the z-transform complex variable \( z \) and the Laplace transform complex variable \( s \) is

\[
z = e^{sT}.
\]

So the transfer function of the system may be expressed in terms of the ratio of the input and output signals in the z-domain as
Thus, the sampled-data system to be identified may be represented as shown in Figure 2.3.

Finally, the relationship between the complex variable \( z \) and the real frequency \( \omega \) is

\[
    z = e^{j\omega T}.
\]

Then, the sampled-data frequency response function for the system is found to be

\[
    H(e^{j\omega T}) = \frac{V(e^{j\omega T})}{U(e^{j\omega T})}.
\]

This relationship is depicted in Figure 2.4.

The direct computation of the input and output spectra from empirical data is commonly accomplished using the Discrete Fourier Transform (DFT) and its efficient implementation with the Fast Fourier Transform (FFT). These topics are reviewed in the next section.

2.3 Fourier Transforms

The spectra for signals in continuous control systems are determined by applying the Fourier Transform. Similarly, the spectra of sampled signals may be obtained by
Figure 2.3. Sampled-data system transfer function.

Figure 2.4. Sampled-data system frequency response function.
applying the Discrete Fourier Transform; and since signals are sampled for a finite time period, or for a finite number of samples, the finite DFT is used. The finite DFT is defined as

\[ F(e^{j\omega T}) = \sum_{k=0}^{N-1} f(kT)e^{-j\omega kT}. \] (2.6)

Note that the spectra of sampled signals are periodic with a period equal to the sampling frequency of

\[ \omega_s = \frac{2\pi}{T}. \] (2.7)

In order to implement DFTs efficiently, computer algorithms termed Fast Fourier Transforms have been developed. The effectiveness of FFTs has been a real boon to digital spectral analysis. However, there are two problems which arise in deriving the spectra of sampled-data signals with FFTs. These are 1) aliasing and 2) leakage. Both phenomena must be considered when computing sampled-data frequency spectra and are explained next.

Aliasing occurs when the sampled signal contains frequency components that are higher than one-half the sampling frequency. Instead of appearing in the spectrum at the proper frequencies, these components are "folded back" about the one-half sample frequency and added to the
spectral components of these lower frequencies. The obvious solution is to sample the signal at a frequency at least twice that of the highest component frequency. However, the spectral content of the signal is not always well-known, and the noise present in the signal is likely to have frequency content that can cause aliasing. Thus, choosing a conservatively high sampling frequency is advisable, if one has this option.

Leakage occurs if the sampled signal is aperiodic, or if the sampling window is not precisely an integral multiple of the signal's period. The finite nature of the sampled signal may be viewed as multiplying an infinitely long signal with a unity-magnitude rectangular window of the same length as the sampled signal. The frequency domain representation of this rectangular window contains significant "side-lobes" over frequencies adjacent to the desired frequency. Therefore, the resultant spectrum obtained by combining these signals in the frequency domain has coarser resolution than would be expected, due to the spreading of spectral information over adjacent frequencies. The usual precaution for leakage is to use a window that is shaped to make the time signal appear to be periodic in the window: this suppresses the side-lobes in the frequency domain.
Typical time windows taper the signal to zero at each end of the time window. One window particularly popular for system identification is the Hanning window, which is

\[ h(t) = 0.5 - 0.5 \cos \frac{2\pi t}{T_r}, \text{ for } 0 \leq t \leq T_r, \text{ and } \]

\[ h(t) = 0, \text{ elsewhere,} \]

where \( T_r \) is the sampled record duration.

NOTE: For the remainder of this thesis in the interest of notational simplification, \( (f) \) will be used to represent the dependence of a signal's representation or of a function on the complex variable \( \exp(j\omega T) \) with the radian frequency \( \omega \) replaced by the equivalent expression \( 2\pi f \). This simplification is summarized as

\[ H(f) = H(e^{j2\pi f T}) = H(e^{j\omega T}). \]

2.4 Deterministic Analysis

Input signals used for exciting systems during the identification process may be divided into the classes of deterministic signals and stochastic, or random, signals. The analysis of systems differs for these two classes of signals, and each is considered separately in this and the succeeding section.
For the ideal sampled-data system outlined previously, the frequency response function is simply

\[ H(f) = \frac{V(f)}{U(f)}. \]  

(2.10)

However, real systems must be modeled by adding a noise signal to the measured output signal. This is necessary because real systems and measurements differ somewhat from the assumed ideal systems. The real system may have slight nonlinearities, a small degree of nonstationarity, minor disturbances, inherent sensor noise, etc. All variations due to these nonideal characteristics are modeled as a noise signal spectrum \( N(f) \) added to the ideal output signal spectrum \( V(f) \) to form the measured output signal spectrum \( Y(f) \). The system block diagram now is illustrated by Figure 2.5, where

\[ Y(f) = V(f) + N(f). \]  

(2.11)

Now, for a deterministic input signal, an estimated frequency response function \( \hat{R}(f) \), may be calculated from the input signal and the measured output signal by noting that

\[ \hat{R}(f) = \frac{Y(f)}{U(f)}. \]  

(2.12)
Figure 2.5. System with noise modeled at output.

Figure 2.6. Unity negative feedback configuration.
\[ \hat{R}(f) = \frac{V(f) + N(f)}{U(f)}, \text{ and} \] (2.13)

\[ \hat{R}(f) = H(f) + \frac{N(f)}{U(f)}. \] (2.14)

Note the important fact that the fidelity of the estimate is improved by increasing the relative amount of the ideal output signal in the measured output signal, or in other words, by increasing the signal-to-noise ratio at the system's output. This observation is the basis of enhancing frequency response functions by prudent selection of deterministic input signals as detailed in Chapter Three. Additionally, if the noise may be characterized as zero-mean and ergodic, then the random error of an averaged magnitude estimate, obtained by averaging \( n \) estimates, is reduced by a factor of \( 1/\sqrt{n} \) \[3\]. This fact offers another tool for empirical frequency response function enhancement, and its practical effectiveness is demonstrated in the experimental section of this thesis.

A final system configuration that needs to be examined is that of identifying a frequency response function that is embedded in a feedback loop. For simplicity, unity negative feedback is assumed as depicted in Figure 2.6.
With knowledge of the input spectrum $X(f)$ and the measured output spectrum $Y(f)$, the control error signal spectrum $U(f)$ is

$$U(f) = X(f) - Y(f).$$ \hfill (2.15)

In view of this relationship, the frequency response function may be estimated using the formula developed for the open-loop configuration, i.e. Equation 2.12.

2.5 **Stochastic Analysis**

The empirical analysis of systems excited by stochastic input signals is considered in this section. Often, the input signal is actually pseudo-random, but its statistics are exemplary of a stochastic signal. The spectral input-output relationships of this analysis are in terms of one-sided spectral density functions. In particular, for the open-loop system with output noise which is uncorrelated with the input signal, the optimal frequency response function estimate, in the least squares sense, is found to be

$$\hat{H}(f) = \frac{\hat{G}_{wy}(f)}{\hat{G}_{uu}(f)},$$ \hfill (2.16)

where $\hat{G}_{wy}(f)$ is the input-output cross-spectral density function estimate, and $\hat{G}_{wu}(f)$ is the input auto-spectral
density function estimate [4]. In this case, the random error of the magnitude estimate is reduced by a factor of $1/\sqrt{2n_d}$ through averaging $n_d$ independent data records [3].

For the unity negative feedback configuration, the following relationship yields the frequency response function estimate of the open-loop transfer function [3]

$$\hat{H}(f) = \frac{\hat{G}_{xy}(f)}{\hat{G}_{xx}(f) - \hat{G}_{xy}(f)}.$$ \hspace{1cm} (2.17)

2.6 **Summary**

In this chapter, the theoretical foundations for the methods of analysis used in the remainder of this thesis have been reviewed. The process of system identification by sampled-data frequency response function determination was outlined, and the differences encountered in deterministic and stochastic excitation methods were considered. In the ensuing chapter, the spectral properties of several deterministic signals are investigated to develop a rationale for improving frequency response function estimates using these signals.
CHAPTER 3.
EXCITATION ENHANCEMENT

3.1 Signal Properties

In this chapter some properties which enhance a deterministic signal's efficacy as the excitation signal for frequency response system identification are considered. Then some signals are examined that offer the opportunity of tailoring the choice of input excitation to meet constraints of the specific system being identified. Also, comments are included concerning the applicability of each signal.

Recall from the development of deterministic analysis in Section 2.4 that

\[ H(f) = Y(f) + N(f) \]  
\[ U(f) \]

This leads to the important results that

\[ \hat{A}(f) = \frac{Y(f)}{U(f)} \text{, and} \]  
\[ Y(f) = V(f) + N(f) \]  
\[ \hat{A}(f) = \frac{V(f) + N(f)}{U(f)} \text{, and} \]

\[ \hat{A}(f) = H(f) + \frac{N(f)}{U(f)} \text{, and} \]
Equation 3.4 clearly characterizes the effect of the input signal spectrum on the estimated frequency response function magnitude: increasing the magnitude of the input spectrum over frequencies of interest improves the estimate $\hat{H}(f)$.

The signals considered for use in the sequel are finite duration sequences that are non-zero for only a portion of the identification time window. The start of the identification window is coincident with the first non-zero element of the sequence, and the input sequence will be permitted to assume non-zero values for $M$ elements, or sample periods. Thus, the signals of interest may be expressed by their $z$-transforms as

$$U(z) = \sum_{k=0}^{M-1} u(kT)z^{-k}.$$  

Furthermore, by invoking the relationship $z = \exp(j\omega T)$ the spectrum of the signal is shown to be

$$U(e^{j\omega T}) = \sum_{k=0}^{M-1} u(kT)e^{-j\omega kT}.$$  

Note that Equation 3.6 is the finite DFT of the signal as introduced in Section 2.3. The foregoing relationships will be used to investigate the signal sequences proposed in this chapter.
A closer look at the relationship $z = \exp(j\omega T)$ will provide some needed insight into the signals presented. In the $z$-plane this relationship describes a unit circle centered at $z = 0$. Real frequencies of interest are restricted to the range of zero to the half-sampling rate, or $0 \leq \omega \leq \omega_s/2$. This describes the half of the unit circle that begins at $z = 1$, or $\omega = 0$, and proceeds in the counter-clockwise direction to $z = -1$, or $\omega = \omega_s/2$. This relationship helps to reconcile the two signal representations above, and it is graphically depicted in Figure 3.1.

3.2 Unit Pulse

An obvious choice for launching this signal investigation is the unit digital pulse. The unit digital pulse is defined by

$$u(kT) = \begin{cases} 1, & \text{for } k = 0, \text{ and} \\ 0, & \text{otherwise.} \end{cases}$$

From Equation 3.6, the spectrum of the unit digital pulse is found to be

$$U(e^{j\omega T}) = u(0) = 1.$$  

The unit digital pulse's characteristics are graphically displayed in Figure 3.2. A shorthand notation will occasionally be used in the sequel to denote the peak
Figure 3.1. z-plane mapping of $0 \leq \omega \leq \omega_s / 2$. 
Figure 3.2. Unit digital pulse's characteristics.
magnitude and duration in sampling periods of an input signal. The form of this notation is (peak magnitude x duration), which for the unit pulse is (1x1) as shown in Figure 3.2.

The most convenient result of Equation 3.8 coupled with Equation 3.1 shows that the frequency response function estimate is simply the measured output spectrum

\[ \hat{R}(f) = Y(f). \]  

(3.9)

Two advantages of unit digital pulse excitation are the simplicity of computation and uniformity of magnitude across the spectrum of interest. Additionally, this signal and estimated frequency response function provide convenient references for comparison with other excitation methods.

The unity amplitude of this pulse is assumed to be sufficiently small so that no structural response constraints are violated, e.g. actuator saturation, non-linear structural behavior, etc. However, the unity spectral magnitude does not enhance the estimated frequency response function by suppressing the noise component as suggested by Equation 3.4.
3.3 Increasing Magnitude

Suppose the magnitude of an input signal is increased by the factor \( a \). That is

\[ u_a(kT) = au(kT). \]  \hspace{1cm} (3.10)

Since the DFT is a linear transformation, the effect of this factor on the signal's spectrum is

\[ U_a(f) = aU(f). \]  \hspace{1cm} (3.11)

So application of Equation 3.4 yields

\[ \hat{H}(f) = H(f) + \frac{N(f)}{aU(f)}. \]  \hspace{1cm} (3.12)

In the particular case of \( u(kT) \) being the unit digital pulse

\[ \hat{H}(f) = H(f) + \frac{N(f)}{a}. \]  \hspace{1cm} (3.13)

This equation shows the effect of increasing the amplitude of the unit pulse excitation by a factor of \( a \). The resulting input signal is noted in the shorthand introduced as \((a \times 1)\), and its characteristics are shown in Figure 3.3 for the signal \((10 \times 1)\). Clearly, the estimated frequency response function is improved by increasing the magnitude of the input.
Figure 3.3. Increased magnitude pulse's characteristics.
This section demonstrates the possibility of enhancing the system identification results by increasing the excitation magnitude. However, as alluded to in the previous section, there are limits on the signal's magnitude; and hence, there are limits on the improvement possible in this manner. On the other hand, the improvement is still uniform across the spectrum, which is quite desirable.

3.4 Increasing Duration

In this section the effects of increasing the duration of a constant amplitude pulse are considered. A constant amplitude pulse that extends over $M$ sample periods has the z-transform

$$U_M(z) = \sum_{k=0}^{M-1} a z^{-k}. \quad (3.14)$$

The expansion of the summation yields

$$U_M(z) = a[1 + z^{-1} + \ldots + z^{-(M-2)} + z^{-(M-1)}]. \quad (3.15)$$

Now multiplying the summation by the unity factor of $z^{-1} / z^{M-1}$ gives

$$U_M(z) = a \left[ \frac{z^{M-1} + z^{M-2} + \ldots + z + 1}{z^{M-1}} \right]. \quad (3.16)$$
The effect of extending the duration of this pulse at low frequencies may be shown by evaluating Equation 3.16 for \( z=1 \), which corresponds to \( \omega = 0 \). This indicates that \( U_M(1) = aM \), so the low-frequency magnitude spectrum is increased by a factor equal to the number of sampling periods over which the pulse extends. Hence, increasing the pulse duration increases the input signal's magnitude spectrum in this region and enhances the estimated frequency response function in the manner explained previously.

Further consideration of Equation 3.16 is necessary to determine the effects at higher frequencies. This representation of the signal points out that nulls may occur in the signal's spectrum if roots of the numerator lie on the unit semi-circle of interest. Note that the roots of the denominator are all at \( z=0 \), and these do not affect the real frequency spectrum. The roots of the numerator may be investigated by simplifying the expression by multiplying it by \((z-1)\) as suggested in [5], which adds a root at \( z=1 \) that may be discarded later. This analysis proceeds as follows:

\[
(z-1)(z^{M-1} + z^{M-2} + \ldots + z + 1) = 0, \tag{3.17}
\]

\[
z^{M-1} = 0, \quad \text{and} \tag{3.18}
\]

\[
z^M = 1. \tag{3.19}
\]
Now, the frequencies of the spectral nulls may be determined by substituting the expression $z = \exp(j\omega T)$ and solving for frequencies that are in the range $0 \leq \omega \leq \omega_s/2$ as

$$\left(e^{j\omega T}\right)^M = 1,$$  \hspace{1cm} \text{(3.20)}

or

$$e^{j\omega MT} = 1.$$  \hspace{1cm} \text{(3.21)}

Equation 3.21 is satisfied for all $\omega$ that satisfy

$$\omega MT = 2n\pi, \text{ } n \text{ an integer, or}$$  \hspace{1cm} \text{(3.22)}

$$\omega = \frac{2n\pi}{MT}.$$  \hspace{1cm} \text{(3.23)}

By substitution of the relationship $T = \frac{2\pi}{\omega_s}$, the equation becomes

$$\omega = \frac{n\omega_s}{M}.$$  \hspace{1cm} \text{(3.24)}

Application of the constraints on $\omega$ leads to constraints on $n$ of

$$\omega \geq 0 \Rightarrow n \geq 0, \text{ and}$$  \hspace{1cm} \text{(3.25)}

$$\omega \leq \frac{\omega_s}{2} \Rightarrow n \leq \frac{M}{2}.$$  \hspace{1cm} \text{(3.26)}
Note that \( n=0 \Rightarrow \omega=0 \Rightarrow z=1 \) and corresponds to the root added earlier and discarded now. Therefore, the admissible numerator roots correspond to the values of \( n \)

\[
\begin{align*}
n &= 1, 2, \ldots, \frac{M}{2}, \text{ for } M \text{ even, and} \\
\end{align*}
\]

\[
\begin{align*}
n &= 1, 2, \ldots, \frac{M-1}{2}, \text{ for } M \text{ odd.}
\end{align*}
\]

The values of \( n \) listed in Equation 3.27 in conjunction with Equation 3.24 identify frequencies at which magnitude nulls appear in the signal's spectrum. These nulls prohibit identification of the frequency response function at these frequencies. However, with proper precautions, signals of constant amplitude and extending over more than one sampling period may be used in certain circumstances. An example of this type of signal is illustrated in Figure 3.4 for a pulse of (1x10).

First, observe that the lowest frequency at which a null occurs is for \( n=1 \), or \( \omega=\omega/M \). So for frequencies sufficiently below this null, the increased duration pulse may be used successfully. Also, note that for signals of length \( M \) and \( M+1 \), the nulls are distinct from each other. This suggests that using this pair of signals and averaging the results may successfully identify modes that would otherwise be masked by the nulls of one signal or the other [5]. However, for pulses extending over several sampling
Increased Duration Pulse (1x10)

Sequence

![Graph showing the sequence of increased duration pulse]

Sample Index k

Increased Duration Pulse (1x10)

Spectrum

![Graph showing the spectrum of increased duration pulse]

Figure 3.4. Increased duration pulse's characteristics.
periods, the nulls cause much difficulty as the frequency approaches the half-sampling rate. Of course, the same considerations of structural limitations must be considered as were in the increased magnitude case of Section 3.3. Finally, note that a combination of increasing magnitude and duration may better fit specific system constraints.

3.5 Shaping Pulses

The drawbacks associated with the nulls encountered in the previous section may be overcome to a large extent by shaping the input signal. The strategy is to avoid nulls below the half-sampling rate. Consider a signal with the z-transform representation [5]

\[
U_M(z) = \frac{(z+1)^{M-1}}{z^{M-1}}.
\]  

(3.28)

Examination of Equation 3.28 shows that nulls on the semi-circle of interest occur only at \( z = -1 \), or \( \omega = \omega_s / 2 \), which is the half-sampling rate. This will avoid much of the problem associated with using a protracted constant amplitude pulse. Further analysis of Equation 3.28 will reveal the input sequence that has this z-transform.

First, expansion of the numerator and expression of the fraction as a series yield the following:
\[ U_M(z) = \frac{z^{M-1} + b_{M-2}z^{M-2} + \ldots + b_1 z + 1}{z^{M-1}} \], and

\[ U_M(z) = 1 + b_{M-2}z^{-1} + \ldots + b_1 z^{-(M-2)} + z^{-(M-1)}. \] (3.30)

So the sequence desired is shown in the expression of the z-transform

\[ U_M(z) = Z\{1, b_{M-2}, \ldots, b_1, 1\}. \] (3.31)

Note that this sequence is the ordered coefficients of the binomial expansion of \((z-1)\) using \((M-1)\) factors in the expansion. Therefore, this signal will be referred to as a binomial weighted triangular pulse, or simply a binomial triangle pulse.

The increased spectral magnitude at low frequencies offered by this signal may be ascertained from Equation 3.28 evaluated at \(z=1\), or \(\omega=0\), which gives

\[ U_M(1) = 2^{M-1}. \] (3.32)

This result indicates that each sampling period extension of this set of signals yields a doubling of the low-frequency magnitude. Thus the magnitude increases exponentially with increments in \(M\), in contrast to linear increments in the case of extending a constant amplitude pulse. However, due to the increasing peak magnitude of the binomial triangle,
these effects may be restricted to a few sampling periods. If it is desirable to use the binomial excitation over a longer time, the magnitude of the signal may be scaled so that system constraints are observed. Of course, this will reduce the spectral magnitude as well, but the nulls are still located at the half-sampling rate. This method provides the designer with some freedom in tailoring the excitation to the system at hand. A binomial triangle pulse with peak magnitude normalized to unity and spanning ten sampling periods (1x10) is shown in Figure 3.5.

The problem with spectral nulls was diminished considerably by the introduction of the binomial triangle signal. However, the null at the half-sampling rate may still mask system modes at frequencies in this region. This leads to the consideration of a signal which is complementary to the binomial triangle signal [6]. Note that the nulls of the following expression all occur at \( z=1 \), or \( \omega = 0 \),

\[
U_M(z) = \frac{(z-1)^M}{z^{M-1}}.
\]

(3.33)

The sequence which has this \( z \)-transform may be identified in a like manner to the binomial triangle analysis as
Figure 3.5. Binomial triangle shaped pulse's characteristics.
\[ U_M(z) = z^{M-1} + (-1)b_{M-2}z^{M-2} + ... \]  
\[ + (-1)^{M-2}b_1z + (-1)^{M-1}, \]  
(3.34)

\[ U_M(z) = 1 + (-1)b_{M-2}z^{-1} + ... \]  
\[ + (-1)^{M-2}b_1z^{-(M-2)} + (-1)^{M-1}z^{-(M-1)}, \]  
(3.35)

\[ U_M(z) = Z\{1,(-1)b_{M-2},...,-(-1)^{M-2}b_1,(-1)^{M-1}\}. \]  
(3.36)

So in this case, the sequence is seen to be the ordered coefficients of the binomial expansion of \((z-1)\) using \((M-1)\) factors in the expansion. Note that the elements of this sequence alternate signs. Therefore, this signal will be referred to as a bipolar binomial weighted triangular signal, or simply a bipolar triangle. Additionally, the cryptic notation used will include a minus sign with the peak magnitude to distinguish it from the binomial triangle, i.e. \((-1\times 10)\) indicates a bipolar triangle with peak magnitude normalized to unity and spanning ten sample periods.

The increase in spectral magnitude offered at the half-sampling rate is determined by evaluating Equation 3.33 at \(z=-1\), or \(\omega = \omega_s/2\), which yields

\[ U_M(-1) = 2^{M-1}. \]  
(3.37)

This result verifies that the bipolar triangle is complementary to the binomial triangle. This suggests using
these signals as a pair to span a wider range of frequencies if necessary. A rationale for combining the results obtained with these two signals is developed next. A bipolar triangle signal is presented in Figure 3.6 that corresponds to the binomial triangle signal in Figure 3.5.

Since the spectral magnitude of the binomial signal decreases as frequency increases, and the spectral magnitude of the bipolar signal increases as frequency increases, it is reasonable to find the frequency at which these spectra are equal, and to use the binomial results for lower frequencies and the bipolar results for higher frequencies. The boundary frequency may be found by identifying the frequency on the semi-circle in the z-plane at which the z-transforms of the two signals have the same magnitude. By equating the magnitudes of Equations 3.28 and 3.33 the expression obtained is

\[
|(z-1)^M| = |(z+1)^M|.
\]  

(3.38)

In terms of frequency, this expression becomes

\[
\left|e^{j\omega T} - 1\right|^M = \left|e^{j\omega T} + 1\right|^M.
\]  

(3.39)

The solution of Equation 3.39 for the range of admissible frequencies yields the boundary frequency of
Bipolar Triangle Shaped Pulse (-1x10) Sequence

![Bipolar Triangle Shaped Pulse Sequence Graph]

Bipolar Triangle Shaped Pulse (-1x10) Spectrum

![Bipolar Triangle Shaped Pulse Spectrum Graph]

**Figure 3.6.** Bipolar triangle shaped signal's characteristics.
Thus, for a given sampling frequency, the system may be identified by using the binomial excitation signal results for frequencies below the quarter-sampling frequency, and using the bipolar excitation signal results for frequencies between the quarter-sampling frequency and the half-sampling frequency. Of course, this doubles the identification effort, but it is a method for consideration in particular systems. The effective excitation spectrum using this strategy is obtained by splicing the appropriate spectral sections of Figures 3.5 and 3.6, and the resulting spectrum is shown in Figure 3.7.

3.6 Summary

In this chapter several methods of altering deterministic signals to enhance their effectiveness in system identification applications have been considered. These methods include altering a signal's magnitude, duration, shape, and using complementary signal pairs. The method or combination of methods that will provide the best results depends primarily upon the system constraints and the frequency range of interest. The practical aspects of using these methods are highlighted in the experiment which follows.
**Shaped Complementary Pulses (1x10)**

**Spliced Spectra**

![Graph showing spliced spectra with magnitude in dB on the y-axis and fraction of sampling frequency on the x-axis.

Figure 3.7. Complementary binomial pair's spectral characteristic.
CHAPTER 4.
EXPERIMENT DESIGN

4.1 Objective

A prime objective of this thesis is the experimental investigation of the effectiveness of the methods of estimated frequency response function enhancement outlined in the previous chapter. Since the identification of LSS systems is the ultimate goal, the experimental model should display LSS characteristics, i.e. lightly damped flexible modes. In addition, some of the concepts associated with DTP, 1-CAT, and MSPS, which were introduced in Chapter One, will be demonstrated in the hypothetical system analysis. For example, a simple control loop will be employed to stabilize the system to permit application of an excitation signal, which is a fundamental step in the DTP procedure. This control loop will incorporate the property of MSPS in order to ensure global system stability. Furthermore, the system selected for the experiment may be viewed as one of the single-input, multiple-output subsystems in Figure 1.1. Thus, the estimated frequency response function is exemplary of an element in the matrix in Equation 1.1, which is the foundation of the 1-CAT methodology.
In order to add credence to the simulation, the continuous structural dynamics will be modeled on an analog computer, and the stabilizing compensation will be provided by a digital controller to form a hybrid simulation. The use of the analog computer introduces limits on system dynamics that may be viewed as structural constraints; and these must be carefully observed, as the admonishments of Chapter Three indicate. In particular, excitation signals must be scaled so that the system behavior remains linear, i.e. that the analog computer amplifiers do not saturate. So it is clear that the ensuing experiment embodies many of the practicalities encountered in an actual system identification endeavor.

4.2 Physical Model

As an aid in visualizing the physical concepts discussed, a hypothetical structure is adopted, which is the same configuration as that used by Maggard [7]. This structure consists of three rigid disks joined at their centers by two massless rods as depicted in Figure 4.1. The motion of this structure is restricted to rotation about the axis through the centers of the disks. However, the two rods exhibit torsional spring constants of $k_1$ and $k_2$ about this axis, so that the system dynamics include a rigid-body mode and two flexible modes. In the interest of
Figure 4.1. Three disk physical model.
computational simplification, the mass polar moments of inertia of the disks are all deemed to be unity. Thus, the specification of the two torsional spring constants will determine the frequencies of the two flexible modes. Also, since LSS modes are very lightly damped, no damping is modeled in the structure.

A torque actuator and a rotational velocity sensor are collocated on the first disk. This complementary pair will be used to form the MSPS control loop to stabilize the structure during the identification process. Additionally, a rotational velocity sensor is placed on the third disk, and the sampled-data frequency response function to be identified is from the digital torque input to the Disk-3 sampled velocity. A block diagram of the sampled-data system is presented in Figure 4.2.

4.3 Mathematical Models

The mathematical models presented in this section are in the form of transfer functions. Since the goal is to implement these transfer functions efficiently on an analog computer, they are stated in forms which facilitate programming the analog computer. The transfer functions derived here may be transformed to differential equations that lead to simple analog computer implementations.
Figure 4.2. Three disk system sampled-data block diagram.
Application of the laws governing rotational motion yields the differential equations describing the behavior of the structure. Manipulation of these equations and Laplace transformation leads to the pertinent transfer functions

\[
\frac{\dot{\theta}_3(s)}{T(s)} = \frac{k_1 k_2}{s \left[ s^4 + 2(k_1 + k_2)s^2 + 3k_1 k_2 \right]}, \quad \text{and} \tag{4.1}
\]

\[
\frac{\dot{\theta}_1(s)}{T(s)} = \frac{s^4 + (k_1 + 2k_2)s^2 + k_1 k_2}{s \left[ s^4 + 2(k_1 + k_2)s^2 + 3k_1 k_2 \right]} \tag{4.2}
\]

The determination of the flexible modes' frequencies is now considered. An analysis of the dynamics of a version of the proposed Space Station indicates that significant flexible modes will span the frequency range of 0.1 to 1.5 Hz. [1]. So these two extreme frequencies are chosen to be the flexible modes of this model. These frequencies are characteristic of the physical model adopted when \( k_1 = 0.2636 \) and \( k_2 = 44.35 \text{ N-m/rad} \). Finally, the expression of the transfer functions in terms of the modal components yields

\[
\frac{\dot{\theta}_3(s)}{T(s)} = \frac{0.3333}{s} - \frac{0.3348s}{s^2 + 0.3948} + \frac{1.488 \times 10^{-3} s}{s^2 + 88.83}, \quad \text{and} \tag{4.3}
\]

\[
\frac{\dot{\theta}_1(s)}{T(s)} = \frac{0.3333}{s} + \frac{0.6667s}{s^2 + 0.3948} - \frac{5.194 \times 10^{-6} s}{s^2 + 88.83}. \tag{4.4}
\]

In general terms, Equation 4.3 may be written as
\[
\frac{\Theta_3(s)}{T(s)} = \Theta_{3R} + \Theta_{3L} + \Theta_{3H},
\]

(4.5)

where \( \Theta_{3R} \) is the rigid body component of the transfer function, and \( \Theta_{3L} \) and \( \Theta_{3H} \) are the low- and high-frequency flexible modes' components, respectively.

Note that the transfer function of the first disk may be expressed as a linear combination of the modal components of the third disk transfer function. This permits implementation of the third disk transfer function on the analog computer and formation of the first disk transfer function by using the existing third disk modal components. This relationship may be expressed as

\[
\frac{\dot{\Theta}_1(s)}{T(s)} = \dot{\Theta}_{3R} - 1.991 \dot{\Theta}_{3L} - 3.491 \times 10^{-3} \dot{\Theta}_{3H}.
\]

(4.6)

4.4 Digital Simulation

As a precursor to the hybrid simulation suggested in this chapter, a digital simulation of the experiment was conducted. Several benefits accrue from this tack. First, the analog computer realization may be modeled, and the range of values assumed by program variables under various experimental conditions may be readily ascertained. This information is necessary to "magnitude-scale" the analog computer variables and also provides a basis for
establishing the "structural constraints." Also, the effects of changing parameters of the identification process, e.g. sampling period or time record window, may be quickly evaluated. In addition, the results of the digital simulation identification process provide reference data to validate the implementation of the hybrid experiment.

The digital simulation was created using the MatrixX control system analysis and design software on an IBM PC-XT microcomputer. The MatrixX programs written to effect the investigation are presented in Appendix A. As a first step, the CALCFRSP program was developed to display the continuous and sampled-data frequency response functions corresponding to the Disk-3 velocity transfer function. Next, the RTLOCUS program was instituted to demonstrate the stabilizing effects of closing the Disk-1 loop in either the continuous or sampled-data domains. Finally, several subprograms were written and combined to provide an interactive system identification program titled SYSTEMID.

In order to generate sampled-data results, a sampling period must be chosen. According to the sampling theorem, the sampling rate must be at least twice the highest frequency of interest in the sampled signal [4]. Since there is a mode of 1.5 Hz. in the structure, a minimum sampling rate of 3 Hz. is required. However, to provide
some tolerance for modal uncertainty and to abate the aliasing of noise, as described in Chapter Two, a somewhat higher sampling rate is necessary. On the other hand, a higher rate increases the number of data points collected for a fixed time record window, and thereby complicates the data collection and analysis. In view of the foregoing, a sampling rate of 5 Hz. was selected for the experiment. Using this rate, the CALCFRSP program produces the ideal Disk-3 velocity sampled-data frequency response function shown in Figure 4.3.

The RTLOCUS program provides some insight into the MSPS stabilizing effects due to closure of the Disk-1 loop. First, for the continuous system, Figure 4.4 shows the root locus. Note that the single pole at the origin will migrate to the left along the real axis with increasing gain. The four poles on the imaginary axis move toward the adjacent zeros on this axis. Closer examination of a typical path taken by these poles is provided by the magnified view of Figure 4.5. This shows that the entire locus for the 0.1 Hz. mode is confined to the left-half plane for all closed-loop gains. Thus, the global effect of MSPS inherent in the complementary actuator-sensor pair configuration is graphically confirmed.
Figure 4.3. Three disk system sampled-data frequency response function.
Figure 4.4. Three disk system continuous root locus.
Figure 4.5. Three disk system continuous root locus of 0.1 Hz. mode.
Now, the feedback compensator gain used to stabilize the modes is selected by investigation of the sampled-data root locus. Some care must be taken in choosing this gain in the sampled-data case. If the gain is too large, MSPS can be lost in the sampled-data system, and the system becomes unstable. Also, careless gain choices may unduly suppress the closed-loop time responses. On the other hand, gains that are too low may not provide sufficient modal damping for limiting responses to desirable excitation signals. Another consideration for deterministic identification is that it is desirable to have the responses due to the pulse inputs to settle within the time record window. This obviates the need for using a tapering window to suppress side-lobe leakage due to the input responses in the Fourier analysis. In consideration of these constraints, a feedback gain of unity was deemed to be sufficient, and the resulting sampled-data root locus generated by the RTLOCUS program is shown in Figure 4.6. Note that all the roots for this gain are within the unit circle, so system stability is assured in the unity negative feedback configuration, which was analyzed in Chapter Two.

The system identification program SYSTEMID was designed to permit the user to have much flexibility in the identification process. The program provides the capability to specify any of the deterministic signals discussed in
Figure 4.6. Three disk system sampled-data root locus for unity gain.
Chapter Three and stochastic identification is possible using pseudo-random white noise. Also, the user may alter the sampling rate and the time record window. However, due to memory constraints, the number of data samples are limited to 256 for each record, and averaging of multiple records is not implemented. Additionally, no noise is added to the output "measurements", so the results are idealized.

4.5 Computations

At this point, a few comments are in order concerning the computation of the Disk-3 velocity open-loop frequency response function estimate. The analysis methods of Chapter Two are sufficient to find the Disk-1 response function estimate, if this is desired. With reference to the system block diagram shown in Figure 4.7, the Disk-3 response function may be estimated by noting the following relationships for deterministic excitations:

\[ \hat{H}_3(f) = \frac{\dot{\phi}_3(f)}{T(f)}, \]  
\[ (4.7) \]
\[ T(f) = X(f) - \dot{\phi}_1(f), \text{ and} \]  
\[ (4.8) \]
\[ \hat{H}_3(f) = \frac{\dot{\phi}_3(f)}{X(f) - \dot{\phi}_1(f)}. \]  
\[ (4.9) \]
Figure 4.7. Three disk system sampled-data frequency response block diagram.
Equation 4.9 shows the computations necessary for determining the estimate in terms of known or measurable signals' spectra.

A parallel analysis for stochastic excitations with the output noises uncorrelated to the input signal yields the following computations:

\[ \hat{R}_3(f) = \frac{\tilde{G}_{\tau\theta_3}(f)}{\tilde{G}_{\tau\tau}(f)}, \]  \hspace{1cm} (4.10)

\[ \tilde{G}_{\tau\tau}(f) = \tilde{G}_{xx}(f) - \tilde{G}_{\theta_1\theta_1}(f), \text{ and} \]  \hspace{1cm} (4.11)

\[ \hat{R}_3(f) = \frac{\tilde{G}_{\tau\theta_3}(f)}{\tilde{G}_{xx}(f) - \tilde{G}_{\theta_1\theta_1}(f)}. \]  \hspace{1cm} (4.12)

In this case, Equation 4.12 shows the necessary computations in terms of spectral density estimates.

4.6 Digital Results

In this section, the results of the digital simulation identification process are presented for each of the excitation signals introduced. Here, only the magnitudes of the frequency response functions are computed and displayed. However, this lays the groundwork for a more extensive investigation in the hybrid experiment.
The first results presented in Figure 4.8 are due to unit digital pulse excitation. The flexible modes are identified clearly at the expected frequencies in accordance with Figure 4.3. The noticeable spreading of the 0.1 Hz. peak occurs because the spectral resolution at this frequency is less than desirable. Since the digital experiment is limited to 256 data samples per record at a sampling period of 0.2 seconds, the record time window is 51.2 seconds. This implies a spectral resolution of about 0.02 Hz. which is a significant amount at the 0.1 Hz. frequency. The corresponding analysis for the hybrid experiment shows that the resolution will be increased fourfold and will not be nearly so objectionable.

Next, the results for an increased magnitude pulse are shown in Figure 4.9. Since no noise was added to the measurements for the digital simulation, this response is essentially the same as the response for the unit digital pulse. Later in the hybrid experiment, the advantages of this signal in the presence of noise will be demonstrated.

The response function for the increased duration pulse is displayed in Figure 4.10. As anticipated, satisfactory results are achieved at low frequencies; but at frequencies approaching the half-sample frequency, the spectral nulls of the excitation introduce aberrations into the identified
Figure 4.8. Frequency response function obtained using unit digital pulse.
Figure 4.9. Frequency response function obtained using increased magnitude pulse.
Figure 4.10. Frequency response function obtained using increased duration pulse.
frequency response function. If modes need to be identified at these frequencies, e.g. the 1.5 Hz. mode, a different excitation signal is indicated.

For the binomial triangle excitation results presented in Figure 4.11, the low frequency fidelity is again apparent. Also, the peak at 1.5 Hz. may be attributed to the response function since the excitation spectrum does not have any nulls below 2.5 Hz. However, this mode is on the borderline for confident identification.

In Figure 4.12 for the bipolar triangle excitation, the low frequency results are erroneous, but the high frequency mode is clearly defined. Again, this suggests using these shaped pulses in tandem for broad-spectrum identification applications. This is another point that is developed in the hybrid experiment.

Finally, Figure 4.13 shows the results of the digital experiment for a single record stochastic identification. Due to the small number of samples in the record and not averaging several results, the response function is not reliable for the identification of either mode. However, there is considerable evidence of the 1.5 Hz. mode. A more realistic application of stochastic identification is provided in the following chapter.
Figure 4.11. Frequency response function obtained using binomial triangle shaped pulse.
Figure 4.13. Frequency response function obtained using unity white noise.
Figure 4.12. Frequency response function obtained using bipolar triangle shaped pulse.
CHAPTER 5.
EXPERIMENT IMPLEMENTATION

5.1 Hybrid Experiment

The hybrid experiment hardware configuration is shown in Figure 5.1. The continuous system dynamics are modeled on the EAI Analog Computer Model TR-20. The digital controller is a Hewlett-Packard (HP) HP-3852A Data Acquisition/Control Unit (DACU). The D/A converter is an HP-44727A Digital to Analog Voltage Converter, and the A/D converter is an HP-44702A High Speed Voltmeter. Both the converters are option modules installed in the HP-3852A DACU. Finally, the digital computer is an HP Series 9000 Model 300 computer which is linked to the DACU via the HPIB parallel bus (IEEE-488).

The hybrid experiment system identification is accomplished in two steps. First, the real-time simulation is executed to elicit time responses due to the desired excitation signal. During this part of the process, the DACU drives the system and logs the necessary data, while the digital computer generates a video display "strip chart" of the first twenty seconds of the simulation. After the required number of time records are collected and stored on diskette, the second part of the identification process,
Figure 5.1. Hybrid experiment hardware configuration.
frequency analysis and response function computation, may be executed. The hybrid experiment incorporates all the input signals that the digital simulation accommodated. In addition, averaging of results for several records is possible in this implementation. Also, there is sufficient noise inherent in the sampled-data system to demonstrate the fidelity differences in response function estimates. The function of each of the major hardware components and its associated program(s) are summarized next.

5.2 Analog Computer

The Disk-3 and Disk-1 velocities' transfer functions are programmed on the TR-20 analog computer as mentioned in Chapter Four. An investigation of the analog computer realization was conducted with the digital simulation to find the range for each differential equation variable. A pseudo-random gaussian sequence of zero mean and unity variance was used as the excitation signal. Based on the results of this examination, a magnitude-scaled program was developed for the analog computer. The program is represented in functional block form in Figure 5.2 using integrators, summers/inverters, and potentiometers [8]. The variables shown in this figure are normalized "x" variables corresponding to the subscript variable. Also, the numbering of the various components in this figure
Figure 5.2. Analog computer simulation functional block diagram.
corresponds to the actual hardware tag numbering. The potentiometers provide magnitude-scaling of the differential equations' variables, feedback gains, and scaling for linear combinations of modal components. The resultant values determined for the potentiometers are recorded in Table 5.1.

<table>
<thead>
<tr>
<th>Analog Computer Potentiometer Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pot.</td>
</tr>
<tr>
<td>------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>

Table 5.1. Analog computer potentiometer settings.

5.3 Data Acquisition/Control Unit

The DACU is loaded with a data acquisition/control subroutine and the specific excitation sequence by the digital computer. During the execution of the subroutine, the DACU performs the following functions:
* Reads and stores the system outputs via the A/D.

* Computes the closed-loop error signal and sends it to the system via the D/A.

* Sends the first twenty seconds of data to the digital computer via the HPIB for strip chart display.

Since the DACU program is downloaded from the digital computer, it is part of the SIMULATION program in the hybrid experiment software listed in Appendix B.

5.4 Digital Computer

There are three programs listed in Appendix B that are used by the digital computer. The first is SIMULATION, which is an interactive program to drive the time data acquisition part of the identification process. The second is SIUDI (System Identification Using Deterministic Inputs), which does the frequency domain analysis, frequency response function computations, and averaging for deterministic excitations. The third is SIUSI (System Identification Using Stochastic Inputs), which performs the corresponding functions for stochastic excitation analysis.

The SIMULATION program performs the following major functions:
* Prompts user to specify the excitation signal and number of data records to add to the storage archive.

* Downloads subroutine and excitation signal to DACU.

* Displays strip chart of time data.

* Retrieves time data records from DACU and stores them in the archive.

The hardware and software outlined above were used to evaluate the effectiveness of the signals discussed in Chapter Three for identifying this hybrid system. The results of this investigation are considered next.

5.5 Hybrid Results

The results of the hybrid experiment presented in this section are divided into two parts. First, the signal set used in the digital experiment is evaluated in this case with the addition of phase characteristics. Then, a group of signals is selected whose characteristics are desirable for this specific system. An extensive identification process is conducted with these signals which includes averaging multiple results. Throughout this section, graphical analysis is again employed to judge the relative effectiveness of the various methods considered.
First, Figure 5.3 displays the unit digital pulse's characteristics, and Figure 5.4 presents the identified frequency response function. Both the magnitude and phase functions are very noisy and contain little information. The 1.5 Hz. mode's magnitude peak is discernible, but on the whole this response function is very unsatisfactory.

Next, the increased magnitude pulse data is given in Figures 5.5 and 5.6 respectively. There is remarkable improvement in the results. The low frequency magnitude and phase are close to the ideal values, and the high frequency magnitude peak is clearly identifiable. The slight shift in frequency of the low frequency mode from the target value of 0.1 Hz. is most likely due to slight inaccuracies inherent in setting the analog computer potentiometers.

The increased duration pulse results are provided in Figures 5.7 and 5.8. The low frequency fidelity compares favorably with that achieved in the increased magnitude case, but the spectral nulls of the input interfere with high frequency identification.

Now, the binomial triangle shaped pulse excitation is considered in Figures 5.9 and 5.10. This signal performs better than the unit digital pulse at low frequencies, but it has no merit at high frequencies. The increased magnitude and increased duration pulses show superior
Figure 5.3. Spectral characteristics of unit digital pulse.
Figure 5.4. Frequency response function using unit digital pulse.
Figure 5.5. Spectral characteristics of increased magnitude pulse.
Figure 5.6. Frequency response function using increased magnitude pulse.
Figure 5.7. Spectral characteristics of increased duration pulse.
Figure 5.8. Frequency response function using increased duration pulse.
Figure 5.9. Spectral characteristics of binomial triangle shaped pulse.
Figure 5.10. Frequency response function using binomial triangle shaped pulse.
results; however, it should be noted that their spectral magnitudes are arbitrarily higher in this case. In the second group of signals considered in this section, the relative strengths of the excitation signals are chosen more equitably.

In the complementary case of the bipolar triangle pulse, the results are shown in Figures 5.11 and 5.12. Again, at this excitation level the results show no merit.

The final result from the first group is the stochastic excitation response function of Figure 5.13. Here the magnitude peaks of the two modes are noticeable, but the low frequency peak is badly spread and the high frequencies are quite noisy. The phase characteristic is also quite noisy, but it tracks the expected phase somewhat at low frequencies.

For the second part of the experiment, the signals were tailored for use under the constraints of the given system. Recall that the analog computer realization was magnitude-scaled based on the results of the digital simulation excited with zero mean, unity variance gaussian noise. Therefore, the digital simulation was used to scale the magnitudes of the excitation signals so the linearity constraints of the analog computer would be observed. Furthermore, due to modes of interest near the half-sampling
Figure 5.11. Spectral characteristics of bipolar triangle shaped pulse.
Figure 5.12. Frequency response function using bipolar triangle shaped pulse.
Figure 5.13. Frequency response function using unity white noise.
frequency, the protracted constant amplitude pulse was omitted. These constraints lead to consideration of the following excitation signals:

* Increased magnitude pulse (10x1).

* Binomial triangle shaped pulse (2x10).

* Bipolar triangle shaped pulse (-2x10).

* Unity white gaussian noise $N(0,1)$.

Additionally, the benefits from averaging results are demonstrated by comparing the average response functions for one, four, and sixteen records in each case.

First, the results for the increased magnitude pulse are shown in Figures 5.14, 5.15, and 5.16. The reduction of noise in the response function estimate due to averaging is clearly demonstrated in both the magnitude and phase plots, especially at high frequencies where the signal to noise ratio is smaller.

Next, the binomial triangle excitation is considered in Figures 5.17, 5.18, and 5.19. These results improve with averaging at low frequencies, but the high frequency results are not usable.
Figure 5.14. Frequency response function using single increased magnitude record.
Figure 5.15. Frequency response function using four increased magnitude records.
Figure 5.16. Frequency response function using sixteen increased magnitude records.
Figure 5.17. Frequency response function using single binomial triangle record.
Figure 5.18. Frequency response function using four binomial triangle records.
Figure 5.19. Frequency response function using sixteen binomial triangle records.
The bipolar triangle signal's performance is displayed in Figures 5.20, 5.21, and 5.22. These results near the half-sampling frequency improve with averaging, but the low frequency regions are not meaningful. In Figure 5.22, the high frequency mode is identifiable. The spliced results of the complementary pulse pair are shown in Figure 5.23. In this method, both the flexible modes are distinguishable.

Finally, the results from noise excitation are presented in Figures 5.24, 5.25, and 5.26. Here, the low frequency results are not of the quality achieved with deterministic excitations. However, the high frequency fidelity improves dramatically with averaging.

The results of the experiment show that there are some clear distinctions in frequency response function estimate fidelity due to the excitation signal. For low frequency fidelity, a deterministic method using a signal of high spectral magnitude is desirable. Also, a deterministic tack may be indicated if averaging is not to be performed, or if only a few records will be used in an average. On the other hand, with a moderate number of records to include in an average, excellent high frequency fidelity is achievable with stochastic methods. Of course, for broad-spectrum applications, a combination of deterministic and stochastic
Figure 5.20. Frequency response function using single bipolar triangle record.
Figure 5.21. Frequency response function using four bipolar triangle records.
Figure 5.22. Frequency response function using sixteen bipolar triangle records.
Figure 5.23. Frequency response function using sixteen spliced triangle records.
Figure 5.24. Frequency response function using single unity white noise record.
Figure 5.25. Frequency response function using four unity white noise records.
Figure 5.26. Frequency response function using sixteen unity white noise records.
approaches may be employed. In any case, the results of this chapter should provide some insight into the successful selection of suitable excitation signals.
CHAPTER 6.
CONCLUSION

6.1 Observations

In this thesis, methods of improving the fidelity of empirical sampled-data frequency response function estimates have been investigated. The successful identification of a high-fidelity system response function is the cornerstone of the LSS design methodology, Design-To-Performance, which was outlined in Chapter One. The analytical methods reviewed in Chapter Two permit the identification of the frequency response function matrix necessary for the One-Controller-At-A-Time design procedure discussed in the introduction.

In order to achieve high-fidelity estimates, several methods of enhancing the properties of excitation signals were considered in Chapter Three. These methods include increasing the signal's magnitude and/or duration, shaping the signal, and using complementary signal pairs. Other means of improving results include averaging several responses and combining deterministic and stochastic methods advantageously.

The system identification process and signals of interest were explored via digital simulation in Chapter
Four. Finally, the practical application of the methods discussed was demonstrated with the hybrid system experiment and results presented in Chapter Five.

6.2 Applications

The methods of enhancing excitation signals used in system identification discussed in this thesis were developed from the viewpoint of LSS applications. However, the same principles apply to any application that requires the empirical identification of a sampled-data frequency response function. Likely applications include very high order systems or systems whose dynamics are not well understood. For instance, many biomedical systems may lend themselves to a similar analysis.

As a direct application of the results detailed in this thesis, the identified response functions could be used to design controllers for the hybrid system. In this manner, the influence of the identification methods on the closed-loop system performance could be investigated. Since this is the purpose for the identification process, it is the true test of adequate fidelity of the identified response function estimate. As a more ambitious project, the analog computer simulation could be replaced with a
physical system, e.g. a pointing system. Such a demonstration should establish the usefulness of the methods described herein.
REFERENCES


APPENDIX A

Digital Simulation Source Code
This appendix contains the source code listings for the digital simulation of Chapter Four. The programs are executable command files for the MatrixX control system analysis and design software package.

The three disk system described in Chapter Four is incorporated into MatrixX variables in the ANCMPINI program, which initializes the digital simulation of the analog computer. The programs for the simulation provide three major functions. The CALCFRSP program permits the generation of continuous and sampled-data frequency response functions for the system. The RTLOCUS program provides the capability to examine the continuous and discrete root loci of the system. Finally, the SYSTEMID program accesses several subprograms to perform a system identification simulation.

There are five subprograms called by the SYSTEMID program. The first is GETPARAM, which gets the system identification parameters for the user. The DISCRTCL module forms the discrete closed-loop system. Then, the GENINPUT module generates the input signal defined by the user. Finally, the time simulation and frequency analysis are performed by the DTRMNRSP program for deterministic analysis or by the STOCHRSP program for stochastic analysis.
NOTE: Comment lines in MatrixX cannot be placed in compound statements. In the listings that follow, comments precede the compound statements which they describe, e.g. WHILE loops.
This file is a MatrixX executable command file. Its purpose is to initialize the MatrixX main program with a state-space description of the system analyzed in the thesis. A modal realization is used paralleling that implemented on the analog computer. This permits simulation of the analog computer's responses to various signals. In order to effectively magnitude scale the analog computer simulation, the dynamic ranges of the states of the realization must be determined. Then the variables generated by the analog computer can be normalized to prevent their exceeding the dynamic range of the computer's amplifiers. Additionally, the MatrixX variables initialized by this program are SAVED in the file ANCMPSIM.VAR for use by other program modules.

The rigid body mode of the system's response is modeled by subsystem A. The 0.1 Hz flexible body mode is modeled by subsystem B, and the 1.5 Hz flexible body mode is modeled by subsystem C.
// AA, AB, AC: coefficient matrices of state vectors in the
// state equations of the subsystems
// BA, BB, BC: coefficient matrices of input vectors in the
// state equations of the subsystems
// CA, CB, CC: coefficient matrices of state vectors in the
// output equations of the subsystems
// DA, DB, DC: coefficient matrices of input vectors in the
// output equations of the subsystems
// NSA, NSB, NSC: integers indicating the subsystems' orders
// SA, SB, SC: matrices composed of coefficient matrices to form
// the MatrixX representations of the subsystems

// BEGIN

// CLEAR SCREEN AND DISPLAY PROGRAM FUNCTION
ERASE
DISPLAY('INITIALIZING ANALOG COMPUTER SIMULATION . . .')
SHORT E

// DEFINE ANALOG COMPUTER SIMULATION SUBSYSTEMS
AA = [0 1; 0 0];
BA = [0; 1/3];
CA = [1 0; 0 1];
DA = [0; 0];
SA = [AA BA; CA DA];
NSA = 2;
AB = [0 1; -0.39478 0];
BB = [0; -0.33482];
CB = [1 0; 0 1];
DB = [0; 0];
SB = [AB BB; CB DB];
NSB = 2;
AC = [0 1; -88.82686 0];
BC = [0; 1.48807E-3];
CC = [1 0; 0 1];
DC = [0; 0];
SC = [AC BC; CC DC];
NSC = 2;

// CLEAR SUBSYSTEM COEFFICIENTS AND SAVE SUBSYSTEM DESCRIPTIONS -------
CLEAR AA A8 AC SA 88 Be CA C8 CC DA DB DC
SAVE('ANCMPSIM.VAR')

// CLEAR VARIABLE STACK -------------------------------------------------------
CLEAR

// END OF ANCMPINI PROGRAM =====================================================
RETURN
PROGRAM

NAME: CALCFRSP

AUTHOR: T T HAMMOND

DATE: 10 APR 1988

PURPOSE:
This file is a MatrixX executable command file. Its purpose is to generate the calculated frequency response for the continuous-time analog computer or the discretized analog computer system. The file ANCMPSIM.VAR must be placed in the current directory prior to the execution of this program.
The response functions calculated are for Disk-3 velocity.

VARIABLES

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHOICE</td>
<td>index of menu selection</td>
</tr>
<tr>
<td>DELTAT</td>
<td>sampling period in seconds</td>
</tr>
<tr>
<td>DSNFRESP</td>
<td>order of discrete system</td>
</tr>
<tr>
<td>DSFRESP</td>
<td>MatrixX description of discrete system</td>
</tr>
<tr>
<td>K</td>
<td>compensator gain</td>
</tr>
<tr>
<td>KFB</td>
<td>feedback matrix</td>
</tr>
<tr>
<td>MFRESP</td>
<td>connect function input matrix</td>
</tr>
<tr>
<td>NAB, NABC</td>
<td>intermediate subsystems' orders</td>
</tr>
<tr>
<td>NFRESP</td>
<td>connect function output matrix</td>
</tr>
<tr>
<td>NSA, NSB</td>
<td></td>
</tr>
</tbody>
</table>
// NSC : integers indicating the subsystems' orders
// NSFRESP : order of analog system
// SA, SB, SC : matrices composed of coefficient matrices to form
// the MatrixX representations of the subsystems
// SAB, SABC : intermediate subsystem matrices
// SFRESP : MatrixX description of analog system

// CONSTANTS

// SELECTION MENUS:
TIMEMENU = ['** MENU **','
            ' CONTINUOUS ','
            ' DISCRETE '];

//
MOREMENU = ['** MENU **','
            ' MORE '
            ' QUIT '];

//
// QUERY RESPONSES:
N = 'N';
NO = 'N';
Y = 'Y';
YES = 'Y';

//
// BEGIN
CLEAR SCREEN AND DISPLAY PROGRAM FUNCTION
ERASE
DISPLAY('FREQUENCY RESPONSE FUNCTION GENERATION . . .')

// RETRIEVE SYSTEM SIMULATION AND FORM DISK-3 VELOCITY OUTPUT
LOAD('ANCMPSIM.VAR')

[SAB, NAB] = APPEND(SA, NSA, SB, NSB);
[SABC, NABC] = APPEND(SAB, NAB, SC, NSC);
MFRESP = [1 1 1]';
NFRESP = [0 1 0 1 0 1];
KFB = [0 0 0 0 0; 0 0 0 0 0; 0 0 0 0 0];
[SFRESP, NFRESP] = CONNECT(SABC, NABC, KFB, MFRESP, NFRESP);

// WHILE LOOP COMMENTS

// SET UP LOOP FOR MULTIPLE RESPONSES

// QUERY USER FOR CONTINUOUS OR DISCRETE RESPONSE

// SPACE CONTINUOUS FREQUENCY DATA LOGARITHMICLY

// DISPLAY CONTINUOUS FREQUENCY RESPONSE FUNCTION

// INQUIRE IF PRINTING IS DESIRED

// QUERY USER FOR SAMPLING PERIOD

// SPACE DISCRETE FREQUENCY DATA LOGARITHMICLY

// DISPLAY SAMPLED-DATA FREQUENCY RESPONSE FUNCTION

// INQUIRE IF PRINTING IS DESIRED

// QUERY USER FOR MORE RESPONSES

CHOICE = 0;

WHILE CHOICE <> 2;...

CHOICE = MENU(TIMEMENU,1);...

IF CHOICE = 1;...

GEOMRATIO = 10**(1/250);...
FRANGE = [0.05*PI 0*ONES(1,500)]';...
FOR I=1:500, FRANGE(I+1)=GEOMRATIO*FRANGE(I); END,....
H = FREQ(SFRESP, NFRESP, FRANGE);

FHZ = FRANGE/(2*PI);

PLOT(FHZ, 20*LOG(ABS(H))/LOG(10),

'TITL/DISK 3 VELOCITY CONTINUOUS FREQUENCY RESPONSE FUNCTION/...

YLAB/db/...

XLAB/FREQUENCY IN HZ/...

LOGX');

INQUIRE CHOICE 'DO YOU WANT TO PRINT THIS FUNCTION . . .';

IF CHOICE = 'Y',

PLOT('PRINTER HOLD');

PLOT(FHZ, 20*LOG(ABS(H))/LOG(10),

'TITL/DISK 3 VELOCITY CONTINUOUS FREQUENCY RESPONSE FUNCTION/...

YLAB/db/...

XLAB/FREQUENCY IN HZ/...

LOGX');

ERASE;

PLOT('DISPLAY HOLD');

END;

ERASE;

ELSE INQUIRE DELTAT 'ENTER SAMPLING PERIOD IN SECONDS';

[DSFRESP, DNFRESP] = DISCRETIZE(SFRESP, NFRESP, DELTAT);

GEOMRATIO = 10**(1/250);

FRANGE = [0.05*PI*DELAT 0*ONES(1,500)]';

FOR I=1:500, FRANGE(I+1)=GEOMRATIO*FRANGE(I); END,

H = FREQ(DSFRESP, DNFRESP, FRANGE, 'DIS');

FHZ = FRANGE/(2*PI*DELAT);

PLOT(FHZ, 20*LOG(ABS(H))/LOG(10),

122
'TITL/DISK 3 VELOCITY SAMPLED-DATA FREQUENCY RESPONSE FUNCTION/... YLAB/dB/...
XLAB/FREQUENCY IN HZ/...

LOGX');...

INQUIRE CHOICE 'DO YOU WANT TO PRINT THIS FUNCTION . . .?';...

IF CHOICE = 'y'

PLOT('PRINTER HOLD');...

PLOT(FHZ, 20*LOG(ABS(H))/LOG(10));...

'TITL/DISK 3 VELOCITY SAMPLED-DATA FREQUENCY RESPONSE FUNCTION/...

YLAB/dB/...

XLAB/FREQUENCY IN HZ/...

LOGX');...

ERASE;...

PLOT('DISPLAY HOLD');...

END;...

ERASE;...

END;...

CHOICE = MENU(MOREMENU,1);...

END

// CLEAR VARIABLE STACK -----------------------------------------------

CLEAR

// END OF CALCFRSP PROGRAM *********************************************

RETURN
PROGRAM

NAME: RTLOCUS

AUTHOR: T T HAMMOND

DATE: 10 APR 1988

PURPOSE:

This file is a MatrixX executable command file. Its purpose is to generate the root locus for the continuous-time analog computer simulation or the discretized analog computer system. The file ANCMPSIM.VAR must be placed in the current directory prior to the execution of this program.

VARIABLES

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHOICE</td>
<td>index of menu selection</td>
</tr>
<tr>
<td>DELTAT</td>
<td>sampling period in seconds</td>
</tr>
<tr>
<td>DSNRTLOC</td>
<td>order of discrete system</td>
</tr>
<tr>
<td>DSRTLOC</td>
<td>MatrixX description of discrete system</td>
</tr>
<tr>
<td>EV</td>
<td>matrix of discrete poles corresponding to GAIN vector</td>
</tr>
<tr>
<td>GAIN</td>
<td>row vector of gains for desired discrete roots</td>
</tr>
<tr>
<td>K</td>
<td>compensator gain</td>
</tr>
<tr>
<td>KFB</td>
<td>feedback matrix</td>
</tr>
<tr>
<td>MRTLOC</td>
<td>connect function input matrix</td>
</tr>
<tr>
<td>NAB, NABC</td>
<td>intermediate subsystems' orders</td>
</tr>
<tr>
<td>NRTLOC</td>
<td>connect function output matrix</td>
</tr>
<tr>
<td>NSA, NSB</td>
<td></td>
</tr>
</tbody>
</table>
// NSC : integers indicating the subsystems' orders
// NSRTLOC : order of analog system
// SA, SB, SC : matrices composed of coefficient matrices to form
// the MatrixX representations of the subsystems
// SAB, SABC : intermediate subsystem matrices
// SRTLOC : MatrixX description of analog system


// CONSTANTS


// SELECTION MENUS:
TIMEMENU = ['** MENU **'
    ' CONTINUOUS ''
    ' DISCRETE '];

// MOREMENU = ['** MENU **'
    ' MORE ''
    ' QUIT '];

// QUERY RESPONSES:
N = 'N';
NO = 'N';
Y = 'Y';
YES = 'Y';

// BEGIN
CLEAR SCREEN AND DISPLAY PROGRAM FUNCTION
ERASE
DISPLAY('ROOT LOCUS GENERATION . . .')

// RETRIEVE SIMULATED SYSTEM AND FORM DISK-1 VELOCITY OUTPUT
LOAD('ANCMPSIM.VAR')

[SAB, NAB] = APPEND(SA, NSA, SB, NSB);
[SABC, NABC] = APPEND(SAB, NAB, SC, NSC);
MRTLOC = [1 1 1]
NRTLOC = [0 1 0 -1.9909 0 -3.4905E-3];
KFB = [0 0 0 0 0 0; 0 0 0 0 0 0; 0 0 0 0 0 0];
[SRTLOC, NRTLOC] = CONNECT(SABC, NABC, KFB, MRTLOC, NRTLOC);

// WHILE LOOP COMMENTS

// SET UP LOOP FOR MULTIPLE LOCI

// QUERY USER FOR CONTINUOUS OR DISCRETE ROOT LOCUS

// DISPLAY CONTINUOUS ROOT LOCUS

// QUERY USER FOR PRINTING CONTINUOUS ROOT LOCUS

// INQUIRE FOR SAMPLING PERIOD AND GAIN VALUES

// DISPLAY DISCRETE ROOT LOCUS

// QUERY USER FOR PRINTING DISCRETE ROOT LOCUS

// QUERY USER FOR MORE LOCI

CHOICE = 0;

WHILE CHOICE <> 2;

CHOICE = MENU(TIMEMENU, 1);

IF CHOICE = 1;

K = RLOCUS(SRTLOC, NRTLOC);

INQUIRE CHOICE 'DO YOU WANT TO PRINT THIS LOCUS . . .';

IF CHOICE = 'Y',

PLOT('PRINTER HOLD');

EV = RLOCUS(SRTLOC, NRTLOC, K);
ERASE;...

PLOT('DISPLAY HOLD');...

END;...

ERASE;...

ELSE INQUIRE DELTAT 'ENTER SAMPLING PERIOD IN SECONDS';...

[DSRTLOC, DNRTLOC] = DISCRETIZE(SRTLOC, NRTLOC, DELTAT);...

INQUIRE GAIN 'ENTER VECTOR FOR GAINS OF ROOTS TO BE SHOWN';...

EV = DRLOCUS(DSRTLOC, DNRTLOC, GAIN);...

GAIN,...

INQUIRE CHOICE 'DO YOU WANT TO PRINT THIS LOCUS ....';...

IF CHOICE = 'Y',...

PLOT('PRINTER HOLD');...

EV = DRLOCUS(DSRTLOC, DNRTLOC, GAIN);...

ERASE;...

PLOT('DISPLAY HOLD');...

END;...

ERASE;...

END;...

CHOICE = MENU(MOREMENU, 1);...

END

// CLEAR VARIABLE STACK -----------------------------------------------

CLEAR

// END OF RTLOCUS PROGRAM =============================================

RETURN
PROGRAM

NAME: SYSTEMID

AUTHOR: T T HAMMOND

DATE: 9 APR 1988

PURPOSE:
This file is a MatrixX executable command file. Its purpose is to drive the system identification process. This is accomplished by calling smaller MatrixX executable command files to perform required tasks.

Preparation to use this file must include creating the ANCMPSIM.VAR file in the DOS default directory. That file contains the analog computer simulation MatrixX description. It is created by executing the ANCMPI.NI.EXC file.

Files read during the execution of this file must be in the DOS default directory. Files written during execution will be placed in this same directory.

VARIABLES

ANSWER : response to question

TYPE : = 1 for deterministic input signal

= 2 for stochastic input signal

CONSTANTS
N = 'N';
NO = 'N';
//
Y = 'Y';
YES = 'Y';
//
// BEGIN
// CLEAR SCREEN AND DISPLAY PROGRAM INSTRUCTIONS
ERASE

//

DISP('******** SYSTEM IDENTIFICATION ********')
DISP(')
DISP(' This program simulates the sampled-data frequency )
DISP(' response function method of system identification )
DISP(' applied to a highly flexible structure.
DISP(')
DISP(' A variety of input signals may be specified by menu )
DISP(' responses. Also the sampling period and length of time )
DISP(' for each sampled record is selectable. The maximum )
DISP(' number of samples must be restricted to 256.
DISP(')
DISP(' The program pauses at several statements until the user )
DISP(' is ready to continue. To continue, enter <RETURN> at )
DISP(' the PAUSE> prompt.
DISP(')
DISP(' Time responses and frequency responses of pertinent )
DISP(' data are displayed graphically. These are plotted on )
DISP('* the default plotting device (video display at power-on). *')
DISP('*
DISP('* * * * * * * * T T HAMMOND 21 APR 88 * * * * * * * *')

//
// WAIT FOR USER RESPONSE ..................................................
DISP(' ')
PAUSE

//
// WHILE LOOP COMMENTS ......................................................
// SET UP LOOP FOR MULTIPLE IDENTIFICATIONS ................................
// GET SYSTEM AND INPUT SIGNAL PARAMETERS FROM USER ........................
// FORM DISCRETE CLOSED-LOOP SYSTEM ...........................................
// GENERATE INPUT SIGNAL SEQUENCE ............................................
// DETERMINISTIC INPUT SIGNAL SYSTEM IDENTIFICATION ........................
// STOCHASTIC INPUT SIGNAL SYSTEM IDENTIFICATION ...........................
// QUERY USER FOR ANOTHER IDENTIFICATION ....................................

ANSWER = 'Y';

WHILE ANSWER = 'Y',...

  EXECUTE('GETPARAM.EXC'),...
  EXECUTE('DISCRTCL.EXC'),...
  EXECUTE('GENINPUT.EXC'),...
  IF TYPE = 1, ...
    EXECUTE('DTRMNRSPEX.C'),...
  ELSE, ...
    EXECUTE('STOCHRSP.EXC'),...
  END, ...

INQUIRE ANSWER 'DO YOU WANT TO DO ANOTHER RESPONSE . . . ',...
END

// END OF SYSTEMID PROGRAM

RETURN
PROGRAM

NAME: GETPARAM

AUTHOR: T T HAMMOND

DATE: 6 APR 1988

PURPOSE:
This file is a MatrixX executable command file. Its purpose is to establish values for the parameters necessary to run the thesis simulation. Default parameters are set and the user is prompted to change any parameters that he/she desires.

VARIABLES

CHOICE : index of menu selection
DELTAT : sampling period in seconds
NSAMPLES : number of samples rounded to next higher power of 2
PULSAMPLTD : input pulse amplitude (normalized -10 to 10)
PULSDURATN : length of input pulse in sample periods
RECORDUR : length of time signals observed in seconds
SHAPE : = 1 for rectangular input pulse
         = 2 for binomial triangle shaped input pulse
         = 3 for bipolar triangle shaped input pulse
TYPE : = 1 for deterministic input signal
       = 2 for stochastic input signal
// CONSTANTS
BLANK15 = ' 
;

// SELECTION MENUS:
CHANGEMENU = ['* MENU *'
   ' ACCEPT '
   ' CHANGE ' ];

// SHAPEMENU = ['* SHAPE MENU *'
   ' RECTANGLE '
   ' BINOMIAL TRIANGLE'
   ' BIPOLAR TRIANGLE ' ];

// TYPEMENU = ['* TYPE MENU *'
   ' DETERMINISTIC '
   ' STOCHASTIC ' ];

// BEGIN

SHORT

// SET DEFAULT PARAMETERS
RECORDUR = 40;
DELTAT = 0.2;
NSAMPLES = 2**ROUND(0.5 + LOG(RECORDUR/DELTAT)/LOG(2));
TYPE = 1;
PULSDURATN = 1;
SHAPE = 1;
PULSAMPLTD = 1;

//

ERASE

// QUERY USER FOR RECORD DURATION -----------------------------------------------
DISPLAY('CURRENT TIME RECORD DURATION IS'), RECORDUR
PAUSE

CHOICE = MENU(CHANGEMENU,1);

IF CHOICE = 2,...

 INQUIRE RECORDUR 'ENTER TIME RECORD DURATION IN SECONDS',...

 RECORDUR,...

 NSAMPLES = 2**ROUND(0.5 + LOG(RECORDUR/DELTAT)/LOG(2));...

 PAUSE,...

END

//

ERASE

// QUERY USER FOR SAMPLING PERIOD -----------------------------------------------
DISPLAY('CURRENT SAMPLING PERIOD IS'), DELTAT
PAUSE

CHOICE = MENU(CHANGEMENU,1);

IF CHOICE = 2,...

 INQUIRE DELTAT 'ENTER SAMPLING PERIOD IN SECONDS',...

 DELTAT,...

 NSAMPLES = 2**ROUND(0.5 + LOG(RECORDUR/DELTAT)/LOG(2));...

 PAUSE,...

END

//
ERASE

// QUERY USER FOR DETERMINISTIC OR STOCHASTIC EXCITATION

DISPLAY('CURRENT INPUT SIGNAL TYPE IS')
DISPLAY(TYPMENU(TYPE + 1,:))
PAUSE

CHOICE = MENU(CHANGEMENU,1);

IF CHOICE = 2,...

TYPE = MENU(TYPMENU,1),....
DISPLAY(TYPMENU(TYPE + 1,:)),...
PAUSE,...

END

//

// DETERMINISTIC SIGNAL SECTION COMMENTS

// QUERY USER FOR PULSE DURATION

// QUERY USER FOR PULSE SHAPE

// QUERY USER FOR PEAK PULSE AMPLITUDE

IF TYPE = 1,...

ERASE,...

DISPLAY('CURRENT PULSE DURATION IS'), PULSDURATN,...
PAUSE,...

CHOICE = MENU(CHANGEMENU,1);...

IF CHOICE = 2,...

INQUIRE PULSDURATN 'ENTER PULSE DURATION IN SAMPLE PERIODS',...

PULSDURATN = ROUND(PULSDURATN),...

PAUSE,...

END,...

ERASE,...
DISPLAY('CURRENT PULSE SHAPE IS'),...
DISPLAY(SHAPEMENU(SHAPE + 1,:)),...
PAUSE,...
CHOICE = MENU(CHANGEMENU,1);...
IF CHOICE = 2,...
    SHAPE = MENU(SHAPEMENU,1),...
    DISPLAY(SHAPEMENU(SHAPE + 1,:)),...
    PAUSE,...
END,...
ERASE,...
DISPLAY('CURRENT PULSE AMPLITUDE IS'), PULSAMPLTD,...
PAUSE,...
CHOICE = MENU(CHANGEMENU,1);...
IF CHOICE = 2,...
    INQUIRE PULSAMPLTD 'ENTER PEAK PULSE AMPLITUDE (-10 TO 10)',...
    PULSAMPLTD,...
    PAUSE,...
END,...
END
//
// CLEAR UNNECESSARY VARIABLES ------------------------------------------
CLEAR BLANK15 CHANGEMENU TYPEMENU SHAPEMENU CHOICE RECORDUR
// END OF GETPARAM PROGRAM -------------------------------------------------
RETURN
// PROGRAM ============================================================

//

// NAME: DISCRTCL

//

// AUTHOR: T T HAMMOND

//

// DATE: 5 APR 1988

//

// PURPOSE:
// This file is a MatrixX executable command file. Its purpose is to
// create a MatrixX description of the sampled-data system examined
// in the thesis. The MatrixX description of the analog computer
// realization must be available in the file ANCMPSIM.VAR in the DOS
// default directory.
// This program incorporates a stabilizing feedback loop to the input
// using collocated, complementary sensing (i.e. the rotational
// velocity of disk one is the feedback to the torque input of disk
// one).

// VARIABLES ==============================================================

// DAB, DABC : intermediate subsystem descriptions
// DELTAT : sampling period in seconds
// DSA, DS8, DSC : descriptions of digital subsystems
// K : compensator gain
// KFB : feedback matrix
// NAB, NABC : intermediate subsystems' orders
// NCL : order of closed-loop system description
// NSA, NSB,
// NSC : integers indicating the subsystems' orders
// SA, SB, SC : matrices composed of coefficient matrices to form
// the MatrixX representations of the subsystems
// SCL : closed-loop system MatrixX description

// CONSTANTS ================================================================

// LOOP CLOSING PARAMETERS:
IN GAIN = [1 1 1];
K = -1;
KFB = K*[0 1 0 -1.9909 0 -3.4905E-3;...
          0 1 0 -1.9909 0 -3.4905E-3;...
          0 1 0 -1.9909 0 -3.4905E-3];

// BEGIN =====================================================================

// CLEAR SCREEN AND DISPLAY PROGRAM FUNCTION -------------------------------
ERASE
DISPLAY('DISCRETIZING THE CLOSED-LOOP SYSTEM . . .')
SHORT E

// RETRIEVE SIMULATED SYSTEM AND DISCRETIZE EACH SUBSYSTEM -----------------
LOAD('ANCMPSIM.VAR')
DSA = DISCRETIZE(SA, NSA, DELTAT);
DSB = DISCRETIZE(SB, NSB, DELTAT);
DSC = DISCRETIZE(SC, NSC, DELTAT);

// CLOSE DISK-1 VELOCITY FEEDBACK LOOP ---------------------------------------
[DAB, NAB] = APPEND(DSA, NSA, DSB, NSB);
[DABC, NABC] = APPEND(DAB, NAB, DSC, NSC);

[SCL, NCL] = CONNECT(DABC, NABC, KFB, INGAIN);

// CLEAR UNNECESSARY VARIABLES
CLEAR DAB DABC DSA DSB DSC KFB NAB NABC NSA NSB NSC SA SB SC

// STORE CLOSED-LOOP DISCRETE SYSTEM ON DISKETTE
SAVE('DISCRTCL.VAR')

// END OF DISCRTCL PROGRAM
RETURN
PROGRAM

NAME: GENINPUT

AUTHOR: T T HAMMOND

DATE: 7 APR 1988

PURPOSE:
This file is a MatrixX executable command file. Its purpose is to generate the input vector TAU for the digital system simulation. The input signal is a rectangular pulse, a binomial triangle shaped pulse, or unity white noise signal as defined by the user in the calling program.

VARIABLES

CHOICE : index of menu selection
COLUMNX : column index counter
DELTAT : sampling period in seconds
LEVEL : level of Pascal's triangle
NEXTPASCAL : next level of Pascal's triangle
NOISEQ : noise sequence
NSAMPLES : number of samples rounded to next higher power of 2
PASCAL : level of Pascal's triangle
PULSAMPLTD : input pulse amplitude (normalized -10 to 10)
PULSDURATN : length of input pulse in sample periods
SHAPE : = 1 for rectangular input pulse
= 2 for binomial triangle shaped input pulse

: = 3 for bipolar triangle shaped input pulse

TAU : deterministic vector input signal, or

stochastic matrix of vector input noise signals

TYPE : = 1 for deterministic input signal

= 2 for stochastic input signal

BEGIN

CLEAR SCREEN AND DISPLAY PROGRAM FUNCTION

ERASE

DISPLAY('GENERATING SYSTEM INPUT SIGNALS . . .')

SIGNAL GENERATION COMMENTS

GENERATE DETERMINISTIC SIGNAL

GENERATE RECTANGULAR SIGNAL

GENERATE SHAPED SIGNAL

SCALE PEAK MAGNITUDE OF DETERMINISTIC SIGNAL

ALTERNATE SIGNS FOR BIPOLAR TRIANGLE

GENERATE PSEUDO-RANDOM UNITY WHITE NOISE SEQUENCE SIGNAL

IF TYPE = 1, ...

IF SHAPE = 1, ...

TAU = [PULSAMPLTD*ONES(1,PULSDURATN)]; ...

ELSE PASCAL = 1; ...

LEVEL = 1; ...

WHILE LEVEL < PULSDURATN, ...

NEXTPASCAL = [1 0*ONES(1,LEVEL-1) 1]; ...

COLUMNIDX = 2; ...

WHILE COLUMNIDX < LEVEL+1, ...
NEXTPASCAL(1,COLUMNDX) = PASCAL(1,COLUMNDX-1)+...
PASCAL(1,COLUMNDX);...

COLUMNDX = COLUMNDX+1;...
END,...
PASCAL = NEXTPASCAL;...
LEVEL = LEVEL +1;...
END,...
TAU = PULSAMPLTD*[PASCAL/MAX(PASCAL)]';...
IF SHAPE = 3,...
FOR COLUMNDX = 1:PULSDURATN,...

   TAU(COLUMNDX,1)=TAU(COLUMNDX,1)*(-1)**(COLUMNDX-1);...

END;...
END;...
END,...

TAU = [TAU; 0*ONES(NSAMPLES-PULSDURATN, 1)];...
ELSE RAND('NORMAL'),...

   TAU = RAND(NSAMPLES, 1);...

END

//
// CLEAR UNNECESSARY VARIABLES -----------------------------------------------

CLEAR COLUMNDX
CLEAR LEVEL NEXTPASCAL NOISEQ PASCAL PULSAMPLTD PULSDURATN SHAPE
ERASE
// END OF GENINPUT PROGRAM ==================================================

RETURN
PROGRAM

NAME: DTRMNRS

AUTHOR: T T HAMMOND

DATE: 9 APR 1988

PURPOSE:

This file is a MatrixX executable command file. Its purpose is to compute the time and frequency responses for the experimental system simulation due to deterministic inputs. Also the desired frequency response function is extracted from the closed-loop configuration.

The system description variables and input signal must be provided by the calling program.

VARIABLES

DELTAT : sampling period in seconds

FHZ : frequency in Hz

G5FEST : estimated disk 3 velocity frequency response function

K : compensator gain

MODES : modal responses

NCL : order of closed-loop system description

NOISEQ : noise sequence

NSAMPLES : number of samples rounded to next higher power of 2

NSIM : discrete simulation index vector
// SCL : closed-loop system MatrixX description
// TAU : deterministic vector input signal, or
//       stochastic matrix of vector input noise signals
// TIME : time in seconds
// VEL1CL : disk 1 closed-loop velocity
// VEL1CLF : disk 1 closed-loop velocity frequency spectrum
// VEL3CL : disk 3 closed-loop velocity
// VEL3CLF : disk 3 closed-loop velocity frequency spectrum

// BEGIN
CLEAR SCREEN AND DISPLAY PROGRAM FUNCTION

ERASE
DISPLAY('COMPUTING SYSTEM RESPONSES ...')

GENERATE TIME RESPONSES

[NSIM, MODES] = DLSIM(SCL, NCL, TAU);
TIME = NSIM*DELTAT;
CLEAR NSIM

DISPLAY INPUT SIGNAL AND DISK-3 RIGID BODY MODES

PLOT(TIME, [TAU MODES(:,[1 2]),...

'TITL/DETERMINISTIC INPUT AND DISK-3 RIGID BODY MODES/...

YLAB/MAGNITUDE/...

XLAB/TIME IN SECONDS/')
PAUSE
ERASE

DISPLAY DISK-3 LOW FREQUENCY MODES

PLOT(TIME, [TAU MODES(:,[3 4]),...

'TITL/DETERMINISTIC INPUT AND DISK-3 0.1 Hz MODES/...
YLAB/MAGNITUDE/...
XLAB/TIME IN SECONDS/

PAUSE
ERASE

// DISPLAY DISK-3 HIGH FREQUENCY MODES -------------------------------------

PLOT(TIME, [TAU MODES(:,[5 6])],
'TITL/DETERMINISTIC INPUT AND DISK-3 1.5 HZ MODES/...
YLAB/MAGNITUDE/...
XLAB/TIME IN SECONDS/

PAUSE
ERASE

CLEAR TIME

// next section added to find max values for normalization==========

//P3L=MAX(A8S(MODES(:,3)))
//PAUSE

//P3H=MAX(A8S(MODES(:,5)))
//PAUSE

//V3R=MAX(A8S(MODES(:,2)))
//PAUSE

//V3L=MAX(A8S(MODES(:,4)))
//PAUSE

//V3H=MAX(A8S(MODES(:,6)))
//PAUSE

//A3L=MAX(A8S(-0.335*TAU-0.395*MODES(:,3)))
//PAUSE

//A3H=MAX(A8S(0.0015*TAU-88.8269*MODES(:,5)))
//PAUSE
MODES = [MODES(:,2) MODES(:,4) MODES(:,6)];

VEL1CL = [MODES(:,1)-1.9909*MODES(:,2)+3.4905E-3*MODES(:,3)];
VEL3CL = [MODES(:,1)+MODES(:,2)+MODES(:,3)];

V1CL=MAX(ABS(VEL1CL))

V3CL=MAX(ABS(VEL3CL))

COMPUTE DISK-3 VELOCITY OPEN-LOOP FREQUENCY RESPONSE FUNCTION -------
G5FEST = [VEL3CL/(TAUF+K*VEL1CL)];

FHZ = [1:(NSAMPLES/2)-1]/(NSAMPLES*DELTAT);
// DISPLAY INPUT SIGNAL SPECTRUM --------------------------------------

PLOT(FHZ, 20*LOG(ABS(TAUF))/LOG(10),
     'TITL/DETERMINISTIC INPUT FREQUENCY SPECTRUM/... 
     YLAB/db/...
     XLAB/FREQUENCY IN HZ/...
     LOGX')
PAUSE
ERASE

// DISPLAY DISK-1 CLOSED-LOOP VELOCITY SPECTRUM ------------------------

PLOT(FHZ, 20*LOG(ABS(VEL1CLF))/LOG(10),
     'TITL/DISK-1 CLOSED-LOOP VELOCITY FREQUENCY SPECTRUM/... 
     YLAB/db/...
     XLAB/FREQUENCY IN HZ/...
     LOGX')
PAUSE
ERASE

// DISPLAY DISK-3 CLOSED-LOOP VELOCITY SPECTRUM ------------------------

PLOT(FHZ, 20*LOG(ABS(VEL3CLF))/LOG(10),
     'TITL/DISK-3 CLOSED-LOOP VELOCITY FREQUENCY SPECTRUM/... 
     YLAB/db/...
     XLAB/FREQUENCY IN HZ/...
     LOGX')
PAUSE
ERASE

// DISPLAY DISK-3 OPEN-LOOP VELOCITY FREQUENCY RESPONSE FUNCTION ------

PLOT(FHZ, 20*LOG(ABS(G5FEST))/LOG(10),
     'TITL/DISK-3 OPEN-LOOP VELOCITY FREQUENCY RESPONSE FUNCTION/...
YLAB/db/

XLAB/FREQUENCY IN Hz/

LOGX'

PAUSE

ERASE

// CLEAR UNNECESSARY VARIABLES

CLEAR DELTAT FHZ G5FEST K NCL NSAMPLES SCL TAU TAUF

CLEAR TYPE VEL1CL VEL1CLF VEL3CL VEL3CLF

// END OF DTRMNRESP PROGRAM

RETURN
PROGRAM

NAME: STOCHRSP

AUTHOR: T T HAMMOND

DATE: 10 APR 1988

PURPOSE:
This file is a MatrixX executable command file. Its purpose is to compute the time and frequency responses for the experimental system simulation due to stochastic inputs. Also the desired frequency response function is extracted from the closed-loop configuration.
The system description variables and input signal must be provided by the calling program.

VARIABLES

DELTAT : sampling period in seconds
PHZ : frequency in Hz
G5FEST : estimated disk 3 velocity frequency response function
K : compensator gain
MODES : modal responses
NCL : order of closed-loop system description
NOISEQ : noise sequence
NSAMPLES : number of samples rounded to next higher power of 2
NSIM : discrete simulation index vector
// SCL : closed-loop system MatrixX description
// TAU : deterministic vector input signal, or
//       stochastic matrix of vector input noise signals
// TIME : time in seconds
// VEL1CL : disk 1 closed-loop velocity
// VEL1CLF : disk 1 closed-loop velocity frequency spectrum
// VEL3CL : disk 3 closed-loop velocity
// VEL3CLF : disk 3 closed-loop velocity frequency spectrum

// BEGIN
// CLEAR SCREEN AND DISPLAY PROGRAM FUNCTION
ERASE
DISPLAY('COMPUTING SYSTEM RESPONSES . . .')
// GENERATE TIME RESPONSES
[NSIM, MODES] = DLSIM(SCL, NCL, TAU);
TIME = NSIM*DELTAT;
CLEAR NSIM
// DISPLAY INPUT SIGNAL AND DISK-3 RIGID BODY MODES
PLOT(TIME, [TAU MODES(:,[1 2])], ...
    'TITL/STOCHASTIC INPUT AND DISK-3 RIGID BODY MODES/...
    YLAB/MAGNITUDE/...
    XLAB/TIME IN SECONDS/')
PAUSE
ERASE
// DISPLAY DISK-3 LOW FREQUENCY MODES
PLOT(TIME, [TAU MODES(:,[3 4])], ...
    'TITL/STOCHASTIC INPUT AND DISK-3 0.1 HZ MODES/...
YLAB/MAGNITUDE/...

XLAB/TIME IN SECONDS/

PAUSE

ERASE

// DISPLAY DISK-3 HIGH FREQUENCY MODES ---------------------------------------

PLOT(TIME, [TAU MODES(:,[5 6])],...
  'TITL/STOCHASTIC INPUT AND DISK-3 1.5 HZ MODES/...
  YLAB/MAGNITUDE/...
  XLAB/TIME IN SECONDS/

PAUSE

ERASE

CLEAR TIME

//next section added to find max values for normalization======================

//P3L=MAX(ABS(MODES(:,3)))
  //PAUSE

//P3H=MAX(ABS(MODES(:,5)))
  //PAUSE

//V3R=MAX(ABS(MODES(:,2)))
  //PAUSE

//V3L=MAX(ABS(MODES(:,4)))
  //PAUSE

//V3H=MAX(ABS(MODES(:,6)))
  //PAUSE

//A3L=MAX(ABS(-0.335*TAU-0.395*MODES(:,3)))
  //PAUSE

//A3H=MAX(ABS(0.0015*TAU-88.8269*MODES(:,5)))
  //PAUSE
//end first max values section

// DELETE POSITION RESPONSES FOR MEMORY CONSERVATION

MODES = [MODES(:,2) MODES(:,4) MODES(:,6)];

// FORM DISK-1 AND DISK-3 VELOCITIES

VEL1CL = [MODES(:,1)-1.9909*MODES(:,2)-3.4905E-3*MODES(:,3)];
VEL3CL = [MODES(:,1)+MODES(:,2)+MODES(:,3)];

//second max values section begins

// V1CL=MAX(ABS(VEL1CL))
// PAUSE

// V3CL=MAX(ABS(VEL3CL))
// PAUSE

//end second max values section

// CLEAR TIME RESPONSES FOR MEMORY CONSERVATION

CLEAR MODES

// FORM FEEDBACK ERROR SIGNAL

U = TAU + K*VEL1CL;

// COMPUTE AUTO-PSD OF INPUT SIGNAL

[TAUF, FIN] = SPECTRUM(TAU, TAU);
CLEAR TAU

TAUF = [TAUF([(NSAMPLES/2+2):(NSAMPLES)])];

// COMPUTE AUTO-PSD OF DISK-1 VELOCITY

[VEL1CLF, FIN] = SPECTRUM(VEL1CL, VEL1CL);
CLEAR VEL1CL

VEL1CLF = [VEL1CLF([(NSAMPLES/2+2):(NSAMPLES)])];

// COMPUTE CROSS-PSD OF ERROR SIGNAL AND DISK-3 VELOCITY

[UVEL3CLF, FIN] = SPECTRUM(VEL3CL, U);
CLEAR VEL3CL U
UVEL3CLF = [UVEL3CLF([(NSAMPLES/2+2):(NSAMPLES)])];

// COMPUTE DISK-3 OPEN-LOOP VELOCITY FREQUENCY RESPONSE FUNCTION -------
G5FEST = [UVEL3CLF./(TAUF+K*VEL1CLF)];

FHZ = FIN([(NSAMPLES/2+2):(NSAMPLES)])/(DELTAT);

// DISPLAY INPUT SIGNAL SPECTRUM -------------------------------------

PLOT(FHZ, 20*LOG(ABS(TAUF))/LOG(10), ...
      'TITL/STOCHASTIC INPUT AUTO-PSD/...
      YLAB/dB/...
      XLAB/FREQUENCY IN HZ/...
      LOGX')

PAUSE

ERASE

// DISPLAY DISK-1 CLOSED-LOOP VELOCITY SPECTRUM----------------------

PLOT(FHZ, 20*LOG(ABS(VEL1CLF))/LOG(10), ...
      'TITL/DISK-1 CLOSED-LOOP VELOCITY AUTO-PSD/...
      YLAB/dB/...
      XLAB/FREQUENCY IN HZ/...
      LOGX')

PAUSE

ERASE

// DISPLAY DISK-3 CROSS-PSD ------------------------------------------

PLOT(FHZ, 20*LOG(ABS(UVEL3CLF))/LOG(10), ...
      'TITL/ERROR SIGNAL AND DISK-3 CLOSED-LOOP VELOCITY CROSS-PSD/...
      YLAB/dB/...
      XLAB/FREQUENCY IN HZ/...
      LOGX')

PAUSE
ERASE

// DISPLAY DISK-3 OPEN-LOOP VELOCITY FREQUENCY RESPONSE FUNCTION -------

PLOT(FHZ, 20*LOG(ABS(G5FEST))/LOG(10),
     'TITL/DISK-3 OPEN-LOOP VELOCITY FREQUENCY RESPONSE FUNCTION/... 
     YLAB/dB/...
     XLAB/FREQUENCY IN HZ/...
     LOGX')

PAUSE

ERASE

// CLEAR UNNECESSARY VARIABLES ----------------------------------------

CLEAR DELTAT FHZ G5FEST K NCL NSAMPLES SCL TAU TAUF
CLEAR TYPE VEL1CL VEL1CLF VEL3CL VEL3CLF

// END OF STOCHRSP PROGRAM =============================================

RETURN
APPENDIX B

Hybrid Simulation Source Code
This appendix contains the source code listings for the hybrid simulation of Chapter Five. The programs are written in HP BASIC for use on a HP Model 9000 computer.

The programs listed perform three major tasks. The SIMULATION program drives the hybrid simulation to collect the time data for the identification process. The SIUDI (System Identification Using Deterministic Inputs) program performs frequency analysis and response function computation for deterministic analysis. The SIUSI (System Identification Using Stochastic Inputs) program provides the corresponding functions for stochastic analysis.
10 PROGRAM
20 ! Name: SIMULATION
30 ! Author: T T HAMMOND
40 ! Date: 1 MAY 1988
50 ! Purpose: This program drives the three-disk sampled-data
60 ! control system simulation. First the HP-3852A Data Acquisi-
70 ! tion/Control Unit is readied by initializing the HP-44702A
80 ! High Speed Voltmeter and HP-44727A Digital to Analog Voltage
90 ! Converter, and downloading the data acquisition/control
100 ! routine. Next the user is queried for the input signal file-
110 ! name and the record numbering sequence for the archive of time
120 ! records. At this point, the Data Acquisition/Control Unit is
130 ! commanded to run the simulation, and this program concurrently
140 ! displays a strip chart view of the input signal and the
150 ! measured velocities of disk one and disk three. During the
160 ! simulation, the user is prompted to operate the analog
170 ! computer and confirm these actions. Finally, the data col-
180 ! lected is stored on diskette in the appropriate archive, and
190 ! the program recycles to begin another simulation if the user
200 ! desires.
210
220 ! SUBPROGRAMS
230 ! The following subprograms from the HP Data Acquisition Manager
240 ! DACQ/300 package are used:
250 ! Get_array(Vectorname$,Array(*))
260 ! Vect_segment(Vectorname$,Start,Number,Result$)
270 ! Strip_init(Graph_rec(*),Num_traces,X_min,X_max,X_sample,}
280 ! Y_min,Y_max,[,Buf_size[,Strip_buf$]]
290 ! Graph_window(Graph_rec(*),X_start[,Y_start[,X_range
300 ! [,Y_range]])
310 ! Graph_autoscale(Graph_rec(*)[,Option$])
320 ! Graph_labels(Graph_rec(*),Title$,:Subtitle$,X_label$,,
330 ! Y_label$,Trace_label$)
340 ! Graph_show(Graph_rec(*)[Trace_num$])
350 ! Strip_points(Graph_rec(*),X_val,Trace_num$,Y_val_a[,Y_val_b
360 ! [,Y_val_c[,Y_val_d]])
370 ! Save_nvect(Input(*),Vectorname$)
380 ! Arch_open(,Archivename$,Archivesize[,Last_pagesize$])
390 ! Page_store(,Inst_addr[,Page_label$,,
400 ! [Destination$]])
410 ! Arch_close
420 !
430 ! VARIABLES ==---------------------------------------------------------------------
440 ! Vel1,Vel3 : disk-1, disk-3 velocities in radians/sec
450 ! Norm_volt : analog computer normalizing voltage in volts
460 ! Scandata : raw disk-1,disk-3 voltmeter readings in volts
470 ! Insignal : input signal sequence in radians/sec
480 ! Strip_time : strip chart abscissa in seconds
490 ! Sample_count : sample number index
500 ! First_rec : first record number to append to archive
510 ! Last_rec : last record number to append to archive
520 ! Record_count : record number index
530 ! Strip_rec : strip chart graph record
540 | Strip_count : strip chart sample number index
550 | Sampling = 1 while sampling data
550 | = 0 otherwise
570 | Insgn1$: name of input signal vector
580 | More$: query response
590 | Title$: strip chart title
600 | Sub_title$: strip chart subtitle
610 | Trace_labels$: strip chart trace labels
620 | X_label$: strip chart abscissa label
630 | Y_label$: strip chart ordinate label
640 |
650 | REAL Vel1(0:1023), Vel3(0:1023), Norm_volt
660 | REAL Scandata(0:1)
670 | REAL Insignal(0:1023)
680 | REAL Strip_time
690 | INTEGER Sample_count, First_rec, Last_rec, Record_count
700 | INTEGER Strip_rec(1:800), Strip_count, Sampling
710 | DIM Insgn1$[101], More$[3]
720 | DIM Title$[40], Sub_title$[40], Trace_labels$[40]
730 | DIM X_label$[20], Y_label$[20]
740 | ! CONSTANTS ===================================================
750 | ! Normvel1 : disk-1 velocity normalizing constant
760 | ! Normvel3 : disk-3 velocity normalizing constant
770 | ! Normsgn1 : input signal normalizing constant
780 |
790 | REAL Normvel1, Normvel3, Normsgn1
800 | Normvel1 = 5
160  Norvel3=5
820  Normsgnl=10
830  ! BEGIN ---------------------------------------------------------
840  ! INITIALIZE DATA ACQUISITION SUBPROGRAMS ---------------------
850  Sys_init
860  ! CONFIGURE HP-3852A HARDWARE ---------------------------------
870  OUTPUT 709; "RST"
880  OUTPUT 709; "USE 200"
890  OUTPUT 709; "CONF DCV"
900  OUTPUT 709; "PACER 0.200, 1024"
910  OUTPUT 709; "PDELAY 0.200"
920  OUTPUT 709; "SWRITE 100, 2, 13"
930  OUTPUT 709; "SWRITE 100, 6, 00, 00"
940  ! UNMASK HP-3852A SOFTWARE INTERRUPT --------------------------
950  OUTPUT 709; "RQS FPS"
960  ! DECLARE HP-3852A VARIABLES -----------------------------------
970  OUTPUT 709; "REAL CNTRLERR, INSIGNAL(1023)"
980  OUTPUT 709; "REAL SGNLSCAL, VEL(1023), VEL3(1023)"
990  OUTPUT 709; "REAL SCANDATA(1), FBGAIN"
1000 OUTPUT 709; "INTEGER COUNT"
1010  ! SET FEEDBACK GAIN -------------------------------------------
1020  OUTPUT 709; "FBGAIN = 1"
1030  ! HP-3852A DATA ACQUISITION AND CONTROL ROUTINE ---------------
1040  OUTPUT 709; "SUB CNTRLLOOP"
1050  OUTPUT 709; "FOR COUNT=0 TO 1023"
1060  OUTPUT 709; "CNTRLERR=0.667*SGNLSCAL*INSIGNAL(COUNT)-0.333*FBGAIN*S
CANDATA(0)"
1070 OUTPUT 709; "INDEX SCANDATA 0"
1080 OUTPUT 709; "WAITFOR PACER"
1090 OUTPUT 709; "MEAS DCV, 401-402 INTO SCANDATA"
1100 OUTPUT 709; "APPLY DCV 100, CNTRLERR"
1110 OUTPUT 709; "VEL1(COUNT) = SCANDATA(0)"
1120 OUTPUT 709; "VEL3(COUNT) = SCANDATA(1)"
1130 OUTPUT 709; "IF COUNT < 100 THEN"
1140 OUTPUT 709; "VREAD SCANDATA"
1150 OUTPUT 709; "END IF"
1160 OUTPUT 709; "NEXT COUNT"
1170 OUTPUT 709; "SRQ"
1180 OUTPUT 709; "SUBEND"
1190 ! DEFINE INTERRUPT SERVICE ROUTINE -------------------------------
1200 ON INTR 7 GOSUB Service_routine
1210 ! LOOP WHILE MORE SIGNALS TO BE APPLIED ------------------------
1220 More$[1,1] = "Y"
1230 WHILE More$[1,1] = "Y"
1240 ! GET USER SUPPLIED INFORMATION ---------------------------------
1250 BEEP
1260 DISP "ENTER NAME OF INPUT SIGNAL VECTOR, e.g. RTX1_VEC";
1270 LINPUT "", Insgnl$
1280 BEEP
1290 DISP "ENTER FIRST RECORD NUMBER";
1300 INPUT "", First_rec
1310 BEEP
1320 DISP "ENTER LAST RECORD NUMBER";
INPUT "", Last_rec

IF Insgnl$<>"NOISE_REC" THEN
    Get_rarray(Insgnl$, Insignal(*)
    MAT Insignal= Insignal/(Normsignal)
END IF

LOOP FOR EACH RECORD -----------------------------------------

PARSE NOISE SIGNAL -------------------------------------------

IF Insgnl$="NOISE_REC" THEN
    Get_rarray("NOISE_VEC", Insignal(*))
    MAT Insignal= Insignal/(Normsignal)
END IF

INITIALIZE STRIP CHART ---------------------------------------

Strip_init(Strip_rec(*),3,0,20,1,-10,10)
Graph_window(Strip_rec(*),0,-10,20,20)
Graph_autoscale(Strip_rec(*))
Title$="Three-Disk Simulation"
Sub_title$="Sampled-data Strip Chart"
X_label$="Time in seconds"
Y_label$="Radians per second"
Trace_labels$="Input!Vel-1!Vel-3"
Graph_labels(Strip_rec(*),Title$,Sub_title$,X_label$,Y_label$, Trace_labels$)

OUTPUT KBD:CHR$(255)&"K":
GRAPHICS ON

Graph_show(Strip_rec(*))

! INITIALIZE HP-3852A VARIABLES ------------------------------------

OUTPUT 709;"DISP OFF"
OUTPUT 709;"SCANDATA(0)=0"
OUTPUT 709;"INDEX SCANDATA, 0"
OUTPUT 709;"INDEX INSIGNAL, 0"
FOR Sample_count=0 TO 1023
OUTPUT 709;"WRITE INSIGNAL "$VAL$(Insignal(Sample_count))"
NEXT Sample_count

! ENABLE INTERRUPT FOR HP-3852A FINISH -----------------------------
Sampling=1
ENABLE INTR 7;2

! PROMPT USER TO ACTIVATE ANALOG COMPUTER --------------------------
BEEP
BEEP
DISP "ACTIVATE ANALOG COMPUTER AND PRESS RETURN . . ."
LINPUT "",Null$

! DISPLAY NUMBER OF RECORD BEING RECORDED --------------------------
DISP "RECORDING RECORD NUMBER ":Record_count

! MEASURE ANALOG COMPUTER NORMALIZING VOLTAGE ----------------------
OUTPUT 709;"MEAS DCV, 403 INTO SGNLSCAL"
OUTPUT 709;"VREAD SGNLSCAL"
ENTER 709;Norm_volt

! START HP-3852A ROUTINE -------------------------------------------
OUTPUT 709;"PTRIG"
OUTPUT 709;"CALL CNTRL"
ADD DATA POINTS TO STRIP CHART FOR 100 SAMPLES

FOR Strip_count=0 TO 99
ENTER 709;Scandata(*)
Strip_time=.2*Strip_count
Strip_points(Strip_rec(*),Strip_time,"11213",Normv1*Insignal(Strip_count),Normv1*Scandata(0),Normv1*Scandata(1))
NEXT Strip_count

WAIT UNTIL HP-3852A FINISHES AND INTERRUPTS

WHILE Sampling=1
END WHILE

PROMPT USER TO RESET ANALOG COMPUTER

BEEP
BEEP
BEEP

DISP "SAMPLING COMPLETED: RESET ANALOG COMPUTER"

GET RAW VELOCITY MEASUREMENTS FROM HP-3852A

OUTPUT 709;"VREAD VEL1"
ENTER 709;Vel1(*)
OUTPUT 709;"VREAD VEL3"
ENTER 709;Vel3(*)

CONVERT VELOCITIES TO RADIANS/SEC

MAT Vel1= (Normv1)*Vel1
MAT Vel1= Vel1/(Norm_volt)
MAT Vel3= (Normv3)*Vel3
MAT Vel3= Vel3/(Norm_volt)

STORE VELOCITIES IN ARCHIVE
2090 Save_rvect(VEl(*),"DISK_1VEL")
2100 Save_rvect(VEl3(*),"DISK_3VEL")
2110 Arch_open(Insgni$[1,7]&"ARC")
2120 Page_store(Insgni$[1,7]&"REC",709,VAL$(Record_count))
2130 Arch_close
2140 ! ACTIVATE HP-3852A DISPLAY -----------------------------------------------
2150 OUTPUT 709;"DISP ON"
2160 ! END OF LOOP FOR EACH RECORD ----------------------------------------------
2170 NEXT Record_count
2180 ! QUERY USER FOR MORE SIGNALS TO BE APPLIED -------------------------------
2190 BEEP
2200 DISP "DO YOU WISH TO DO ANOTHER INPUT SIGNAL? (Y or N)";
2210 LINPUT ",More$
2220 ! END OF LOOP FOR MORE SIGNALS --------------------------------------------
2230 END WHILE
2240 STOP
2250 ! HP-3852A INTERRUPT SERVICE ROUTINE --------------------------------------
2260 Service_routine: !
2270 Ser_poll=SPOLL(709)
2280 OUTPUT 709;"STA?"
2290 ENTER 709;Stat_word
2300 Sampling=0
2310 RETURN
2320 ! END OF HP-3852A INTERRUPT SERVICE ROUTINE -------------------------------
2330 ! END OF SIMULATION PROGRAM -------------------------------------------------
2340 END
10 Procedure Name: SIUDT (System Identification Using Deterministic Input)
20 Author: T T HAMMOND
30 Date: May 1988
40 Purpose: This program computes and averages the sampled-data
50 frequency response function for disk-3 velocity for deterministic excitations. First a random disk is established for computational efficiency. Next the user is queried for input
60 filename and number of records to average. Then the FFT of
70 the input signal is performed. Now FFTs for the disk-1 and
80 disk-3 velocities for each record to be averaged are performed
90 and summed. Finally the average disk-3 open-loop frequency
100 response function is calculated and stored on diskette.

150 Subprograms
160 The following subprograms from the HP Data Acquisition Manager
170 D/AQ/300 package are used:
180 Save_rvect(Input(*),Vectorname$)
190 Vect_fft(Type$,Vectorname$,Results[$,Window$])
200 Arch_query(Archive_name$,Book_name$,Page_label$,Field_name$,
210 Record$,Results$)
220 Get_array(Vectorname$,Array(*))
230
240 The following subprograms appear at the end of this program:
250 FNMag_db(Real_part,Imag_part,Nr_samples)
260 FNPPhase_deg(Real_part,Imag_part)
280 ! VARIABLES

290 ! Sample_rec$ : input signal filename

300 ! Nr_avg$ : number of records in average

310 ! Nr_records : number of records to average

320 ! All_zeros : array of zeros

330 ! Frequencies : FFT frequency array

340 ! Disk_!real,

350 ! Disk_!imag : real and imaginary parts of FFT for velocity !

360 . ! Disk_3real,

370 ! Disk_3imag : real and imaginary parts of FFT for velocity 3

380 ! Sum_!real,

390 ! Sum_!imag : real and imaginary sums of FFTs for velocity !

400 ! Sum_3real,

410 ! Sum_3imag : real and imaginary sums of FFTs for velocity 3

420 ! Input_real,

430 ! Input_imag : real and imaginary parts of FFT for input

440 ! Denom_real,

450 ! Denom_imag : real and imaginary parts of denominator

460 ! Denom_mag : denominator magnitude in dB

470 ! Denom_faz : denominator phase in degrees

480 ! Numer_mag : numerator magnitude in dB

490 ! Numer_faz : numerator phase in degrees

500 ! Xferfn_mag : frequency response function magnitude in dB

510 ! Xferfn_faz : frequency response function phase in degrees

520 !

530 DIM Sample_rec$[10], Nr_avg$[2]
REAL Nr_records

! CONSTANTS

! Nr_samples : number of samples
! Nr_sectors : number of ram disk sectors
! Sample_per : sample period in seconds

INTEGER Nr_samples, Nr_sectors
REAL Sample_per

Nr_sectors=330
Nr_samples=1024
Sample_per=.2

BEGIN

INITIALIZE DATA ACQUISITION SUBPROGRAMS

SET UP RAM DISK FOR EFFICIENT EXECUTION

INITIALIZE ":MEMORY,0,14", Nr_sectors

QUERY USER FOR INPUT SIGNAL NAME AND NUMBER OF RECORDS TO AVERAGE

BEEP

DISP "ENTER NAME OF INPUT SIGNAL VECTOR USED FOR SYSTEM EXCITATION"

INPUT "", Sample_rec$

BEEP

DISP "ENTER NUMBER OF RECORDS TO INCLUDE IN AVERAGE"

INPUT "", Nr_records
169

800  !  GENERATE ZERO VECTOR AND FREQUENCY VECTOR ------------------------
810  ALLOCATE REAL All.zeros(1:Nr.samples), Frequencies(1:Nr.samples)
820  FOR I=1 TO Nr.samples
830    All.zeros(I)=0
840    Frequencies(I)=(I-1)/(Sample_per*Nr_samples)
850  NEXT I
860  Save_rvect(All.zeros(*), "ALL_zeros:MEMORY, 0, 14")
870  Save_rvect(Frequencies(*), "FREQ_VEC:700, 1")
880  DEALLOCATE Frequencies(*)
890  !  PERFORM FFT ON INPUT SIGNAL -------------------------------------
900  !  USE RECTANGULAR WINDOW ------------------------------------------
910  Vec_fft("FFT", Sample_rec$&":700, 1; ALL_zeros:MEMORY, 0, 14", "INPUT_"
        "REAL:MEMORY, 0, 14; INPUT_imag:MEMORY, 0, 14", "N")
920  !  SET UP LOOP TO PROCESS NUMBER OF RECORDS SPECIFIED ----------------
930  ALLOCATE REAL Sum_1real(1:Nr.samples), Sum_1imag(1:Nr.samples)
940  ALLOCATE REAL Sum_3real(1:Nr.samples), Sum_3imag(1:Nr.samples)
950  ALLOCATE REAL Disk_1real(1:Nr.samples), Disk_1imag(1:Nr.samples)
960  ALLOCATE REAL Disk_3real(1:Nr.samples), Disk_3imag(1:Nr.samples)
970  MAT Sum_1real= All.zeros
980  MAT Sum_1imag= All.zeros
990  MAT Sum_3real= All.zeros
1000 MAT Sum_3imag= All.zeros
1010 FOR J=1 TO Nr_records
1020  !  RECALL SAMPLED-DATA FOR DISK-1, DISK-3 VELOCITIES FROM ARCHIVE
1030  !  -----------------------------------------------------------------
1040  Arch_query(Sample_rec$[1, 7] & "ARC:700, 1", Sample_rec$[1, 7] & "REC",
VALS(J),"*" ,"* " , "DISK_1VEL:MEMORY,0, 14; DISK_3VEL:MEMORY,0, 14")
1050 ! PERFORM FFT ON DISK-1 CLOSED-LOOP VELOCITY SIGNAL ------------
1060 ! USE RECTANGULAR WINDOW -------------------------------------
1070 vect_fft("FFT", "DISK_1VEL:MEMORY,0, 14; ALL_ZEROS:MEMORY,0, 14", "DISK_1REAL:MEMORY,0, 14; DISK_1IMAG:MEMORY,0, 14", "N")
1080 ! PERFORM FFT ON DISK-3 CLOSED-LOOP VELOCITY SIGNAL ------------
1090 ! USE RECTANGULAR WINDOW -------------------------------------
1100 vect_fft("FFT", "DISK_3VEL:MEMORY,0, 14; ALL_ZEROS:MEMORY,0, 14", "DISK_3REAL:MEMORY,0, 14; DISK_3IMAG:MEMORY,0, 14", "N")
1110 ! ACCUMULATE SUM OF VELOCITY FFT'S -----------------------------
1120 Get_rarray("DISK_1REAL:MEMORY,0, 14", Disk_1real(*))
1130 Get_rarray("DISK_1IMAG:MEMORY,0, 14", Disk_1imag(*))
1140 Get_rarray("DISK_3REAL:MEMORY,0, 14", Disk_3real(*))
1150 Get_rarray("DISK_3IMAG:MEMORY,0, 14", Disk_3imag(*))
1160 MAT Sum_1real = Sum_1real+Disk_1real
1170 MAT Sum_1imag = Sum_1imag+Disk_1imag
1180 MAT Sum_3real = Sum_3real+Disk_3real
1190 MAT Sum_3imag = Sum_3imag+Disk_3imag
1200 NEXT J
1210 DEALLOCATE Disk_1real(*), Disk_1imag(*)
1220 DEALLOCATE Disk_3real(*), Disk_3imag(*)
1230 ! COMPUTE DISK-3 VELOCITY OPEN-LOOP FREQUENCY RESPONSE FUNCTION
1240 ! DENOMINATOR -----------------------------------------------
1250 ALLOCATE REAL Input_real(1:Nr_samples), Input_imag(1:Nr_samples)
1250 ALLOCATE REAL Denom_real(1:Nr_samples), Denom_imag(1:Nr_samples)
1270 ALLOCATE REAL Denom_mag(1:Nr_samples), Denom_faz(1:Nr_samples)
1280 Get_rarray("INPUT_REAL:MEMORY,0, 14", Input_real(*))
Get_array("INPUT_IMAGE:MEMORY,0,14",Input_image(*))

MAT Input_real= (Nr_records)*Input_real

MAT Input_imag= (Nr_records)*Input_imag

MAT Denom_real= Input_real-Sum_lreal

MAT Denom_imag= Input_imag-Sum_limag

FOR I=1 TO Nr_samples

Denom_mag(I)=FNMag_db((Denom_real(I)),(Denom_imag(I)),(Nr_samples))

Denom_faz(I)=FNPhase_deg((Denom_real(I)),(Denom_imag(I)))

NEXT I

DEALLOCATE Input_real(*),Input_imag(*)

DEALLOCATE Denom_real(*),Denom_imag(*)

DEALLOCATE Sum_lreal(*),Sum_limag(*)

! COMPUTE DISK-3 VELOCITY OPEN-LOOP FREQUENCY RESPONSE FUNCTION

allocate REAL Numer_mag(1:Nr_samples),Numer_faz(1:Nr_samples)

allocate REAL Xferfn_mag(1:Nr_samples),Xferfn_faz(1:Nr_samples)

FOR I=1 TO Nr_samples

 Numer_mag(I)=FNMag_db((Sum_3real(I)),(Sum_3imag(I)),(Nr_samples))

 Numer_faz(I)=FNPhase_deg((Sum_3real(I)),(Sum_3imag(I)))

 Xferfn_mag(I)=Numer_mag(I)-Denom_mag(I)

 Xferfn_faz(I)=Numer_faz(I)-Denom_faz(I)

IF ABS(Xferfn_faz(I))=180 THEN Xferfn_faz(I)=Xferfn_faz(I)-360*
SGN(Xferfn_faz(I))

NEXT I
1520 ! STORE DISK-3 OPEN-LOOP VELOCITY FREQUENCY RESPONSE FUNCTION

1530 ! DISKETTE -----------------------------------------------------

1540 IF Nr_records<10 THEN
1550 Nr_avg$=""&VAL$(Nr_records)
1560 ELSE
1570 Nr_avg$=VAL$(Nr_records)
1580 END IF

1610 END OF SIMULATION PROGRAM ====================================

1620 END

1630 SUBPROGRAMS ==================================================

1640 COMPUTE MAGNITUDE IN dB OF REAL AND IMAGINARY FFT RESULTS ----

1650 DEF FNMag_db(REAL Real_part,IMag_part,INTEGER Nr_samples)

1660 RETURN 20*LGT(Nr_samples*SQR(Real_part^2+IMag_part^2))

1670 FNEND

1700 DEF FNPhase_deg(REAL Real_part,IMag_part)

1710 SELECT SGN(Real_part*IMag_part)

1720 CASE -1,1

1730 IF Real_part>0 THEN

1740 RETURN 180*ATN(IMag_part/Real_part)/PI

1750 ELSE

1750 RETURN 180*(ATN(IMag_part/Real_part)/PI+SGN(IMag_part))

1770 END IF

1780 CASE 0
1790 IF Real_part<>0 THEN
1800 RETURN 180*(1-SGN(Real_part))
1810 ELSE
1820 RETURN 90*SGN(Imag_part)
1830 END IF
1840 END SELECT
1850 FNEND
1860 END OF SUBPROGRAMS
1870 END OF SUBPROGRAMS
PROGRAM ================

Name: SIUSI (System Identification Using Stochastic Input)

Author: T T HAMMOND

Date: 1 MAY 1988

Purpose: This program computes and averages the sampled-data frequency response function for disk-3 velocity for stochastic excitations. First a ram disk is established for computational efficiency. Next the user is queried for input signal filename and number of records to average. Then FFTs are performed on the input signal and disk-1 and disk-3 velocities for each record. Also the numerator and denominator of the average response function are formed. Finally the average disk-3 open-loop frequency response function is calculated and stored on diskette.

SUBPROGRAMS ================

The following subprograms from the HP Data Acquisition Manager DACQ/300 package are used:

Save_rvect(Input(*),Vectorname$)

Vect_segment(Vectorname$,Start,Number,Result$)

Vect_fft(Type$,Vectorname$,Result$(,Window$))

Arch_query(Archivename$,Bookname$,Pagelabel$,Fieldname$,Record$,Result$)

Get_rarray(Vectorname$,Array(*)

The following subprograms appear at the end of this program:

FNMag_db(Real_part,Imag_part,Nr_samples)
260 ! FNPhase_deg(Real_part,Imag_part)
290 ! FNWindow(Element_num,Total_num)
310 ! VARIABLES =========
320 ! Sample_rec$ : input signal filename
330 ! Nr_avg$ : number of records in average
340 ! Nr_records : number of records to average
350 ! All_zeros : array of zeros
360 ! Frequencies : FFT frequency array
370 ! Input_real,
380 ! Input_imag : real and imaginary parts of FFT for input
390 ! Disk-1real,
400 ! Disk-1imag : real and imaginary parts of FFT for velocity 1
410 ! Disk-3real,
420 ! Disk-3imag : real and imaginary parts of FFT for velocity 3
430 ! Error_real,
440 ! Error_imag : real and imaginary parts of error signal
450 ! Error_real_sq,
460 ! Error_imag_sq : squares of real and imaginary parts of error
470 ! Denom_sum_real: cumulative denominator of response function
480 ! Numer_sum_real,
490 ! Numer_sum_imag: cumulative numerator of response function
500 ! Product_1,
510 ! Product_2,
520 ! Product_3,
530 ! Product_4 : temporary product terms
540  ! Denom_mag   : denominator magnitude in dB
550  ! Numer_mag   : numerator magnitude in dB
560  ! Numer_faz   : numerator phase in degrees
570  ! Xferfn_mag  : frequency response function magnitude in dB
580  ! Xferfn_faz  : frequency response function phase in degrees
590  
600  DIM Sample_rec$[10],Nr_avg$[2]
610  REAL Nr_records
620  
630  ! CONSTANTS ---------------------------------------------------
640  ! Nr_samples  : number of samples
650  ! Nr_sectors  : number of sectors
660  ! Sample_per  : sample period in seconds
670  
680  INTEGER Nr_samples,Nr_sectors
690  REAL Sample_per
700  Nr_sectors=500
710  Nr_samples=1024
720  Sample_per=.2
730  
740  ! BEGIN -------------------------------------------------------
750  ! INITIALIZE DATA ACQUISITION SUBPROGRAMS -------------------
760  Sys_init
770  ! SET UP RAM DISK FOR EFFICIENT EXECUTION ---------------------
780  INITIALIZE ":MEMORY,2,"$4",Nr_sectors
790  ! QUERY USER FOR INPUT SIGNAL NAME AND NUMBER OF RECORDS TO ----
800  ! AVERAGE ------------------------------------------------------
177

810 BEEP
820 DISP "ENTER NAME OF INPUT SIGNAL VECTOR USED FOR SYSTEM EXCITATION"
830 LINPUT ",Sample_rec$
840 BEEP
850 DISP "ENTER NUMBER OF RECORDS TO INCLUDE IN AVERAGE"
860 INPUT ",Nr_records
870 ! GENERATE ZERO VECTOR AND FREQUENCY VECTOR
880 ALLOCATE REAL All_zeros(1:Nr_samples),Frequencies(1:Nr_samples)
890 FOR I=1 TO Nr_samples
900 All_zeros(I)=0
910 Frequencies(I)=(I-1)/(Sample_per*Nr_samples)
920 NEXT I
930 Save_rvec(All_zeros(\*),"ALL_ZEROS:MEMORY,0,14")
940 Save_rvec(Frequencies(\*),"FREQ_VECT:700,1")
950 DEALLOCATE Frequencies(\*)
960 ! SET UP LOOP TO PROCESS NUMBER OF RECORDS SPECIFIED
970 ALLOCATE REAL Input_real(1:Nr_samples),Input_imag(1:Nr_samples)
980 ALLOCATE REAL Disk_1real(1:Nr_samples),Disk_1imag(1:Nr_samples)
990 ALLOCATE REAL Disk_3real(1:Nr_samples),Disk_3imag(1:Nr_samples)
1000 ALLOCATE REAL Error_real(1:Nr_samples),Error_imag(1:Nr_samples)
1010 ALLOCATE REAL Error_real_sq(1:Nr_samples),Error_imag_sq(1:Nr_samples)
1020 ALLOCATE REAL Denon_sum_real(1:Nr_samples)
1030 ALLOCATE REAL Numer_sum_real(1:Nr_samples),Numer_sum_imag(1:Nr_samples)
1040  ALLOCATE REAL Product_1(1:Nr_samples),Product_2(1:Nr_samples)
1050  ALLOCATE REAL Product_3(1:Nr_samples),Product_4(1:Nr_samples)
1060  MAT Denom_sum_real= All_zeros
1070  MAT Numer_sum_real= All_zeros
1080  MAT Numer_sum_imag= All_zeros
1090  FOR J=1 TO Nr_records
1100  ! RECALL INPUT SIGNAL FOR JTH RECORD --------------------------
1110  Vect_segment(Sample_rec$[1,7]&"REC",((J-1)*Nr_samples+1),(Nr_samples),"INPUT_SGNL:MEMORY,0,14")
1120  ! PERFORM FFT ON INPUT SIGNAL ---------------------------------
1130  ! USE HANNING WINDOW ------------------------------------------
1140  Vect_fft("FFT","INPUT_SGNL:MEMORY,0,14!ALL_ZEROS:MEMORY,0,14","INPUT_REAL:MEMORY,0,14!INPUT_IMG:MEMORY,0,14","U")
1150  ! RECALL SAMPLED-DATA FOR DISK-1,DISK-3 VELOCITIES FROM ARCHIVE
1160  !----------------------------------------------------------------------------
1170  Arch_query(Sample_rec$[1,7]&"ARC:,700,1",Sample_rec$[1,7]&"REC", VAL$(J),"*","*","DISK_1VEL:MEMORY,0,14!DISK_3VEL:MEMORY,0,14")
1180  ! PERFORM FFT ON DISK-1 CLOSED-LOOP VELOCITY SIGNAL -------------
1190  ! USE HANNING WINDOW ------------------------------------------
1200  Vect_fft("FFT","DISK_1VEL:MEMORY,0,14!ALL_ZEROS:MEMORY,0,14","DISK_1REAL:MEMORY,0,14!DISK_1IMAG:MEMORY,0,14","U")
1210  ! PERFORM FFT ON DISK-3 CLOSED-LOOP VELOCITY SIGNAL ------------
1220  ! USE HANNING WINDOW ------------------------------------------
1230  Vect_fft("FFT","DISK_3VEL:MEMORY,0,14!ALL_ZEROS:MEMORY,0,14","DISK_3REAL:MEMORY,0,14!DISK_3IMAG:MEMORY,0,14","U")
1240  ! ACCUMULATE NUMERATOR AND DENOMINATOR OF RESPONSE FUNCTION ----
1250  ! SUMS ----------------------------------------------------------
Get_rarray("INPUT_REAL:MEMORY,0,14",Input_real(*))
Get_rarray("INPUT_IMAG:MEMORY,0,14",Input_imag(*))
Get_rarray("DISK_1REAL:MEMORY,0,14",Disk_1real(*))
Get_rarray("DISK_1IMAG:MEMORY,0,14",Disk_1imag(*))
Get_rarray("DISK_3REAL:MEMORY,0,14",Disk_3real(*))
Get_rarray("DISK_3IMAG:MEMORY,0,14",Disk_3imag(*))

MAT Error_real= Input_real-Disk_1real
MAT Error_imag= Input_imag-Disk_1imag
MAT Error_real_sq= Error_real . Error_real
MAT Error_imag_sq= Error_imag . Error_imag
MAT Denom_sum_real= Denom_sum_real+Error_real_sq
MAT Denom_sum_real= Denom_sum_real+Error_imag_sq
MAT Product_1= Error_real . Disk_3real
MAT Product_2= Error_imag . Disk_3imag
MAT Product_3= Error_real . Disk_3imag
MAT Product_4= Error_imag . Disk_3real
MAT Numer_sum_real= Numer_sum_real+Product_1
MAT Numer_sum_real= Numer_sum_real+Product_2
MAT Numer_sum_imag= Numer_sum_imag+Product_3
MAT Numer_sum_imag= Numer_sum_imag+Product_4

NEXT J
DEALLOCATE Input_real(*),Input_imag(*)
DEALLOCATE Disk_1real(*),Disk_1imag(*)
DEALLOCATE Disk_3real(*),Disk_3imag(*)
DEALLOCATE Error_real(*),Error_imag(*)
DEALLOCATE Error_real_sq(*),Error_imag_sq(*)

MAT Error_real_sq= Error_real . Error_real
MAT Error_imag_sq= Error_imag . Error_imag
MAT Denom_sum_real= Denom_sum_real+Error_real_sq
MAT Denom_sum_real= Denom_sum_real+Error_imag_sq
MAT Product_1= Error_real . Disk_3real
MAT Product_2= Error_imag . Disk_3imag
MAT Product_3= Error_real . Disk_3imag
MAT Product_4= Error_imag . Disk_3real
MAT Numer_sum_real= Numer_sum_real+Product_1
MAT Numer_sum_real= Numer_sum_real+Product_2
MAT Numer_sum_imag= Numer_sum_imag+Product_3
MAT Numer_sum_imag= Numer_sum_imag+Product_4

NEXT J
DEALLOCATE Input_real(*),Input_imag(*)
DEALLOCATE Disk_1real(*),Disk_1imag(*)
DEALLOCATE Disk_3real(*),Disk_3imag(*)
DEALLOCATE Error_real(*),Error_imag(*)
DEALLOCATE Error_real_sq(*),Error_imag_sq(*)

MAT Error_real_sq= Error_real . Error_real
MAT Error_imag_sq= Error_imag . Error_imag
MAT Denom_sum_real= Denom_sum_real+Error_real_sq
MAT Denom_sum_real= Denom_sum_real+Error_imag_sq
MAT Product_1= Error_real . Disk_3real
MAT Product_2= Error_imag . Disk_3imag
MAT Product_3= Error_real . Disk_3imag
MAT Product_4= Error_imag . Disk_3real
MAT Numer_sum_real= Numer_sum_real+Product_1
MAT Numer_sum_real= Numer_sum_real+Product_2
MAT Numer_sum_imag= Numer_sum_imag+Product_3
MAT Numer_sum_imag= Numer_sum_imag+Product_4

NEXT J
DEALLOCATE Input_real(*),Input_imag(*)
DEALLOCATE Disk_1real(*),Disk_1imag(*)
DEALLOCATE Disk_3real(*),Disk_3imag(*)
DEALLOCATE Error_real(*),Error_imag(*)
DEALLOCATE Error_real_sq(*),Error_imag_sq(*)
1520 DEALLOCATE Product_1(*), Product_2(*)
1530 DEALLOCATE Product_3(*), Product_4(*)
1540 ! COMPUTE DISK-3 VELOCITY OPEN-LOOP FREQUENCY RESPONSE FUNCTION
1550 ! -----------------------------------------------
1560 ALLOCATE REAL Denom_mag(1:Nr_samples)
1570 FOR I=1 TO Nr_samples
1580 Denom_mag(I)=FMagDb((Denom_sum_real(I)),(All_zeros(I)),(Nr_samples))
1590 NEXT I
1600 DEALLOCATE Denom_sum_real(*)
1610 ALLOCATE REAL Numer_mag(1:Nr_samples), Numer_faz(1:Nr_samples)
1620 ALLOCATE REAL Xferfn_mag(1:Nr_samples), Xferfn_faz(1:Nr_samples)
1630 FOR I=1 TO Nr_samples
1640 Numer_mag(I)=FMagDb((Numer_sum_real(I)),(Numer_sum_imag(I)),(Nr_samples))
1650 Numer_faz(I)=FNPhase_deg((Numer_sum_real(I)),(Numer_sum_imag(I)))
1660 Xferfn_mag(I)=Numer_mag(I)-Denom_mag(I)
1670 Xferfn_faz(I)=Numer_faz(I)
1680 IF ABS(Xferfn_faz(I))>180 THEN Xferfn_faz(I)=Xferfn_faz(I)-360*SGN(Xferfn_faz(I))
1690 NEXT I
1700 ! STORE DISK-3 OPEN-LOOP VELOCITY FREQUENCY RESPONSE FUNCTION ON DISKETTE -----------------------------------------------
1720 IF Nr_records<10 THEN
1730 Nr_avg$="_"&VAL$(Nr_records)
1740 ELSE
1750 Nr_avg$=VAL$(Nr_records)
1760 END IF
1770 Save_rvect(Xferfn_mag(*),"AVG_"&Nr_avg$&"_MAG:700,1")
1780 Save_rvect(Xferfn_faz(*),"AVG_"&Nr_avg$&"_FAZ:700,1")
1790 END

1800 ! SUBPROGRAMS=================================================================

1810 ! COMPUTE MAGNITUDE IN dB OF REAL AND IMAGINARY FFT RESULTS ----
1820 DEF FNMag_db(REAL Real_part,Imag_part,INTEGER Nr_samples)
1830 RETURN 20*LGT(Nr_samples*SQR(REAL_part^2+Imag_part^2))
1840 FNEND

1850 !

1860 ! COMPUTE PHASE IN DEGREES OF REAL AND IMAGINARY FFT RESULTS ---
1870 DEF FNPhase_deg(REAL Real_part,Imag_part)
1880 SELECT SGN(Real_part*Imag_part)
1890 CASE -1,1
1900 IF Real_part>0 THEN
1910 RETURN 180*ATN(Imag_part/Real_part)/PI
1920 ELSE
1930 RETURN 180*(ATN(Imag_part/Real_part)/PI+SGN(Imag_part))
1940 END IF
1950 CASE 0
1960 IF Real_part<>0 THEN
1970 RETURN 180*(1-SGN(Real_part))
1980 ELSE
1990 RETURN 90*SGN(Imag_part)
2000 END IF
END SELECT

FNEND

! HANNING WINDOW -----------------------------------------------

DEF FNWindow(Element_num,Total_num)
RETURN .5-.5*COS(2*PI*Element_num/Total_num)
FNEND

END OF SUBPROGRAMS -----------------------------------------------