IMPLEMENTATION OF A MODAL FILTERING PROCEDURE

David Raye Fraser
B. Sc. H. (App. Sc.) Queen’s University at Kingston, 1986

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE

in
THE FACULTY OF GRADUATE STUDIES
DEPARTMENT OF ELECTRICAL ENGINEERING

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
April 1988

© David Raye Fraser
In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical Engineering
The University of British Columbia
1956 Main Mall
Vancouver, Canada

Date:  
April 20, 1988
Abstract

A FORTRAN program has been developed in order to investigate the process of modal parameter estimation and non-parametric system identification.

The theory underlying the process of modal parameter estimation is reviewed and the decoupling of a MIMO system into several SISO systems is demonstrated. Modal filtering is shown to be useful in the field of non-parametric system identification and it is shown that it may also be of some use in the field of signal processing.

The program is documented. It simulates the output of a n-th order system from which a smaller order subsystem can be decoupled. The modal parameters of a subsystem output signal and its first two derivatives and the modal parameters of a second subsystem output and its first derivative are calculated. The unit step response of the theoretical system and the subsystem are then calculated. The signals are then modal filtered to produce the periodic unit step response and the periodic unit square wave response. Finally, the discrete Fourier coefficients of the periodic unit step response are calculated.
# Table of Contents

Abstract

Acknowledgement

1 Introduction

2 The Modal Function
   2.1 Identification of Modal Parameters
      2.1.1 Relationship Between the Modal Parameters of System States
      2.1.2 Identification of the $M_1$ Parameters
      2.1.3 Numerical Modal Parameter Estimation
      2.1.4 Justification of Analytical Integration Technique
   2.2 Solving for the Modal Parameters
   2.3 Solving for the Modal Constant

3 Non-Parametric System Identification
   3.1 The Unit Step Response
   3.2 The Periodic Unit Step Response
   3.3 The Periodic Unit Square Wave Response

4 The Modal Program
   4.1 Data Input
   4.2 Noise Simulation
   4.3 Modal Parameter Calculation
D.1 Listing of the Subroutines DSLIMP and FINV ................. 103
  D.1.1 Listing of Subroutine DSLIMP ............................ 103
  D.1.2 Listing of Subroutine FINV ............................... 109
D.2 Listing of Subroutine FFT .................................... 112
List of Tables

A.1 Summary of the modal parameter estimates for example 1. .......... 64
A.2 Summary of non-parametric system responses for subsystem S1. .... 65
A.3 Summary of non-parametric system responses for Subsystem S1. .... 65
A.4 Summary of FFT coefficients. ........................................... 68

B.5 Modal parameter estimates in the presence of additive measurement
    noise; VAR=0.0002. ...................................................... 69
B.6 Modal parameter estimates in the presence of additive measurement
    noise; VAR=0.0004. ...................................................... 71
B.7 Modal parameter estimates in the presence of additive measurement
    noise; VAR=0.0008. ...................................................... 71
List of Figures

2.1 Interval, \( w(t) \), used to identify modal parameters. ....................... 14
2.2 Interval, \( w(t) \), used to calculate the \( M_0 \) constant. ....................... 20
4.3 Intervals, \( w_t \), used to evaluate the modal parameters for the two subsys­

tems. ......................... 33
5.4 Interval, \( w(t) \). ......................................................... 44
A.5 Control input, \( c_0 \) for example A. ........................................ 54
A.6 Intervals for example A. .......................................................... 55
A.7 System responses for S1. .......................................................... 66
A.8 System responses for S2. .......................................................... 67
B.9 Graph indicating the linear relationship between modal parameter esti­

timate biases and the noise standard deviation. .............................. 70
Acknowledgement

First and foremost, I would like to acknowledge the invaluable help of my supervisor, Dr. E.V. Bohn, for his academic and financial aid without which this thesis would not have been possible. Also, I would like to thank the many people at U.B.C. who have made my stay here a pleasurable one.
Early efforts to develop system models relied primarily on determining the response of the system to sinusoidal and transient signals. These methods were continuous in nature and usually involved differential equations or transfer function representations of the system. As most realistic, physical systems are of a continuous nature, this was the obvious approach.

As digital computers came into more widespread use and became more powerful, the trend developed to use discrete methods of system models with a corresponding decrease in the use of continuous model identification (CMI).

One of the difficulties involved with CMI was the need to know the time-derivatives of the signal, usually in the presence of noise. Whether the techniques involved the measurement or the actual calculation of the time-derivatives, they tended to only work reliably in cases where the input/output noise was small. In practice these algorithms were useful only in deterministic, low noise situations. The difficulty in expressing time-derivatives in terms of input-output data lead, in the early nineteen sixties, to the development of new methods that avoided evaluation of derivatives. One of the offsprings of this work was the 'method function' or 'modulating function' method.

In the continuous single-input single-output (SISO) case the output and input are related and expressed as a weighted sum of output and input signal sets. Usually the signal sets took the form of time-derivatives or delayed time-derivatives of the output or input. Since the calculation of these signal set components involve time-derivatives
that are usually unavailable to the observer, it was desirable to perform some sort of linear operation on the signal bases so as to obtain terms that could be measured. This is the basis for most method function algorithms.

Many different methods of method functions have evolved. Shinbrot [21] multiplied the signal set by known functions and then integrated over the data sampling period. Others (Perdreauville and Goodson [15]) involved characterizing the system by partial differential equations and identifying the system parameters by the use of two dimensional method functions.

One problem with method functions is the requirement for evaluation of integral terms. To overcome this obstacle, method functions of time that are the output of linear, time-invariant filters are chosen so that the definite integration terms may be measured directly from the filter output. By constructing a chain of successive cascaded filters all the required definite integrals can be evaluated. The Poisson moment functional (PMF) method (Saha and Rao [19]) is one example of such a technique.

Recently, the use of Walsh functions and block pulse functions has been investigated (Rao and Palanisamy [17], Bohn [1]). This method involves transforming the signal into a Walsh series, or sum of weighted Walsh functions, similar to a Fourier series. In this case a ‘persistently exciting’ Walsh function input such as a square wave is used. Ideally the input is periodic, which allows the use of simple averaging to eliminate random, measurement noise from the estimated data. An important advantage of the Walsh function approach is that the evaluation of integral terms that arise can be reduced to simple algebra (Bohn [2], [3]).

In the case of a multiple-input multiple-output (MIMO) system, the problem of estimating a large number of unknown parameters arises for which most of the above algorithms start to loose their usefulness. Also there is the problem of initial conditions of the system, which not only add extra parameters to the estimation problem, but
also leads to uncertainty in the calculation of multiple integrals.

The method of ‘modal functions’ (Bohn and El-Shafey [7]) has been developed to alleviate these problems. The obstacle of initial conditions is avoided because the modal function representation of a signal is independent of initial conditions. It is shown in this thesis that for the case of a MIMO system, several SISO subsystems can be decoupled from the system and the parameters solved for relatively easily since the number of unknowns in the subsystems is significantly less than in the original system.

It is also shown that modal function techniques can be used in the field of non-parametric system identification and signal processing. The modal representation of a signal can be used to identify the unit step response of the unknown system from sampled input-output data. Also the non-periodic unit step response can be filtered by a modal filter to produce a periodic unit step response that is an exact representation of the unit step response. Furthermore, the output to a unit square wave input to the system can be calculated from the non-periodic data.

From the periodic unit step and square wave responses the fast Fourier transform (FFT) of the signals can be calculated exactly from non-periodic data. This eliminates the problem of windows and the associated spectral smearing that arises in standard FFT methods.

In this thesis the concepts of a modal function and the derivation of the modal unit step responses are reviewed. A computer program is described that will simulate the analytic modal filter relations on an IBM PC.

Emphasis in the thesis is placed on the analysis of modal function properties that are new and that are useful for parametric and non-parametric system identification. The PC program that is developed is meant to investigate the properties in a simple and convenient manner for the user.
Chapter 2

The Modal Function

Through the use of modal functions it is possible to decouple a MIMO system into several SISO subsystems of smaller order. Consider a MIMO linear time-invariant system with a strictly proper transfer function and a state-space representation

\[
\dot{x} = Ax + Bu, \ y = Cx, \tag{2.1}
\]

where \( x \) is the state, \( u \) is the control, and \( y \) is the output. For illustrative purposes we will consider a fourth order system consisting of two subsystems. The system to be discussed is a one input-two output system with phase variable canonical form.

\[
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_3 \\
\dot{x}_4
\end{bmatrix}
= \begin{bmatrix}
0 & 1 & 0 & 0 \\
-a_{11} & -a_{12} & -a_{13} & -a_{14} \\
0 & 0 & 0 & 1 \\
0 & 0 & -a_{23} & -a_{24}
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{bmatrix}
+ \begin{bmatrix}
b_1 \\
b_2 \\
b_3 \\
b_4
\end{bmatrix}u
\tag{2.2}
\]

\[
\begin{bmatrix}
y_1 \\
y_2
\end{bmatrix}
= \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{bmatrix}.
\]

The most convenient representation for system identification is to rewrite the system in input-output form. To do this we must obtain an input-output differential equation without reference to the internal system states.

Start by rewriting (2.2) as

\[
y_1^{(1)} = x_1^{(1)} = x_2 + b_1u \tag{2.3}
\]
Chapter 2. The Modal Function

\[ x^{(1)}_2 = -a_{11}x_1 - a_{12}x_2 - a_{13}x_3 - a_{14}x_4 + b_2u \]  \hfill (2.4)

\[ y^{(1)}_2 = x^{(1)}_3 = x_4 + b_3u \]  \hfill (2.5)

\[ x^{(1)}_4 = -a_{23}x_3 - a_{24}x_4 + b_4u \]  \hfill (2.6)

where \( y^{(k)} \) denotes the \( k \)-th derivative of \( y \) with respect to \( t \). By rearranging (2.3) to get \( x_2 \) explicitly we obtain

\[ x_2 = y^{(1)}_1 - b_1u. \]  \hfill (2.7)

By performing a similar operation on (2.4),

\[ y^{(2)}_1 - b_1u^{(1)} + a_{11}y_1 + a_{12}y^{(1)}_1 - a_{11}b_1u + a_{13}y_2 + a_{14}x_4 = b_2u \]  \hfill (2.8)
results.

Performing the same procedure on the third and fourth states of the system gives

\[ x_4 = y^{(1)}_2 - b_3u \]  \hfill (2.9)

Therefore, we obtain

\[ y^{(2)}_2 - b_3u^{(1)} + a_{23}y_2 + a_{24}y^{(1)}_2 - a_{24}b_3u = b_4u. \]  \hfill (2.10)

Substituting (2.9) into (2.8) to eliminate the \( x_4 \) term gives

\[ y^{(2)}_1 - b_1u^{(1)} + a_{11}y_1 + a_{12}y^{(1)}_1 - a_{11}b_1u + a_{13}y_2 + a_{14}y^{(1)}_2 - a_{14}b_3u = b_2u. \]  \hfill (2.11)

Note that we have now written expressions that relate the system outputs to the system inputs without using any of the internal system states. From (2.11) and (2.10) we want to write the system (2.2) in what is known as the ‘input-output’ form. To do this we must group the inputs on the right hand side of an equation and the outputs on the left hand side. This gives

\[ (y^{(2)}_2 + a_{12}y^{(1)}_1 + a_{11}y_1) + (a_{14}y^{(1)}_2 + a_{13}y_2) = b_1u^{(1)} + (a_{11}b_1 + b_2 + a_{14}b_3)u \]

\[ y^{(2)}_2 + a_{24}y^{(1)}_2 + a_{23}y_2 = b_3u^{(1)} + (a_{24}b_3 + b_4)u \]  \hfill (2.12)
Chapter 2. The Modal Function

If we express (2.12) as a matrix equation we obtain the input-output form for the system as follows,

\[
\begin{bmatrix}
   d_{11}(y_1) + d_{12}(y_2) \\
   0 + d_{22}(y_2)
\end{bmatrix} = \begin{bmatrix}
   F_1(u) \\
   F_2(u)
\end{bmatrix}
\tag{2.13}
\]

where

\[
\begin{align*}
   d_{11}(y_1) &= y_1^{(2)} + a_{12}y_1^{(1)} + a_{11}y_1 \\
   d_{12}(y_2) &= a_{14}y_2^{(1)} + a_{13}y_2 \\
   d_{22}(y_2) &= y_2^{(2)} + a_{24}y_2^{(1)} + a_{23}y_2 \\
   F_1(u) &= b_1u^{(1)} + (a_{11}b_1 + b_2 + a_{14}b_3)u \\
   F_2(u) &= b_3u^{(1)} + (a_{24}b_3 + b_4)u.
\end{align*}
\tag{2.14}
\]

In the case where \(y_1(t)\) and \(y_2(t)\) are completely decoupled, \(d_{12}(y_2) = 0\), and if \(y_1(t)\) and \(y_2(t)\) should have all modes in common then \(d_{21}(y_1)\) would be non-zero.

In practice, the input-output representation of a system is the most convenient way for dealing with modal functions. Also note that by taking Laplace transforms of (2.13) a transfer function representation of the system is obtained.

To obtain the characteristic equation of the system we observe that for the zero input case,

\[
\mathcal{L} \left\{ \begin{bmatrix}
   d_{11}(y_1) + d_{12}(y_2) \\
   0 + d_{22}(y_2)
\end{bmatrix} \right\} = \begin{bmatrix}
   s^2 + a_{12}s + a_{11} & a_{14}s + a_{13} \\
   0 & s^2 + a_{24}s + a_{23}
\end{bmatrix} \begin{bmatrix}
   Y_1(s) \\
   Y_2(s)
\end{bmatrix} = 0.
\tag{2.15}
\]

Let us define the following notation;

\[
\mathcal{L}\{d_{ij}(y_k)\} = d_{ij}(s)Y_k(s).
\tag{2.16}
\]
Chapter 2. The Modal Function

Now we can rewrite (2.15) more compactly as

\[
\begin{bmatrix}
d_{11}(s) & d_{12}(s) \\
0 & d_{22}(s)
\end{bmatrix}
\begin{bmatrix}
Y_1(s) \\
Y_2(s)
\end{bmatrix} = 0.
\] (2.17)

The characteristic equation of the system is then given by

\[
\det \begin{bmatrix}
d_{11}(s) & d_{12}(s) \\
0 & d_{22}(s)
\end{bmatrix} = d_{11}(s)d_{22}(s) = 0.
\] (2.18)

In the illustrative example being used here we obtain the following characteristic equation.

\[
0 = d_{11}(s)d_{22}(s) = (s^2 + a_{12}s + a_{11})(s^2 + a_{24}s + a_{23}) = s^4 + \tilde{a}_3s^3 + \tilde{a}_2s^2 + \tilde{a}_1s + \tilde{a}_0
\] (2.19)

More generally the characteristic equation of the system is as follows:

\[
s^n + a_{n-1}s^{n-1} + \cdots + \tilde{a}_0 = 0
\] (2.20)

and yields the system poles \(s_k, k = 1, \ldots, n\). For simplicity, it is assumed that the poles are distinct.

The modal output function for the system is defined to be

\[
M(y(t), n) = y(t) + m_1y(t - t_d) + \cdots + m_ny(t - nt_d),
\] (2.21)

where \(m_k\) are the modal parameters, \(t_d\) is the modal time delay, and

\[
z_k = e^{\gamma t_d}, k = 1, \ldots, n,
\] (2.22)

are the zeros of the modal characteristic equation,

\[
c(z, n) = z^n + m_1z^{n-1} + \cdots + m_n = 0.
\] (2.23)
Chapter 2. The Modal Function

For distinct poles, the output response for zero input has the form

$$y_0(t) = c_1 e^{t} + \cdots + c_n e^{n t}. \quad (2.24)$$

Since $c(z_k, n) = 0$, it is seen that

$$M(y_0(t), n) = 0. \quad (2.25)$$

To obtain persistent excitation, general piecewise constant inputs are chosen and (2.25) is used to eliminate unknown initial conditions.

The unit-step response vector $g^T(t) = [g_1(t), g_2(t)]$ is defined as the solution of (2.13) with input $u = 1$, $t > 0$, satisfying the initial conditions

$$g(0) = 0, g^{(1)}(0) = 0 \quad (2.26)$$

Derivatives of the unit-step on the right-hand side of (2.12) result in the unit-impulse function $\delta(t)$, where $\delta(0) = 0$, $\delta(0^+) = 0$. Initial conditions for $t = 0^+$ are given by

$$g(0^+) = 0 \quad (2.27)$$
$$g_2^{(1)}(0^+) = b_3 \quad (2.28)$$
$$g_1^{(1)}(0^+) = b_1 - a_{14} b_3. \quad (2.29)$$

When $t \geq 0^+$, the unit step response function has the following representation

$$g_i(t) = g_i(\infty) + c_{i1} e^{s_1 t} + \cdots + c_{in} e^{s_n t}. \quad (2.30)$$

where

$$g_1(\infty) = \frac{a_{11} b_1 + b_2 + a_{14} b_3 - a_{13} g_2(\infty)}{a_{11}} \quad (2.31)$$

and where

$$g_2(\infty) = \frac{a_{24} b_3 + b_4}{a_{23}}. \quad (2.32)$$
When \( t \geq n_i t_d \), it follows from (2.23) that for each subsystem

\[
M(g_i(t), n_i) = g_i(t) + m_{i1}g_i(t - t_d) + \cdots + m_{in}g_i(t - n_i t_d) = M_{i0}
\]

where (see 2.23)

\[
M_{i0} = c(1, n_i) y_i(\infty).
\]

2.1 Identification of Modal Parameters

2.1.1 Relationship Between the Modal Parameters of System States

An analysis in this section is given of modal parameter estimation from input-output data and the use of modal functions for estimating output derivatives.

Let \( z(t) \) be defined as the modal vector and let

\[
y(t) = c_1 e^{s_1 t} + \cdots + c_n e^{s_nt} = ez(t)
\]

where \( e = [1,1,\ldots,1] \),

and \( z^T(t) = [c_1 e^{s_1 t}, \ldots, c_n e^{s_nt}] \).

The modal output equation for the zero-input response (2.35) is defined by

\[
y(t) \quad + \quad M_1 Y(t) = 0
\]

where \( M_1 = [m_1, m_2, \ldots, m_n] \),

and \( Y^T(t) = [y(t - \tau_1), \ldots, y(t - \tau_n)] \).

where \( \tau_1 < \tau_2 < \cdots < \tau_n \) are modal time-shift parameters. If we define

\[
V_m = \begin{bmatrix} e^{-s_1 \tau_1} & e^{-s_2 \tau_1} & \cdots & e^{-s_n \tau_1} \\ e^{-s_1 \tau_2} & e^{-s_2 \tau_2} & \cdots & e^{-s_n \tau_2} \\ \vdots & \vdots & \cdots & \vdots \\ e^{-s_1 \tau_n} & e^{-s_2 \tau_n} & \cdots & e^{-s_n \tau_n} \end{bmatrix},
\]
then $V_m z(t) = Y(t)$, and from (2.38) it follows that

$$e + M_1 V_m = 0.$$  \hfill (2.42)

We can similarly define the modal equation for output derivatives:

$$y^{(i)}(t) + M_{i+1} Y(t) = 0.$$  \hfill (2.43)

From (2.35) it is seen that

$$y^{(i)}(t) = [s_1^i, \ldots, s_n^i] z(t) = e D^i(s_k) z(t),$$  \hfill (2.44)

where

$$D(s_k) = \begin{bmatrix} s_1 & 0 & \cdots & 0 \\ 0 & s_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & s_n \end{bmatrix}.$$  \hfill (2.45)

Using the definition of $V_m$ we can rewrite (2.43) as

$$e D^i(s_k) + M_{i+1} V_m = 0.$$  \hfill (2.46)

Define the following Vandermonde matrix, $V_a$, as

$$V_a = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ s_1 & s_2 & \cdots & s_n \\ \vdots & \vdots & \ddots & \vdots \\ s_1^n & s_2^n & \cdots & s_n^n \end{bmatrix}.$$  \hfill (2.47)

and define $e_i = [0, \ldots, 0, 1, 0, \ldots, 0]$, with the 1 in the i-th position.

Using the above definition it is seen that,

$$e D^i(s_k) = e_{i+1} V_a,$$  \hfill (2.49)
and (2.46) can be written as
\[ e_{i+1}V_a + M_{i+1}V_m = 0. \] (2.50)

Note that \( eD^i(s_k) = e_{i+1}V_a = [s_1^i, \ldots, s_n^i]. \)

Summarizing the above for \( y, y^{(1)}, y^{(2)} \), we can see that

\[
\begin{align*}
M_1V_m &= -e \\
M_2V_m &= -eD(s_k) = M_1V_mD(s_k) \\
M_3V_m &= -eD^2(s_k) = M_1V_mD^2(s_k).
\end{align*}
\]

(2.51) (2.52) (2.53)

Now we can state that

\[
\begin{align*}
M_2V_m &= M_1V_mD(s_k) \\
M_2 &= M_1V_mD(s_k)V_m^{-1} \\
M_2 &= M_1\overline{A}.
\end{align*}
\]

(2.54) (2.55) (2.56)

Furthermore,

\[
\begin{align*}
M_3V_m &= M_1V_mD^2(s_k) \\
M_3 &= M_1V_mD^2(s_k)V_m^{-1} \\
M_3 &= M_2\overline{A}.
\end{align*}
\]

(2.57) (2.58) (2.59) (2.60)

where

\[ \overline{A} = V_mD(s_k)V_m^{-1}. \] (2.61)

It is seen from (2.56) and (2.60) that modal output derivative parameters \( M_k, k \geq 2 \), can be determined from \( M_1 \) and the system matrix \( \overline{A} \). The analytic expressions (2.41) and (2.61) are useful for the numerical evaluation of \( M_k \) when the system poles are known.
Chapter 2. The Modal Function

2.1.2 Identification of the $M_i$ Parameters

For piecewise constant inputs $u_j$ over a data interval $T_j \{t_j < t < t_{j+1}\}$ it is seen from (2.33) that

$$y(t) + M_i Y(t) = M_0 u_j, t_j + n t_d \leq t \leq t_{j+1} \tag{2.62}$$

To identify the $M_i$ parameters we note that equation (2.62) yields

$$\langle w_j(t), y(t) \rangle + M_1 \langle w_j(t), Y(t) \rangle = M_0 \langle w_j(t), u_j(t) \rangle. \tag{2.63}$$

where $\langle w_j(t), y(t) \rangle$ represents the inner product of the two functions and $w(t)$ is a suitable function defined on $T_j$. Now, choose $T_j, w_j$ such that

$$\langle w_j(t), u_j(t) \rangle = 0. \tag{2.64}$$

To evaluate the left hand side terms of (2.63) we must evaluate the following

$$\langle w_j(t), e^{st} \rangle = \int_{t_j}^{t_{j1}} e^{st} dt - \int_{t_{j1}}^{t_{j2}} e^{st} dt + \cdots + \int_{t_{j(N-1)}}^{t_{jN}} e^{st} dt$$

$$= \frac{1}{s} \sum_{k=1}^{N} \alpha_{jk} e^{st_k} \tag{2.65}$$

Using (2.37) and (2.65) yields

$$\langle w_j(t), z(t) \rangle = D^{-1}(s_k) \sum_{k=1}^{N} \alpha_{jk} z(t_{jk})$$

$$= D^{-1}(s_k) Z_j. \tag{2.66}$$

The data vector (2.66) is useful for the analysis of the identification procedure. Using (2.66) it is seen that the processed output data has the representation

$$\langle w_j(t), y(t) \rangle = e(T_j, w_j(t)z(t))$$

$$= \left[ e D^{-1}(s_k) Z_j \right], \tag{2.67}$$

$$\langle w_j(t), Y(t) \rangle = \left[ V_m D^{-1}(s_k) Z_j \right]. \tag{2.68}$$
Substituting (2.67) and (2.68) into (2.63) and choosing \( n \) data intervals gives

\[
[eD^{-1}(s_k)Z] + M_1 [V_mD^{-1}(s_k)Z] = 0 \tag{2.69}
\]

where \( Z = [Z_1, Z_2, \ldots, Z_n] \tag{2.70} \)

Matrix inversion gives

\[
[eD^{-1}(s_k)Z] [V_mD^{-1}(s_k)Z]^{-1} + M_1 = 0 \tag{2.71}
\]

or,

\[
eV_m^{-1} + M_1 = 0
\]

\[
M_1 = -eV_m^{-1}. \tag{2.72}
\]

The result given by (2.72) is the defining equation for \( M_1 \) (see 2.51).

2.1.3 Numerical Modal Parameter Estimation

For an \( n - th \) order system we must identify \( n \) modal parameters as well as the \( M_0 \) constant in (2.34). Given the definitions of a modal output function for output \( y_1(t) \)

\[
y_1(t) + m_{11}y_1(t - t_d) + \cdots + m_{1n}y_1(t - nt_d) = M_{10}u_i, \tag{2.73}
\]

we can take the inner product of the above equation with respect to some interval, \( I_i \), and obtain

\[
\langle w, y(t) \rangle + m_{11} \langle w, y(t - t_d) \rangle + \cdots + m_{1n} \langle w, y(t - nt_d) \rangle = \langle w, M_{10}u_i \rangle \tag{2.74}
\]

where

\[
\langle w(t), f(t) \rangle = \int_{-\infty}^{+\infty} w(t)f(t)dt. \tag{2.75}
\]
In order to identify the \( n \) \( m_i \) parameters we choose the interval \( I \), given by the function \( w(t) \), as (see figure 2.1)

\[
w(t) = \begin{cases} 
1 & , a t_d \leq t < b t_d \\
-1 & , b t_d \leq t < c t_d \\
0 & , \text{otherwise}
\end{cases}
\]

(2.76)

where

\[
b = \frac{a + c}{2}.
\]

(2.77)

and where \( t_d \) is a suitable sampling interval.

Evaluating \( \langle w(t), y(t) \rangle \) we obtain:

\[
\langle w(t), y(t) \rangle = \langle w(t), c_0 + c_1 e^{s_1 t} + \cdots + c_n e^{s_n t} \rangle
\]

\[
= \langle w(t), c_0 \rangle + \langle w(t), c_1 e^{s_1 t} \rangle + \cdots + \langle w(t), c_n e^{s_n t} \rangle
\]

(2.78)

\[
= c_0 \langle w(t), 1 \rangle + c_1 \langle w(t), e^{s_1 t} \rangle + \cdots + c_n \langle w(t), e^{s_n t} \rangle
\]
The inner product terms of (2.78) can be evaluated analytically by

\[
\langle w(t), e^{st} \rangle = \int_{at_d}^{bt_d} e^{st} dt - \int_{ct_d}^{bt_d} e^{st} dt \\
= \frac{1}{s_i} \left( e^{s_i t} \bigg|_{at_d}^{bt_d} - \frac{1}{s_i} \left( e^{s_i t} \bigg|_{ct_d}^{bt_d} \right) \right) \\
= \frac{1}{s_i} \left( e^{s_i bt_d} - e^{s_i at_d} - e^{s_i ct_d} + e^{s_i bt_d} \right) \\
= \frac{1}{s_i} \left( 2e^{s_i bt_d} - e^{s_i at_d} - e^{s_i ct_d} \right). 
\]

Clearly,

\[
\langle w(t), c \rangle = 0 \tag{2.80}
\]

where \(c\) is any constant. Therefore (2.74) reduces to

\[
\langle w(t), y_1(t) \rangle + m_{11} \langle w(t), y_1(t - t_d) \rangle + \cdots + m_{1n} \langle w(t), y_1(t - nt_d) \rangle = 0 \tag{2.81}
\]

In order to identify the \(n\) modal parameters we must evaluate (2.81) over \(n\) distinct intervals, \(w_i(t)\), that satisfy (2.76), and choose a different value for \(u_i\), in (2.73), to simulate the persistent excitation of the system modes.

To identify \(y_1^{(1)}\) and \(y_1^{(2)}\), the following equations are used:

\[
y_1^{(1)}(t) + m_{21} y_1(t) + \cdots + m_{2(n-1)} y_1(t - (n-1)t_d) = M_{20} u_i, \tag{2.82}
\]

\[
y_1^{(2)}(t) + m_{31} y_1(t) + \cdots + m_{3(n-1)} y_1(t - (n-1)t_d) = M_{30} u_i. \tag{2.83}
\]

By taking the inner products of (2.82) and (2.83) over an interval \(w_i\) as in (2.76), the following is obtained

\[
\langle w(t), y_1^{(1)}(t) \rangle + m_{21} \langle w(t), y_1(t) \rangle + \cdots + m_{2(n-1)} \langle w(t), y_1(t - (n-1)t_d) \rangle = 0, \tag{2.84}
\]

\[
\langle w(t), y_1^{(2)}(t) \rangle + m_{31} \langle w(t), y_1(t) \rangle + \cdots + m_{3(n-1)} \langle w(t), y_1(t - (n-1)t_d) \rangle = 0. \tag{2.85}
\]

Since

\[
y_1^{(1)}(t) = c_1 s_1 e^{s_1 t} + \cdots + c_n s_n e^{s_n t} \tag{2.86}
\]
Chapter 2. The Modal Function

and

\[ y_1^{(2)}(t) = c_1 s_1^2 e^{s_1t} + \cdots + c_n s_n^2 e^{s_nt} \] (2.87)

it follows that

\[ \langle w(t), y_1^{(1)}(t) \rangle = c_1 s_1 \langle w(t), e^{s_1t} \rangle + \cdots + c_n s_n \langle w(t), e^{s_nt} \rangle, \] (2.88)

\[ \langle w(t), y_1^{(2)}(t) \rangle = c_1 s_1^2 \langle w(t), e^{s_1t} \rangle + \cdots + c_n s_n^2 \langle w(t), e^{s_nt} \rangle. \] (2.89)

It is also necessary to calculate the \( \langle w(t), y_1(t - j t_d) \rangle \) terms. This is done by the following equation

\[ \langle w(t), y_1(t - j t_d) \rangle = c_1 \langle w(t), e^{s_1(t - j t_d)} \rangle + \cdots + c_n \langle w(t), e^{s_n(t - j t_d)} \rangle \]
\[ = c_1 e^{-s_1 j t_d} \langle w(t), e^{s_1 t} \rangle + \cdots + c_n e^{-s_n j t_d} \langle w(t), e^{s_n t} \rangle \] (2.90)

As in the case of the first state of the system, the modal parameters for \( y_1^{(1)} \) and \( y_1^{(2)} \) states can be identified by evaluating (2.84) and (2.85) over \( n \) distinct intervals.

2.1.4 Justification of Analytical Integration Technique

In practice we cannot analytically evaluate the \( \langle w(t), y(t) \rangle \) terms of (2.74) but must use some method of numerical integration. However for the computer simulation of (2.74) a process of analytical integration has been used. It is neccessary to justify the validity of this choice. The theorem below shows that given a signal that may be expressed as a linear combination of exponential terms, there exists an exact analytic relationship between the result obtained through analytic integration and through a trapezoidal approximation to the integration.

Theorem 2.1.1 If a signal, \( y(t) \), can be expressed as

\[ y(t) = c_1 e^{s_1 t} + c_2 e^{s_2 t} + \cdots + c_n e^{s_n t}, \]
and defining

\[ \langle w(t), y(t) \rangle = \int_{-\infty}^{+\infty} w(t)y(t)dt. \]  

(2.91)

then, if \( \langle T_j, y(t) \rangle \) and \( \langle I_j, y(t) \rangle \) refer to evaluating (2.91) by trapezoidal approximation and by analytic integration respectively, the following relationship holds:

\[ \langle T_j, e^{st} \rangle = F(s)\langle I_j, e^{st} \rangle, \]  

(2.92)

where

\[ F(s) = \frac{s\Delta}{2} \cdot \frac{e^{s\Delta} + 1}{e^{s\Delta} - 1}. \]  

(2.93)

Proof: Given that

\[ y(t) = c_1e^{s_1t} + c_2e^{s_2t} + \cdots + c_ne^{s_nt}, \]

we can write

\[ \langle w(t), y(t) \rangle = c_1\langle w(t), e^{s_1t} \rangle + c_2\langle w(t), e^{s_2t} \rangle + \cdots + c_n\langle w(t), e^{s_nt} \rangle. \]  

(2.94)

Therefore to evaluate (2.94) we must only evaluate

\[ \langle w(t), e^{s_it} \rangle, i = 1, \ldots, n. \]  

(2.95)

Using analytic integration, and assuming for now that \( \langle I_j, w(t) \rangle = 1, t_j \leq t \leq t_{j+1}, \)

\[ \langle I_j, e^{st} \rangle = \int_{t_j}^{t_{j+1}} e^{st}dt = \frac{e^{st_{j+1}} - e^{st_j}}{s}, \]  

(2.96)

where \( t_{j+1} - t_j = N_j\Delta. \)

Evaluating the same expression in (2.95) but using a trapezoidal approximation, of \( N \) partitions, we get

\[ \langle T_j, e^{st} \rangle = \frac{\Delta}{2} \left[ e^{s_jt} + 2e^{s(t_j+\Delta)} + \cdots + 2e^{s(t_j+(N-1)\Delta)} + e^{st_{j+1}} \right] \]

\[ = \frac{\Delta e^{s_it_j}}{2} \left[ 1 + 2(\sigma + \cdots + \sigma^{N-1}) + \sigma^N \right], \]  

(2.97)

where \( \sigma = e^{s\Delta} \) and \( t_{j+1} = t_j + N\Delta \).
Now if we let

\[ x = \sigma + \sigma^2 + \cdots + \sigma^{N-1} \]

then \( \sigma x = \sigma^2 + \cdots + \sigma^{N-1} + \sigma^N \)

and subtracting the two equations, we get

\[ x(1 - \sigma) = \sigma - \sigma^N \]

\[ x = \frac{\sigma - \sigma^N}{1 - \sigma}. \quad (2.98) \]

Substituting (2.98) into (2.97) we get

\[
\begin{align*}
\langle T_j, e^{st} \rangle &= \frac{\Delta}{2} \left[ 1 + \sigma^N + \frac{2(\sigma - \sigma^N)}{1 - \sigma} \right] e^{st_j} \\
&= \frac{\Delta}{2(1 - \sigma)} \left[ 1 + \sigma^N - \sigma - \sigma^{N+1} + 2\sigma - 2\sigma^N \right] e^{st_j} \\
&= \frac{\Delta}{2(1 - \sigma)} \left[ 1 + \sigma - \sigma^N - \sigma^{N+1} \right] e^{st_j} \\
&= \frac{\Delta(1 - \sigma)(1 - \sigma^N)}{2(1 - \sigma)} e^{st_j} \\
&= \frac{\Delta}{2} \cdot \frac{\sigma + 1}{\sigma - 1} \left[ e^{sNj\Delta} - 1 \right] e^{st_j} \\
&= \frac{\Delta}{2} \cdot \frac{e^{s\Delta} + 1}{e^{s\Delta} - 1} \left[ e^{sNj\Delta} - 1 \right] e^{st_j} \\
&= \frac{s\Delta}{2} \cdot \frac{e^{s\Delta} + 1}{e^{s\Delta} - 1} \langle I_j, e^{st} \rangle \quad (2.100)
\end{align*}
\]

Therefore

\[
\begin{align*}
\langle T_j, e^{st} \rangle &= F(s) \langle I_j, e^{st} \rangle \quad (2.101) \\
\text{where } F(s) &= \frac{s\Delta}{2} \cdot \frac{e^{s\Delta} + 1}{e^{s\Delta} - 1} \quad (2.102)
\end{align*}
\]

and we can easily relate the approximate integration of (2.94) to the exact analytic integration.
Chapter 2. The Modal Function

Theorem 2.1.1 shows that it is valid to use the exact analytic integration formulas for the PC Modal program since we can easily relate this to the result that we would get using the trapezoidal approximation that would be used in practice if we were estimating the modal parameters of an actual physical system using sampled output data.

2.2 Solving for the Modal Parameters

Once the inner product modal equations (2.81), (2.84), and (2.85), have been evaluated over \( n \) intervals the following system of equations can be set up and the modal parameters solved for:

\[
\begin{bmatrix}
\langle w_1, y(t - t_d) \rangle & \langle w_1, y(t - 2t_d) \rangle & \cdots & \langle w_1, y(t - nt_d) \rangle \\
\langle w_2, y(t - t_d) \rangle & \langle w_2, y(t - 2t_d) \rangle & \cdots & \langle w_2, y(t - nt_d) \rangle \\
\vdots & \vdots & \ddots & \vdots \\
\langle w_n, y(t - t_d) \rangle & \langle w_n, y(t - 2t_d) \rangle & \cdots & \langle w_n, y(t - nt_d) \rangle
\end{bmatrix}
\begin{bmatrix}
m_{11} \\
m_{12} \\
\vdots \\
m_{1n}
\end{bmatrix}
=
\begin{bmatrix}
-\langle w_1, y(t) \rangle \\
-\langle w_2, y(t) \rangle \\
\vdots \\
-\langle w_n, y(t) \rangle
\end{bmatrix}
\tag{2.103}
\]

and

\[
\begin{bmatrix}
\langle w_1, y(t) \rangle & \langle w_1, y(t - t_d) \rangle & \cdots & \langle w_1, y(t - (n - 1)t_d) \rangle \\
\langle w_2, y(t) \rangle & \langle w_2, y(t - t_d) \rangle & \cdots & \langle w_2, y(t - (n - 1)t_d) \rangle \\
\vdots & \vdots & \ddots & \vdots \\
\langle w_n, y(t) \rangle & \langle w_n, y(t - t_d) \rangle & \cdots & \langle w_n, y(t - (n - 1)t_d) \rangle
\end{bmatrix}
\begin{bmatrix}
m_{i1} \\
m_{i2} \\
\vdots \\
m_{in}
\end{bmatrix}
=
\begin{bmatrix}
-\langle w_1, y^{(i)}(t) \rangle \\
-\langle w_2, y^{(i)}(t) \rangle \\
\vdots \\
-\langle w_n, y^{(i)}(t) \rangle
\end{bmatrix}
\tag{2.104}
\]

where \( i = 2, 3 \).

To solve (2.103) and (2.104) Gaussian elimination is used.
2.3 Solving for the Modal Constant

To solve for $M_{10}$, a new interval, $w$ is defined of the following form (see figure 2.2).

$$w(t) = \begin{cases} 
1, & at_d \leq t \leq bt_d \\
0, & \text{otherwise}
\end{cases} \quad (2.105)$$

Reevaluating (2.74) we see that

$$\langle w(t), M_{10} \rangle = \int_{at_d}^{bt_d} M_{10} dt = (b - a)t_d M_{10} \quad (2.106)$$

and that

$$\langle w(t), e^{st} \rangle = \int_{at_d}^{bt_d} e^{st} dt = \frac{1}{s} (e^{bt_d} - e^{at_d}) \quad (2.107)$$

Therefore, from (2.74) and (2.79)

$$\langle w(t), y_1(t) \rangle + m_{11} \langle w(t), y_1(t - t_d) \rangle + \cdots + m_{1n} \langle w(t), y_1(t - nt_d) \rangle = (b - a)t_d M_{10} \quad (2.108)$$

or

$$M_{10} = \frac{1}{t_d(b - a)} (\langle w(t), y_1(t) \rangle + m_{11} \langle w(t), y_1(t - t_d) \rangle + \cdots + m_{1n} \langle w(t), y_1(t - nt_d) \rangle) \quad (2.109)$$
where \( \langle w(t), y_1(t - it_d) \rangle, i = 0, 1, \ldots, n \) is given by (2.90) and (2.107).

Similarly,

\[
M_{20} = \frac{1}{t_d(b - a)} \left( \langle w(t), y_1^{(1)}(t) \rangle + m_{21} \langle w(t), y_1(t) \rangle + \cdots + m_{2n} \langle w(t), y_1(t - (n - 1)t_d) \rangle \right)
\]  

(2.110)

and

\[
M_{30} = \frac{1}{t_d(b - a)} \left( \langle w(t), y_1^{(2)}(t) \rangle + m_{31} \langle w(t), y_1(t) \rangle + \cdots + m_{3n} \langle w(t), y_1(t - (n - 1)t_d) \rangle \right)
\]  

(2.111)

where \( \langle w(t), y'(t) \rangle \) is given by (2.88) and (2.90) and \( \langle w(t), y''(t) \rangle \) is given by (2.89) and (2.90).

The above process will identify the modal parameters of a system for \( y, y^{(1)}, \) and \( y^{(2)} \).

It has been shown that through the use of modal parameters a signal may be represented by a continuous-time ARMA type of equation independent of initial conditions. For a MIMO system, each output can be represented separately, which decreases the number of parameters that must be estimated and decreases the time required to obtain accurate estimates.

The parameters can be identified from input-output data or from system poles, and the knowledge of the system poles enables us to obtain modal function representations of the time-derivatives of the signal with very little extra computational effort. The estimates of the parameters can be easily carried out using simple matrix algebra which is easily implemented on a digital computer.
Chapter 3

Non-Parametric System Identification

Non-parametric system identification can be made in the time-domain or in the frequency domain. In the time-domain this usually involves identification of the impulse response or the unit-step response. In the frequency domain this involves determining the frequency response of the system (Wellstead [26], Young [27]).

Through the use of modal parameters and the modal output function it is possible to obtain the non-parametric unit step response directly from input-output data. It is also possible to convert non-periodic data to periodic data and determine the frequency response of the system. This may have applications in the field of spectral analysis.

3.1 The Unit Step Response

To simplify the notation let $g(t)$ represent one of the elements of the unit-step response matrix. If the system input, $u$, is piecewise continuous over a sequence of time intervals of length $T_k > n t_d$ then the output response $y(t)$, associated with $g(t)$, can be given, for $t > T_{-1}$ as

$$y(t) = y_0 + g(t + T_{-1})u(0) + g(t)\Delta u(1) + \cdots,$$

$$= v(t) + g(t)\Delta u(1) + \cdots, \quad (3.112)$$

where $u(k)$ is the input over the interval $0^+ \leq t - (T_1 + \cdots + T_k) \leq T_{k+1}$, $y_0 = y(T_{-1})$, $\Delta u(k) = u(k) - u(k - 1)$, and $v(t) = y_0 + g(t + T_{-1})u(0)$. Note that $y_0$ represents the initial conditions of the system.
The modal function for \( y(t) \) is given by
\[
M(y(t), n) = M(y_0(t), n) + M(g(t + T_d), n)u(0) + M(g(t), n)\Delta u(1) + \cdots
\]
\[
= \begin{cases} 
M_0u(0) + M(g(t), n)\Delta u(1) & , t \geq 0^+ \\
M(v(t), n) & , t \leq 0 
\end{cases}
\]
(3.113)

where \( M(v(t), n) = M_0u(0), t \geq 0^+ \).

This can be generalized for the \( k \)-th interval and for \( t \geq 0^+ \) as
\[
M(y(t + T_1 + \cdots + T_{k-1}), n) = M_0u(k-1) + M(g(t), n)\Delta u(k).
\]
(3.114)

Rewriting \( M(y(t), n) \) as \( Y(t, n) \) we obtain
\[
Y((t + T_1 + \cdots + T_{k-1}), n) = M_0u(k-1) + G(t, n)\Delta u(k).
\]
(3.115)

By using the simplified case for the first interval, \( k = 1 \), the following is obtained,
\[
Y(t, n) = M_0u(0) + G(t, n)\Delta u(1).
\]
(3.116)

Solving (3.116) for \( G(t, n) \) we obtain,
\[
G(t, n) = \frac{Y(t, n) - M_0u(0)}{\Delta u(1)},
\]
(3.117)

where for the unit step response \( \Delta u(1) = 1 \).

From the definition of the modal function:
\[
G(t, n) = g(t) + m_1g(t - t_d) + \cdots + m_ng(t - nt_d),
\]
(3.118)
solving for \( g(t) \) yields
\[
g(t) = G(t, n) - m_1g(t - t_d) - \cdots - m_ng(t - nt_d).
\]
(3.119)

It is seen from (3.117) and (3.119) that, since \( g(t) = 0, t < 0 \), \( g(t) \) can be recursively evaluated for \( t > 0 \) given only input-output data.
The modal function $G(t, n)$ can be identified by use of (3.117). The non-parametric identification of $g(t)$ is then obtained by recursive use of (3.119). It is seen from (2.33) that $G(t, n) = M_0$ when $t \geq nt_d$. It is of interest to investigate the properties of $G(t, n)$ in the transition interval $0 \leq t \leq nt_d$. Consider the generalization of (2.14) to an $n$-th order differential equation of the form $d(y(t)) = F(u(t))$ having a unit-step input solution $g(t)$. It follows from the definition of the unit-step response that

$$d(G(t, n)) = F(\alpha(t)),$$  

(3.120)

where $\alpha(t)$ is piecewise constant in the transition interval.

$$\alpha(t) = \begin{cases} 1 & ,0^+ \leq t < t_d \\ 1 + m_1 & ,t_d \leq t < 2t_d \\ \vdots \\ 1 + m_1 + \cdots + m_n & ,nt_d \leq t \end{cases}$$  

(3.121)

### 3.2 The Periodic Unit Step Response

The piecewise constant function $\alpha(t)$ and the solution of (3.120) can be expressed in terms of Walsh functions. In using Walsh functions it is convenient to use a normalized time $x = \frac{t}{t_d}$ such that the data interval corresponds to $-\frac{1}{2} \leq x \leq \frac{1}{2}$. For notational convenience, $\sigma = \frac{1}{2}$ is used. Since $G(nt_d, n) = M_0$, $G(0, n) = 0$ it is possible to define $G_p(x, n)$, the periodic modal unit step response on the interval $-\sigma \leq x \leq \sigma$ by

$$G_p(x, n) = -\sigma M_0 + G(x, n),$$  

$$= -0.5M_0 + G(x, n).$$  

(3.122)

The function $G_p(x, n)$ satisfies the periodicity condition

$$G_p(x, n) = -G_p((x - \sigma), n) = -G_p((x + \sigma), n)$$  

(3.123)
for an odd symmetric function of period $2\sigma = 1$.

It is seen from (3.117) and (3.122) that the periodic function $G_p(x, n)$ can be identified from non-periodic output data. It is shown in the next section that the system frequency response can be obtained from $G_p(x, n)$. This is an important result since it allows the use of non-periodic data in determining the frequency response. Classical spectral analysis methods require window functions and the use of approximations when non-periodic data is sampled to identify spectral components.

For analysis and computer simulation the equation

$$G_p(x, n) = -\sigma M_0 + g(t) + m_1g(t - t_d) + \cdots + m_ng(t - nt_d) \quad (3.124)$$

obtained by substituting (3.118) into (3.122) is useful.

### 3.3 The Periodic Unit Square Wave Response

Traditional spectral analyses methods use periodic inputs and outputs to identify the system transfer function. In this case, by averaging over each cycle of the input-output, we can reduce the effect of measurement noise. However, for efficient parameter estimation we usually desire a persistently excited input and in most cases, this is a non-periodic signal. In the previous section it was shown that a periodic modal function, $G_p(x, n)$ can be identified from non-periodic data. This implies that there should be a relationship between $G_p(x, n)$ and $H(j\omega)$, the transfer function. To obtain this relationship we consider a periodic unit square wave input and let $R(x, n)$ represent the periodic output response. Since the system input is the Walsh function $s_{al}(1, x)$, we use the notation $R(x, n) = R_{s_1}(x, n)$. The following theorem is proven in [4].
Theorem 3.3.1 Let \((n + 1)\tau_d = \sigma = 0.5\). The modal function \(G_p(x, n)\) and response function \(R_{s_1}(x, n)\) satisfy the modal equation

\[
G_p(x, n) = \sigma(R_{s_1}(x, n) + m_1R_{s_1}(x - \tau, n) + \cdots + m_nR_{s_1}(x - n\tau, n))
\]  

(3.125)

and the matrix equation

\[
L[R_{s_1}(x, n)] = L[G_p(x, n)]C^{-1}
\]

where

\[
L[G_p(x, n)] = \begin{bmatrix}
G_p(x, n) & G_p(x - \tau, n) & \cdots & G_p(x - n\tau, n)
\end{bmatrix}
\]

(3.126)

\[
C = \begin{pmatrix}
c_0 & -c_n & \cdots & -c_1 \\
c_1 & c_0 & \cdots & -c_2 \\
\vdots & \vdots & \ddots & \vdots \\
c_n & c_{n-1} & \cdots & c_0
\end{pmatrix}
\]

(3.127)

where \(c_0 = \sigma, c_k = \sigma m_k, k = 1, \ldots, n\).

Proof: It follows from (3.120), (3.121), and (3.119) that \(D(G_p(x, n)) = N(\alpha(x))\), where \(\alpha(x)\) satisfies the modal equation

\[
\alpha(x) = \sigma [sal(1, x) + m_1sal(1, x - \tau) + \cdots + m_nsal(1, x - n\tau)].
\]

(3.128)

It is seen from (3.128) that the input is a sequence of \(sal(1, x - j\tau), j = 0, \ldots, n\), terms. Equation (3.125) follows from the definition of \(R_{s_1}(x, n)\) and by noting that \(G_p(x, n)\) and \(sal(1, x)\) satisfy condition (3.122). The existence proof for \(C^{-1}\) follows by contradiction. If \(\det(C) = 0\), then constants \(\gamma_k, k = 1, \ldots, n\), exist such that

\[
F(x) = R_{s_1}(x, n) + \gamma_1R_{s_1}(x - \tau, n) + \cdots + \gamma_nR_{s_1}(x - n\tau, n) = 0.
\]

(3.129)

Equation (3.129), however, contradicts the fact that since \(D[R_{s_1}(x, n)] = sal(1, x) \neq 0\), then \(D(F(x)) \neq 0\).
Chapter 3. Non-Parametric System Identification

It follows from a Fourier series input-output representation that the transfer function matrix is given by

\[ H(j\omega_k) = \sigma^2k \int_0^\sigma R_{s1}(x, n)e^{j2\pi kx}dx \]  

(3.130)

where \( \omega_k = 2\pi k \).

Since \( R_{s1}(x - \sigma) = -R_{s1}(x) \) and since \( (n+1)\tau_d = \sigma \), it follows from (3.125) that

\[
\begin{bmatrix}
1 & m_1 & m_2 & \cdots & m_n \\
-m_n & 1 & m_1 & \cdots & m_{n-1} \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
-m_1 & -m_2 & \cdots & -m_n & 1
\end{bmatrix}
\begin{bmatrix}
R_{s1}(x) \\
R_{s1}(x - \tau_d) \\
\vdots \\
R_{s1}(x - n\tau_d)
\end{bmatrix}
= 2
\begin{bmatrix}
G_p(x, n) \\
G_p(x - \tau_d, n) \\
\vdots \\
G_p(x - n\tau_d, n)
\end{bmatrix}
\]

(3.131)

The matrix in (3.131) is a Toeplitz matrix. Solving for \( R_{s1}(x) \) gives

\[ R_{s1}(x) = \gamma_0G_p(x, n) + \gamma_1G_p(x - \tau_d, n) + \cdots + \gamma_nG_p(x - n\tau_d, n), \]  

(3.132)

where

\[ [\gamma_{ij}] = [C^{-1}_{ij}], \]

which can be used in (3.130) to determine the frequency response. Once again the periodicity conditions that

\[ R_{s1}(x, n) = -R_{s1}(x - \sigma, n) = -R_{s1}(x + \sigma, n) \]  

(3.133)

holds as in the case of \( G_p(x, n) \).

From (3.130) and (3.131) we can see that \( H(j\omega_k) \) can now be identified from non-periodic data without having to resort to the use of window functions and their resulting spectral smearing errors. For the modal function approach, no assumptions need be placed on the normalized output data, \( y(x) \).
Chapter 4

The Modal Program

The program Modal is written in FORTRAN and is stored on floppy diskette for use with an IBM P.C. or any compatible micro computer.

The program simulates a system that can be decoupled into a subsystem of order $N_1$ and a subsystem of order $N_2$, $N_2 \leq N_1$. Any or all of the poles of the two subsystems may be common to both. This is to simulate mode coupling between the two subsystems.

The modal parameters of the first 3 states $y_1(t)$, $y'_1(t)$, and $y''_1(t)$, of the first subsystem as well as the modal parameters of the first two states $y_2(t)$ and $y'_2(t)$, of the second subsystem are calculated. The modal parameters for $y_1(t)$ and $y_2(t)$ are then used to calculate the unit step response, $g(t)$, of the system and the subsystem. From that, the periodic unit step response and the periodic square wave response are calculated. Finally the finite Fourier coefficients are calculated for the periodic unit step response.

4.1 Data Input

The input that the user must supply simulates the noise-free output of a system and its subsystem in the form of (2.24)

$$y(t) = c_0 + c_1 e^{s_1 t} + \cdots + c_n e^{s_n t}$$
This is justified in that the output samples of any system may be processed using Prony's method to yield the \( c \) coefficients and the \( s \) exponents of (2.24).

It should be noted that complex data entries must be entered in the form

\[
(x, y).
\]

The user is first prompted for the order of the first subsystem, then for the order of the second subsystem, and finally for the number of modes of subsystem \( S_1 \) that are coupled with subsystem \( S_2 \). After that the uncoupled coefficients of \( y_1(t) \) and their corresponding complex exponential values are entered from the keyboard. The coefficients and exponential terms of subsystem \( S_2 \) are then entered. For \( S_2 \), the coefficients and exponents for the coupled modes must be entered before those of the uncoupled modes.

For the coupled modes of the subsystems we can look at the coupled modes of \( S_1 \) as driving inputs to the subsystem. The coefficients for the coupled modes of \( S_1 \), \( cc \), are therefore related to the corresponding coefficients of \( S_2 \) by a constant term.

If \( S_1 \) has \( N_1 \) modes, of which \( N_s \) are states, and \( N_C \) are modes coupled with \( S_2 \), then

\[
N_1 = N_s + N_C. \tag{4.135}
\]

For \( S_2 \), all \( N_2 \) modes are states. (For \( S_2 \), \( N_S = N_2 \).)

We can now rewrite the output \( y_1(t) \) to separate the states from the coupled modes as

\[
y_1(t) = c_{10} + c_{11}e^{s_1 t} + \cdots + c_{1N_S}e^{s_{1N_S} t} + y_c(t), \tag{4.136}
\]

where

\[
y_c(t) = cc_{1}e^{\lambda_1 t} + \cdots + cc_{N_C}e^{\lambda_{N_C} t}, \tag{4.137}
\]
In (4.137), \( \lambda_i \) and \( cc_i \) are the exponents and coefficients of the coupled modes. As before, we can express \( y_2(t) \) as

\[
y_2(t) = c_0 + c_1 e^{\lambda_1 t} + \cdots + c_{2N_2} e^{\lambda_{2N_2} t}.
\]  

(4.138)

In this case, the following modes are coupled:

\[
\begin{align*}
S2(1) &= S1(N_S + 1) \\
S2(2) &= S1(N_S + 2) \\
&\quad \vdots \\
S2(N_C) &= S1(N_S + N_C) = S1(N_1)
\end{align*}
\]

(4.139)

The relationship between the coefficients of the coupled modes can now be determined,

\[
cc_j = \frac{d_{12}(\lambda_j)}{d_{22}(\lambda_j)} \cdot c_{2j}.
\]

(4.140)

where \( d_{12} \) and \( d_{22} \) are from (2.13) and are given by

\[
d_{22}(\lambda_j) = (\lambda_j - s_{11})(\lambda_j - s_{12}) \cdots (\lambda_j - s_{1N_S})
\]

(4.141)

\[
d_{12}(\lambda_j) = r_{N_C+1}(\lambda_j - r_1) \cdots (\lambda_j - r_{N_C}).
\]

(4.142)

The \( r_j \) values must also be entered from the keyboard when prompted for.

In order to evaluate the modal parameters of an \( n \)-th order system the system must be persistently excited over \( n \) intervals. To simulate this, the user must specify \( N_1 \) values of \( c_0 \), the steady-state output in (2.24). This has the effect of simulating a piecewise constant input to the system. For the estimation of the modal parameters the values of \( c_0 \) have little effect. However, the calculation of the system responses use the \( c_0 \) value for the first interval. Therefore, to obtain realistic 'unit' input responses, \( c_0 \), should be of order 1. As well, the duration of that input must be specified for each subsystem. This is done by choosing the duration of a half cycle of the interval \( w \) in (2.74). Finally, the modal delay \( t_d \), must be specified.
4.2 Noise Simulation

In order to simulate the presence of measurement noise in the system a pseudo-random Gaussian noise is produced and added to the noise-free output (4.136, 4.138). The following algorithm [8] was used to simulate the noise ($\sim N(0,1)$) and is implemented in the subroutine Random.

- Choose $R_0$ (between 0 and 1)

- $R_1 = 9821R + 0.211327$

- $R_1$ = fractional part of $R_1$

- $z = \sqrt{-2 \log R_1 \cos(2\pi R_1)}$

- $y = \sqrt{-2 \log R_1 \sin(2\pi R_1)}$

- $w = z + iy$

- $R_0 = R_1$

The value $W$ represents a complex random noise. The user is prompted for the noise variance and the initializing value of $R_0$, thus producing Gaussian noise of any variance by the following formula.

\[
\text{If } X \sim N(0,1) \quad \text{and } \quad Y \sim N(0,\sigma^2) \quad \text{then } \quad y = x\sigma.
\]
4.3 Modal Parameter Calculation

4.3.1 Setting up the Modal System

The modal parameters are calculated as described in section 2.1. A loop is set up to calculate the rows of the systems of (2.103) and (2.104). The number of rows equals the order of the system or the subsystem, that is $N_1$ or $N_2$.

The $(w_i, e^{s_j t_i}), i = 1, \ldots, N$ terms are calculated by the function Ex. Though some of these values may be complex, once they are scaled and summed to produce $(w_i(t), y_j^{(i)}(t)), i = 1, \ldots, N, j = 0, 1, 2$ the result is real. An additive Gaussian noise is added to all the inner product terms to simulate the effects of noise in the system.

The elements of the matrix of delayed inner product terms (2.103) are evaluated using (2.90).

This almost completes the calculations for the first interval $w_i$. What remains to be done is to make a time shift to account for the modal time constraint given in (2.62).

The beginning of the second interval is taken to be the order of the subsystem plus the length of the sampling interval, $w_i$, measured in units of time delays, $t_d$. The delay corresponding to the number of time delays times the order of the subsystem is chosen to meet the modal time constraint.

The total interval delay, $t_{1i} = (N_i + 2t_{2i})t_d$, where $t_{1i}$ is the beginning of the next interval, and $t_{2i}$ is half the duration of $w_i$, is evaluated for both subsystems, and the maximum is used as $t_{NT}$. This ensures greater sampled signal levels for $y(t)$ (see (2.24)), as $y(t)$ decays exponentially with time.

Before we start the coefficient updating we first save a copy of the original coefficients to be used in evaluating the modal constant. The updating itself is done by
Figure 4.3: Intervals, \( w_i \), used to evaluate the modal parameters for the two subsystems.
making the following observations. If over the first interval \( 0 < t < t_1 \),

\[
y(t) = c_0 + c_1 e^{s_1 t} + \cdots + c_n e^{s_n t}
\]  

then

\[
y(t) - c_0 = c_1 e^{s_1 t} + \cdots + c_n e^{s_n t}
\]

and

\[
\frac{dy}{dt} = c_1 s_1 e^{s_1 t} + \cdots + c_n s_n e^{s_n t}
\]

\[
\vdots
\]

\[
\frac{d^{n-1}y}{dt^{n-1}} = c_1 s_1^{n-1} e^{s_1 t} + \cdots + c_n s_n^{n-1} e^{s_n t}
\]

by evaluating \( y(t), \frac{dy}{dt}, \ldots, \frac{d^{n-1}y}{dt^{n-1}} \) at \( t = t_1 \) we obtain the updated coefficients for the second interval \( \hat{c}_i \):

\[
\hat{c}_1 + \cdots + \hat{c}_{N_S} = c_{10} + c_1 e^{s_1 t_1} + \cdots + c_{N_S} e^{s_{N_S} t_1} - \lambda_1 \hat{c}_1 - \cdots - \lambda_{N_C} \hat{c}_{N_C}
\]

\[
s_1 \hat{c}_1 + \cdots + s_{N_S} \hat{c}_{N_S} = s_1 c_1 e^{s_1 t_1} + \cdots + s_{N_S} c_{N_S} e^{s_{N_S} t_1} - \lambda_1 \hat{c}_1 - \cdots - \lambda_{N_C} \hat{c}_{N_C}
\]

\[
\vdots
\]

\[
s_1^{N_S-1} \hat{c}_1 + \cdots + s_{N_S}^{N_S-1} \hat{c}_{N_S} = s_1^{N_S-1} c_1 e^{s_1 t_1} + \cdots + s_{N_S}^{N_S-1} c_{N_S} e^{s_{N_S} t_1} - \lambda_1^{N_S-1} \hat{c}_1 - \cdots - \lambda_{N_C}^{N_S-1} \hat{c}_{N_C}.
\]

where the \( \hat{c}_i \) for S1 are calculated before the actual updating from the updated coefficients of S2. The S2 coefficients are updated before the S1 coefficients. Note, that for S2, \( N_S = N_2 \) and \( N_C = 0 \).

Therefore we can set up the following Vandermonde system to solve for the \( \hat{c}_i \) coefficients for the next interval

\[
\begin{bmatrix}
1 & 1 & \cdots & 1 \\
\vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \ddots & \vdots \\
\end{bmatrix}
\begin{bmatrix}
\hat{c}_1 \\
\vdots \\
\hat{c}_n \\
\end{bmatrix}
= 
\begin{bmatrix}
y(t_1) - c_0 - \hat{c}_1 - \cdots - \hat{c}_{N_C} \\
\frac{dy}{dt} |_{t=t_1} - \lambda_1 \hat{c}_1 - \cdots - \lambda_{N_C} \hat{c}_{N_C} \\
\vdots \\
\frac{d^{n-1}y}{dt^{n-1}} |_{t=t_1} - \lambda_1^{N_S-1} \hat{c}_1 - \cdots - \lambda_{N_C}^{N_S-1} \hat{c}_{N_C}
\end{bmatrix}
\]

(4.146)
where \( n = N_s \).

The coefficient updating is done by the subroutine Update and the Vandermonde system is solved by the subroutine Vandm which is based on algorithm 5.6-2 in [9].

### 4.3.2 Obtaining the Modal Parameters

The modal parameters for the first states are obtained by solving the matrix equation (2.103) by the use of the subroutine Dslimp [14]. The subroutine performs Gaussian elimination in double precision arithmetic by the use of an iterative routine. It continues the iterations until one of three events occur.

- the error vector calculated by the subroutine is less than a preset tolerance, the variable EPS.
- the iterations start to diverge.
- until a preset maximum number of iterations is performed.

The subroutine Check prints out the modal parameters, the value of EPS, which indicates the accuracy of the solution returned by Dslimp, and evaluates the equations of (2.81) to ensure that the equality holds.

### 4.3.3 Evaluating the Modal Constant

The evaluation of the \( M_{i0} \) term is done by the subroutine Const which evaluated (2.109) to obtain \( M_{i0} \).

The \( \langle w(t), e^{it} \rangle \) terms are calculated by the function Wo where \( w(t) \) is given by (2.105). The function Woy in turn, uses Wo to evaluate the \( \langle w(t), y(t - it_d) \rangle, i = 1, \ldots, N \) terms of (2.109) and to calculate \( M_{i0} \).
4.3.4 Finding the Modal Parameters for the Second and Third States

With the work that had to be done to calculate the modal parameters of the first state of the system and the subsystem the calculation of the modal parameters of the second and third state is relatively trivial.

The matrix of (2.104) is the matrix of (2.103) but shifted one column to the right. The $nt_d$ delayed terms of (2.103) drop out and the first column of (2.104) is simply the non-delayed inner product terms of (2.103).

Once this shifting is complete the modal parameters are obtained by the same procedure of the subroutines Dsli2p, Check, and Const described above.

4.4 Calculating the System Responses

Program Modal calculated three different responses of the system and the subsystem:

- the unit step response.

and through the process of modal filtering:

- the periodic unit step response.

- the periodic unit square wave, sal(1, x), response.

4.4.1 Unit Step Response

The unit step responses for the system and the subsystem are calculated using the theory outlined in chapter 3. The user is prompted for the number of values of the unit step response desired, up to fifty points. The response is sampled at intervals $t_d$.

The function $Y$ is used to evaluate $y(t)$ and $Y(t, n)$ is calculated using (2.33) which gives $G(t, n)$ using (3.117). For purposes of convenience and without loss of generality
it has been decided to let \( v(t) = y(t), t < 0 \). Then since

\[
y(t) = v(t) + g(t)\Delta u(1),
\]

\[
y(t) = \begin{cases} v(t), & t < 0 \\ 2v(t), & t > 0 \end{cases}
\]

(4.147)

due to the fact that \( g(t) = 0, t < 0 \). Then \( g(t) \) is evaluated using the recursive relationship of (3.119).

### 4.4.2 Periodic Unit Step Response

The periodic unit step response, an odd symmetric function of period \( 2\sigma = 2(N + 1)\ell_d \), is calculated recursively from (3.118).

### 4.4.3 Periodic Unit Square Wave Response

The response to an input of the Walsh function, \( \text{sal}(1,x) \), is also odd, symmetric, and of period \( 2\sigma \). The response is given by (3.132).

In order to generate the \( R_{s_1}(x,n) \) values we must know the \( \gamma_i \) parameters. The subroutine Incoef derives the \( \gamma \) parameters from the modal parameters. Incoef sets up the matrix \( C \) in (3.127) and then calls upon the subroutine Finv [14] that inverts \( C \). From \( C^{-1} \) the \( \gamma_i \) parameters can be read off the first column of \( C^{-1} \).

For both the periodic unit step response and the periodic unit square wave response only the \( N+1 \) values from \( t = 0 \) to \( t = \sigma \) are calculated. The other \( N+1 \) values are generated form these.
4.4.4 Calculation of the Finite Fourier Transform Coefficients

To calculate the finite Fourier transform coefficients of the periodic unit step response we use the subroutines Prefft and Fft [12] to implement a fast Fourier transform algorithm on one period of $G_p(t, n)$. Since the algorithm only works for a number of data points equal to $2^k$, we must pass to the subroutine more than one period of data values. In the subroutine Setfft, which calls Prefft and Fft, the values of one complete period of $G_p(t, n)$ is stored in an array. Then subsequent values from the next period are added to the array until the total number of values is $2^k$. These values are then used by the fast Fourier transform subroutines.
Chapter 5

Modal Parameter Estimation with Noise

In any reasonable, physical system there is always some uncertainty in readings or measurements made on the outputs of the system. In the program Modal, it is possible to add a zero-mean Gaussian noise to the system under consideration and observe the effects of the noise on the modal parameter estimation and non-parametric system identification. However, it is beyond the scope of the present program to implement any process of statistical averaging or other technique to attempt to reduce the effect of the noise. What the PC program does is to give a measure of the 'sensitivity' of the various modal function techniques to an additive noise. This can be of assistance in estimating the number of data intervals required to achieve a given accuracy in the presence of noise. The use of more sophisticated techniques to reduce or eliminate the effects of noise will have to be dealt with at a future time.

5.1 Sensitivity Analysis of Modal Parameters

The addition of noise to the systems modelled by the program Modal has the effect of giving the user an indication of the sensitivity of the modal parameter estimates to an additive Gaussian noise. During the process of modal parameter estimation we are required to sample the outputs of the system. The measurement of output will contain an uncertainty due in part to the inherent uncertainty of the measurement device. This uncertainty can be best modelled as an additive, random noise with uniform distribution, centered at zero and with range \([-W, W]\), where $W$ is the accuracy of the
measurement device.

\[
f_x(x) = \begin{cases} 
  k, & -W \leq x \leq W \\
  0, & \text{otherwise}
\end{cases}
\]  

(5.148)

In practice, the inner product terms used to estimate the modal parameters must be evaluated by some technique of numerical integration such as the trapezoidal rule (see section 5.3). In evaluating the trapezoidal approximation to the integral, a number of data samples must be summed. Since all of the data samples are corrupted by an additive uniformly distributed noise, the central limit theorem [18], says that the final approximate integration will be corrupted by an additive Gaussian noise.

To illustrate the effects of the addition of additive Gaussian noise to a signal it will be demonstrated what happens when noise is added to a first order signal. (The result may be easily generalized to the case of a n-th order signal by the use of matrix notation.)

Consider the case of

\[y(t) = c_0 + c_1 e^{st}.\]  

(5.149)

Taking the inner products of (5.149) with the interval \(w(t)\) as in (2.74) we obtain

\[
\langle w(t), y(t) \rangle = c_1 \langle w(t), e^{st} \rangle = c_1 E,
\]  

(5.150)

where

\[
E = \frac{1}{s} \left[ 2e^{st_d} - e^{s(t+d)} - e^{s(t-d)} \right].
\]  

(5.151)

From (5.150) we obtain the modal function representation of (5.149),

\[y(t) + M_1 y(t - t_d) = M_0.\]  

(5.152)

For the noise free case it is observed that

\[
\langle w(t), y(t) \rangle + M_1 \langle w(t), y(t - t_d) \rangle = 0
\]
Chapter 5. Modal Parameter Estimation with Noise

\[
c_1 E + c_1 E M_1 e^{-st_d} = 0
1 + M_1 e^{-std} = 0 \quad (5.153)
\]

where (5.153) is the definition of \( M_1 \). If noise is added to the system, the following is seen,

\[
\ddot{y}(t) = y(t) + n(t) \quad (5.154)
\]

\[
\langle w(t), \ddot{y}(t) \rangle = \langle w(t), y(t) \rangle + \bar{n}, \quad (5.155)
\]

and the following data equation is obtained,

\[
c_1 E + \bar{n} + \dot{M}_1 \left[ c_1 e^{-std} E + \bar{n} \right] = 0. \quad (5.156)
\]

If (5.156) is solved for the change in modal parameter estimates caused by noise, \( M_1 - \dot{M}_1 \), it is seen that

\[
(M_1 - \dot{M}_1)c_1 e^{-std} E = \left[ 1 + \dot{M}_1 \right] \bar{n}
M_1 - \dot{M}_1 = \bar{n} \left[ 1 + \dot{M}_1 \right] E^{-1} e^{std} c_1^{-1} \quad (5.157)
\]

where \( \frac{\bar{n}}{c_1} \equiv \text{the noise to signal ratio} \).

To use a numerical example choose \( a=1, b=2, c=3, t_d=0.1, \) and \( s=-3 \). Then

\[
E = \frac{1}{3} \left[ 2e^{-0.6} - e^{-0.3} - e^{0.9} \right] = -0.01659
\]

\[
E^{-1} e^{std} = E^{-1} e^{-0.3} = -44.659
\]

\[
M_1 = -e^{-0.3} = -0.74082
\]

Therefore the relative change in \( M_1 \) due to noise is

\[
\frac{M_1 - \dot{M}_1}{M_1} \equiv \left( \frac{\bar{n}}{c_1} \right) \left[ 1 + \frac{1}{M_1} \right] (-44.659) = 15.62 \left( \frac{\bar{n}}{c_1} \right) \quad (5.158)
\]

where \( M_1 \) has been used instead of \( \dot{M}_1 \) in the \( \left[ 1 + \frac{1}{M_1} \right] \) term.
It can be seen that the term that most influences the sensitivity of $M_1$ is the $E^{-1}e^{st_d}$ term that is quite large in this case. This demonstrates the importance of choosing the factors that make up the $E^{-1}e^{st_d}$ term; the sample interval, $t_d$, and the actual form of the interval $w(t)$. For these preliminary investigations, $w(t)$ was chosen simply to eliminate the constant $M_0$ terms and $t_d$ was completely arbitrary. Clearly the choice of these parameters is an area that deserves closer study. Equation (5.157) is suitable for sensitivity studies and for non-statistical averaging of the minimum number of intervals.

In (5.158) it is seen that if sensitivity of less than 1% is desired, we require

$$15.62 \left( \frac{n}{c_1} \right) = 0.01$$

and if $c_1 \approx 1$, then

$$n = \frac{0.01}{15.62} = 6.40 \times 10^{-4}$$

5.2 Statistical Model for the Simulation of Noise

The general vector form of (5.157) is given by

$$M - \tilde{M} = \bar{n}(e + \tilde{M})(V_mEC)^{-1}$$

$$= \gamma \bar{n}(e + \tilde{M}) \Gamma$$

(5.159)

where $\Gamma = (V_mEC)^{-1}$ and the factor $\gamma$ is chosen such that the elements of matrix $\Gamma$ are of order unity. It is clear from (5.158) and (5.159) that some form of averaging must be used to reduce the effect of $\gamma \bar{n}$ on the estimate.

Statistical ensemble averaging yields, since $E[\bar{n}] = 0$,

$$E[\tilde{M}] = M - \gamma E[\bar{n}\tilde{M}] \Gamma.$$

(5.160)

Multiplying (5.159) by $\bar{n}$ yields

$$-E[\bar{n}\tilde{M}] = \gamma \sigma^2 (e + M) \Gamma,$$

(5.161)
where \( \bar{\sigma}^2 = E[\tilde{n}^2] \) and where the approximation \( E[\tilde{n}^2 \bar{M}] = \bar{\sigma}^2 M \) is used. Substituting (5.161) into (5.160) yields

\[
E[\bar{M}] = M + (\gamma \bar{\sigma})^2 (e + M) \Gamma^2.
\] (5.162)

Equation (5.162) clearly illustrates the parameter bias term that is common to all least-square estimation procedures. The standard methods of eliminating bias is to use two-stage least-squares or maximum-likelihood estimation, or instrumental variables. In the modal function approach, due to the elimination of transients, the control input is a natural instrumental variable. Furthermore, since continuous-time data is sampled, integration can be used as a form of data compression and noise reduction. This is of importance since \( \gamma \) may be large (see (5.158)).

Equation (5.162) is a standard least-squares form showing parameter bias and can be implemented in a computer program.

It follows from the discussion given in section 5.3 that the reduced effective noise variance is determined by

\[
\bar{\sigma} = \frac{\sigma}{\beta \sqrt{N_1 N_2}},
\] (5.163)

where \( \beta \) is an attenuation factor associated with a high-frequency analog noise filter, \( N_1 \) = number of samples used for integration, and \( N_2 \) = number of intervals used.

**Example:** Let, \( \sigma = 0.1 \), and \( \gamma = 100.0 \). Then

\[
(\gamma \bar{\sigma}) = 10^{-2}
\]

If it is desired that

\[
\beta \sqrt{N_1 N_2} = 3.1 \times 10^2,
\]

then one possible choice of \( N_1, N_2 \), and \( \beta \) is that \( N_1 = N_2 = 100 \), and \( \beta = 3.1 \).
5.3 Noise Reduction Through Numerical Integration

To reduce the sensitivity of modal parameter estimates to the factor $\gamma \sigma$ it is desirable to use preprocessing reduce the variance of $n$ by as much as possible. It can be shown that by using the process of trapezoidal approximation that the noise variance may be reduced to an arbitrarily small level.

Assume that the system output, $y(t)$, is corrupted by an additive uniform noise so that

$$\tilde{y}(t) = y(t) + n(t). \quad (5.164)$$

Therefore, for an interval, $w(t)$, as shown in figure 5.4,

$$\langle w(t), \tilde{y}(t) \rangle = \langle w(t), y(t) \rangle + \langle w(t), n(t) \rangle$$

$$= \langle w(t), y(t) \rangle + \tilde{\nu}. \quad (5.165)$$

The first term of (5.165) can be evaluated either analytically or by trapezoidal approximation as shown previously but the $\tilde{\nu}$ term must be evaluated by an approximation method since we cannot model $n(t)$ as a sum of weighted exponentials.
Evaluating $\bar{v}$ we get
\[
\bar{v} = \frac{\Delta}{2} [\nu(t_1) + 2(\nu(t_1 + \Delta) + \cdots + \nu(t_1 + (N - 1)\Delta)) + \nu(t)]
\]
\[
= -\frac{\Delta}{2} [\nu(t) + 2(\nu(t + \Delta) + \cdots + \nu(t + (N - 1)\Delta)) + \nu(t_2)]
\]
\[
= \frac{\Delta}{2} [\nu(t_1) + 2(\nu(t_1 + \Delta) + \cdots + \nu(t_1 + (N - 1)\Delta) + 0
\]
\[
+ \nu(t + \Delta) + \cdots + \nu(t + (N - 1)\Delta)) + \nu(t_2)]
\]  
(5.166)

Since each $\nu(t)$ term is a uniformly distributed random variable and we are adding up 2N of them, we can, for sufficiently large N, say that $\bar{v} \sim N(0, \sigma^2)$ due to the central limit theorem [18].

The next step is to evaluate $E[\bar{v}(t_1)\bar{v}^T(t_1)]$. Let $E[\nu(t_1)\nu^T(t_1)] = \sigma^2 I$. Then
\[
E[\bar{v}\bar{v}^T] = \left( \frac{\Delta}{2} \right)^2 \left[ \sigma^2 + 2 \left( \sigma^2 + \cdots + \sigma^2 + 0 + \sigma^2 + \cdots + \sigma^2 \right) + \sigma^2 \right] I
\]
\[
= \left( \frac{\Delta \sigma}{2} \right)^2 [2 + 4(N - 1)] I
\]
\[
= \Delta^2 \sigma^2 \left[ N - \frac{1}{2} \right] I
\]
(5.168)

From (5.168) we can see that if we choose enough partitions for the trapezoidal approximation, $\Delta$ will become as small as we like and $E[\bar{v}\bar{v}^T]$ will be as close to zero as we wish. So we can, with a sufficient number of samples, reduce the noise variance in estimating modal parameters.

5.4 Further Concerns Associated with Modal Functions

There has been much work in recent years dealing with the fitting of exponentials and specifically dealing with Prony’s method. Since the method of modal functions is derived from Prony’s method, these works have a direct bearing on this thesis.
A basic consideration is whether techniques based on Prony's method perform any better than the older, more traditional FFT approach. Bucker [5] and Spitznogle and Quazi [22] compared the two techniques and found that in some cases, Prony's method was superior to the FFT approach. In particular, Prony's method gave better resolution in the estimation of poles and behaved better in low-noise situations.

Van Blaricum and Mittra [24] showed how the poles and residues of a system can be extracted directly from its transient response when the number of poles are known. However, in the case where the number of poles or the order of the system is unknown it is necessary to estimate the order of the system before applying either Prony's method or modal parameter estimation. Holt and Antill [10] used singular value decomposition (SVD) to determine the number of terms to use in Prony's method by observing the singular values of an equally spaced data matrix. If a system is of order n, then the singular values of a data matrix of order greater than n should become quite small after the n-th singular value. Van Blaricum and Mittra [25] used standard deviations to determine the number of poles of a system when noise was present.

Another area of interest concerning modal parameter estimation is what an optimum sampling period would be. Crittenden et al. [6] investigated the effect of sampling period on Prony's method when identifying state parameters of a continuous time system.

The problem of what constitutes 'persistently excited inputs' has been looked at by Schaubert [20] in the case of step-like excitation. When using modal functions, the $c_0$ coefficients and the intervals, $w(t)$, represent a form of pseudo-random step-like excitation but just what would be an optimum input must be investigated further.

To deal with noise two procedures seem to have been put forward. The first involves the modelling of a system by a higher order model. By using forward-backward linear prediction (BFLP) the actual poles of a system may be distinguished from those poles
due to noise by superimposing the results on a unit circle. While the legitimate poles are mirrored across the unit circle for the two cases, the noise poles remain constant (Kumaresan and Tufts [11] and Tufts and Kumaresan [23]). Poggio et al. [16] also investigated the use of rank-overspecification and filtering on noise.

The second method consists of using SVD to decompose a noisy data matrix into a data matrix and a noise matrix. This is done by observing the singular values of the noisy data matrix and constructing a new data matrix by excluding the smaller singular values that are assumed to be caused by noise (Holt and Antill [10], Kumaresan and Tufts [11], and Tufts and Kumaresan [23]). Both of these methods could be useful for the estimation of modal parameters and this should be investigated further.
Chapter 6

Discussion and Conclusions

In this thesis, it has been shown that the method of modal functions offers a new and very useful tool for both parametric and non-parametric system identification. Modal functions are independent of system initial conditions and allow the representation of not only a signal, but also its time-derivatives by a simple continuous-time ARMA model. For the case of a MIMO system, the number of parameters that must be estimated is greatly reduced by being able to decouple the larger system into smaller subsystems. This increases the speed and simplifies the parameter estimation problem. The actual estimation of the parameters is carried out through the use of simple matrix algebra and, in practical cases, by using trapezoidal approximation to calculate integral terms, it is possible to reduce the noise variance.

Through the use of modal functions we can directly identify the system unit step response and the square wave response directly from input-output data. As well, modal filtering can produce a periodic unit step response which may be used to find FFT coefficients without the use of window functions. The identification of the periodic square-wave response from non-periodic data may be of use in the field of signal processing.

The PC program Modal will enable the user to investigate the above processes while varying the different external parameters in an easy and efficient manner and will further the understanding of modal functions.
Bibliography


Appendix A

A Trial Run of Program Modal

To demonstrate the use of the program Modal the following example is given. The following system of one input and two outputs was simulated.

\[
y_1(t) = c_{10} + 100e^{-t} - 100e^{-2t} + 100e^{-3t}
\]

\[
y_2(t) = c_{10} - 100e^{-3t} + 100e^{-2t},
\]

For this system, it can be seen that subsystem S1 consists of the modes \((-1, -2, -3)\) and that subsystem S2 consists of the modes \((-3, -2)\) (ie. the third mode of S1 is coupled with the first mode of S2). The factored roots of \(d_{12}(y_2)\) were given as \((-1, -1)\).

For values of \(c_{10}\), the following were chosen for each interval: 100, \(-100\), 0. This corresponds to a control input of a unit-impulse for interval 1, and unit-steps for intervals 2 and 3 (see figure A.5). The integration intervals, \(w_i(t), t = 1, 2, 3\), were defined by the duration of half of the interval, and were chosen to be (see figure A.6)

- for S1: 8, 8, 4.
- for S2: 8, 8.

The sampling period was chosen to be 0.1 seconds and a value of 0.0001 was chosen for VAR, the reduced effective noise variance (see 5.163).

What follows next are the prompts and messages displayed on the PC monitor (in typewriter print) and the proper responses from the user.

Modal parameter estimation program.
First, sample data must be generated.

Each of the 2 outputs is assumed to be a sum of complex exponentials.

Subsystems: S1 has 3 states and S2 has 2 states.

Enter # of modes of subsystem S1.

3

Enter # of modes of subsystem S2.

2

Enter the number of uncoupled modes of S1.

2

All complex entries must be of the form (x,y).

Subsystem S1, uncoupled modes only!

Enter real coefficient 1

100
Figure A.6: Intervals for example A.
Enter real coefficient 2
-100
Enter complex exponential 1
(-1,.0.)
Enter complex exponential 2
(-2,.0.)
Subsystem S2: Enter coupled modes before uncoupled!
Enter real coefficient 1
-100
Enter real coefficient 2
100
Enter complex exponential 1
(-3,.0.)
Enter complex exponential 2
(-2,.0.)
Enter the roots of the char. equation, r(i).
Enter complex root, R(1).
(-1,.0.)
Enter complex root, R(2).
(-1,.0.)
To identify the modal parameters of S1, S2 you must specify 3 intervals. Specify the length of each half-cycle interval as a multiple of T.
Enter CO value for interval 1
100
Enter CO value for interval 2
Appendix A. A Trial Run of Program Modal

-100
Enter CO value for interval 3
0
Subsystem 1
Enter the half-cycle duration of interval 1
8
Enter the half-cycle duration of interval 2
8
Enter the half-cycle duration of interval 3
4
Subsystem 2
Enter the half-cycle duration of interval 1
8
Enter the half-cycle duration of interval 2
8
Enter the sampling period, T, in seconds.
.1
Enter the noise variance.
0.0001
Random number generator initialization.
Enter R(0); (between 0 and 1)
.78478143

This completes the information that the user must enter from the PC keyboard. The output is displayed on the screen and stored in a file named MODAL.LIS. For this example the file MODAL.LIS is shown below.
Modal Parameter Estimation Program.

Data generation parameters:

Subsystem 1 has 3 states and 3 modes.
Subsystem 2 has 2 states and 2 modes.

\[
\begin{array}{llll}
C_1(1) &= 1.00000E+03 & 0.00000E+00 & C_1(2) = -1.00000E+03 & 0.00000E+00 \\
C_1(3) &= 1.00000E+03 & 0.00000E+00 & C_1(3) = 0.00000E+00 \\
C_2(1) &= -1.00000E+03 & 0.00000E+00 & C_2(2) = 1.00000E+03 & 0.00000E+00 \\
S_1(1) &= -1.00000E+01 & 0.00000E+00 & S_1(2) = -2.00000E+01 & 0.00000E+00 \\
S_1(3) &= -3.00000E+01 & 0.00000E+00 & S_1(3) = 0.00000E+00 \\
S_2(1) &= -3.00000E+01 & 0.00000E+00 & S_2(2) = -2.00000E+01 & 0.00000E+00 \\
R(1) &= -1.00000E+01 & 0.00000E+00 & R(2) = -1.00000E+01 & 0.00000E+00 \\
\end{array}
\]

CO for interval 1 = 1.00000E+03
CO for interval 2 = -1.00000E+03
CO for interval 3 = 0.00000E+00

Subsystem 1

Interval # 1 has a half cycle of $8T$ s.
Interval # 2 has a half cycle of $8T$ s.
Interval # 3 has a half cycle of $4T$ s.

Subsystem 2

Interval # 1 has a half cycle of $8T$ s.
Interval # 2 has a half cycle of $8T$ s.

The sampling period is $100E+00$ seconds.
The additive noise $\sim N(0, .0001)$

New C parameters for interval 2

\[
\begin{align*}
C1(1) &= .814957E+03 \quad .000000E+00 \\
C1(2) &= -.100224E+04 \quad .000000E+00 \\
C1(3) &= .400335E+03 \quad .000000E+00 \\
C2(1) &= -.400335E+03 \quad .000000E+00 \\
C2(2) &= .602237E+03 \quad .000000E+00
\end{align*}
\]

New C parameters for interval 3

\[
\begin{align*}
C1(1) &= -.278108E+03 \quad .000000E+00 \\
C1(2) &= .477579E+03 \quad .000000E+00 \\
C1(3) &= -.198660E+03 \quad .000000E+00 \\
C2(1) &= .198660E+03 \quad .000000E+00 \\
C2(2) &= -.286527E+03 \quad .000000E+00
\end{align*}
\]

The modal parameters, M 1 are:

\[
\begin{align*}
(1) &= -.252332E+01 \quad (2) = .212513E+01 \\
(3) &= -.596610E+00 \\
\end{align*}
\]

Checking!

EPS adds up to .185583E-14

The modal identity adds up to, -.313777E-05

The modal identity adds up to, -.370271E-05

The modal identity adds up to, -.124320E-05

The M 1 constant is .548101E+00.

The modal parameters, M 4 are:

\[
\begin{align*}
(1) &= -.156070E+01 \quad (2) = .607591E+00 \\
\end{align*}
\]

Checking!
EPS adds up to \( .000000E+00 \)

The modal identity adds up to, \( .613459E-07 \)

The modal identity adds up to, \( .931981E-06 \)

The M 4 constant is \( .468830E+01 \).

O***ITERATIVE IMPROVEMENT IS DIVERGING

The modal parameters, M 2 are:

\[
(1) = -1.28752E+02 \quad (2) = 1.61173E+02 \\
(3) = -3.22496E+01
\]

Checking!

EPS adds up to \(-.346528E-13\)

The modal identity adds up to, \(-.315092E-05\)

The modal identity adds up to, \(-.301753E-04\)

The modal identity adds up to, \(-.474889E-05\)

The M 2 constant is \( .169810E+01 \).
O***ITERATIVE IMPROVEMENT IS DIVERGING

The modal parameters, M 3 are:
(1) = -.422736E+02 (2) = .942608E+02
(3) = -.515318E+02

Checking!
EPS adds up to -.568457E-12
The modal identity adds up to, -.705720E-05

The modal identity adds up to, .573695E-06

The modal identity adds up to, -.385358E-05

The M 3 constant is .449344E+02.

O***ITERATIVE IMPROVEMENT IS DIVERGING

The modal parameters, M 5 are:
(1) = -.753411E+01 (2) = .780764E+01

Checking!
EPS adds up to -.743869E-14
The modal identity adds up to, -.689855E-06

The modal identity adds up to, -.111486E-04
Appendix A. A Trial Run of Program Modal

The $M_5$ constant is \( .273397E+02 \).

The first 10 values of the step response are:

Subsystem 1

<table>
<thead>
<tr>
<th>$G(1)$</th>
<th>$G(2)$</th>
<th>$G(3)$</th>
<th>$G(4)$</th>
<th>$G(5)$</th>
<th>$G(6)$</th>
<th>$G(7)$</th>
<th>$G(8)$</th>
<th>$G(9)$</th>
<th>$G(10)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>( .199878E+03 )</td>
<td>( .182303E+03 )</td>
<td>( .168947E+03 )</td>
<td>( .158627E+03 )</td>
<td>( .150512E+03 )</td>
<td>( .144020E+03 )</td>
<td>( .138739E+03 )</td>
<td>( .134377E+03 )</td>
<td>( .130724E+03 )</td>
<td>( .127627E+03 )</td>
</tr>
</tbody>
</table>

Subsystem 2

<table>
<thead>
<tr>
<th>$G(1)$</th>
<th>$G(2)$</th>
<th>$G(3)$</th>
<th>$G(4)$</th>
<th>$G(5)$</th>
<th>$G(6)$</th>
<th>$G(7)$</th>
<th>$G(8)$</th>
<th>$G(9)$</th>
<th>$G(10)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>( .999805E+02 )</td>
<td>( .107748E+03 )</td>
<td>( .112087E+03 )</td>
<td>( .114145E+03 )</td>
<td>( .114727E+03 )</td>
<td>( .114386E+03 )</td>
<td>( .113503E+03 )</td>
<td>( .112334E+03 )</td>
<td>( .111046E+03 )</td>
<td>( .109747E+03 )</td>
</tr>
</tbody>
</table>

The 4 values of the periodic step response are

Subsystem 1

<table>
<thead>
<tr>
<th>$GP(1)$</th>
<th>$GP(2)$</th>
<th>$GP(3)$</th>
<th>$GP(4)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>( .199604E+03 )</td>
<td>( -.322326E+03 )</td>
<td>( .133432E+03 )</td>
<td>( .213424E+00 )</td>
</tr>
</tbody>
</table>

Subsystem 2

<table>
<thead>
<tr>
<th>$GP(1)$</th>
<th>$GP(2)$</th>
<th>$GP(3)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>( .976363E+02 )</td>
<td>( -.506359E+02 )</td>
<td>( .232760E+01 )</td>
</tr>
</tbody>
</table>

The Periodic Square Wave Response is given by:

Subsystem 1
Appendix A. A Trial Run of Program Modal

<table>
<thead>
<tr>
<th>Subsystem 1</th>
<th>Subsystem 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>GAMMA(1) = -0.253115E+01</td>
<td>GAMMA(1) = 0.474376E-01</td>
</tr>
<tr>
<td>GAMMA(2) = -0.138587E+01</td>
<td>GAMMA(2) = 0.109263E+01</td>
</tr>
<tr>
<td>GAMMA(3) = 0.409452E+00</td>
<td>GAMMA(3) = 0.167645E+01</td>
</tr>
<tr>
<td>GAMMA(4) = 0.246822E+01</td>
<td>GAMMA(4) =</td>
</tr>
</tbody>
</table>

The 4 values of the per. square wave resp. are

| RS(1) = 0.236009E+03 | RS(1) = 0.869770E+02 |
| RS(2) = 0.209802E+03 | RS(2) = 0.100376E+03 |
| RS(3) = 0.190169E+03 | RS(3) = 0.108466E+03 |
| RS(4) = 0.175230E+03 | RS(4) = |

FFT coefficients of Gp(t).

<table>
<thead>
<tr>
<th>Subsystem 1</th>
<th>Subsystem 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1) = 0.000000E+00</td>
<td>(1) = 0.470004E+01</td>
</tr>
<tr>
<td>(2) = -0.569322E+01</td>
<td>(2) = 0.460756E+01</td>
</tr>
<tr>
<td>(3) = 0.000000E+00</td>
<td>(3) = 0.483083E+01</td>
</tr>
<tr>
<td>(4) = 0.188673E+00</td>
<td>(4) = -0.953087E+01</td>
</tr>
<tr>
<td>(5) = 0.475347E+02</td>
<td>(5) = -0.279153E+01</td>
</tr>
<tr>
<td>(6) = 0.162702E+02</td>
<td>(6) = 0.479253E+01</td>
</tr>
<tr>
<td>(7) = 0.162702E+02</td>
<td>(7) = 0.953087E+01</td>
</tr>
<tr>
<td>(8) = 0.188673E+00</td>
<td>(8) = 0.460756E+01</td>
</tr>
</tbody>
</table>
Appendix A. A Trial Run of Program Modal

Table A.1: Summary of the modal parameter estimates for example 1.

The modal parameter estimates are summarized in table A.1. The non-parametric system responses for subsystem S1 are summarized in table A.2 and for subsystem S2 in table A.3.

The FFT coefficients of the periodic unit step response are summarized in table A.3.

The system responses for S1 and S2 are graphed in figures A.7 and A.8 respectively.
### System Responses, Subsystem S1

<table>
<thead>
<tr>
<th>Time</th>
<th>$G(t, n)$</th>
<th>$G_p(t, n)$</th>
<th>$R_{z1}(t, n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>199.878</td>
<td>199.604</td>
<td>236.009</td>
</tr>
<tr>
<td>0.1</td>
<td>182.303</td>
<td>-322.326</td>
<td>209.802</td>
</tr>
<tr>
<td>0.2</td>
<td>168.947</td>
<td>133.432</td>
<td>190.169</td>
</tr>
<tr>
<td>0.3</td>
<td>158.627</td>
<td>0.213424</td>
<td>175.230</td>
</tr>
<tr>
<td>0.4</td>
<td>150.512</td>
<td>-199.604</td>
<td>-236.009</td>
</tr>
<tr>
<td>0.5</td>
<td>144.020</td>
<td>322.326</td>
<td>-209.802</td>
</tr>
<tr>
<td>0.6</td>
<td>138.739</td>
<td>-133.432</td>
<td>-190.169</td>
</tr>
<tr>
<td>0.7</td>
<td>134.377</td>
<td>-0.213424</td>
<td>-175.230</td>
</tr>
<tr>
<td>0.8</td>
<td>130.724</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.9</td>
<td>127.627</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table A.2: Summary of non-parametric system responses for subsystem S1.

### System Responses, Subsystem S2

<table>
<thead>
<tr>
<th>Time</th>
<th>$G(t, n)$</th>
<th>$G_p(t, n)$</th>
<th>$R_{z1}(t, n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>99.9805</td>
<td>97.6363</td>
<td>86.9770</td>
</tr>
<tr>
<td>0.1</td>
<td>107.748</td>
<td>-50.6359</td>
<td>100.376</td>
</tr>
<tr>
<td>0.2</td>
<td>112.087</td>
<td>2.32760</td>
<td>108.466</td>
</tr>
<tr>
<td>0.3</td>
<td>114.145</td>
<td>-97.6363</td>
<td>-86.9770</td>
</tr>
<tr>
<td>0.4</td>
<td>114.727</td>
<td>50.6359</td>
<td>-100.376</td>
</tr>
<tr>
<td>0.5</td>
<td>114.386</td>
<td>-2.32760</td>
<td>-108.466</td>
</tr>
<tr>
<td>0.6</td>
<td>113.503</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.7</td>
<td>112.334</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.8</td>
<td>111.046</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.9</td>
<td>109.747</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Table A.3: Summary of non-parametric system responses for Subsystem S1.
Figure A.7: System responses for S1.
Figure A.8: System responses for S2.
### Table A.4: Summary of FFT coefficients.

<table>
<thead>
<tr>
<th>FFT Coefficients</th>
<th>Subsystem S1</th>
<th>Subsystem S2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.000000 +j 0.000000</td>
<td>4.70004 +j 0.000000</td>
</tr>
<tr>
<td></td>
<td>-5.69322 +j 18.8673</td>
<td>4.60756 +j 16.2702</td>
</tr>
<tr>
<td></td>
<td>0.000000 +j 0.000000</td>
<td>4.83083 -j 9.53087</td>
</tr>
<tr>
<td></td>
<td>85.5347 +j 72.2399</td>
<td>4.79253 -j 2.79153</td>
</tr>
<tr>
<td></td>
<td>0.000000 +j 0.000000</td>
<td>44.9472 +j 0.000000</td>
</tr>
<tr>
<td></td>
<td>85.5347 -j 72.2399</td>
<td>4.79253 +j 2.79153</td>
</tr>
<tr>
<td></td>
<td>0.000000 +j 0.000000</td>
<td>4.83083 +j 9.53087</td>
</tr>
<tr>
<td></td>
<td>-5.69321 -j 18.8673</td>
<td>4.60756 -j 16.2702</td>
</tr>
</tbody>
</table>
Appendix B

A Method of Reducing the Bias of Noisy Estimates

It was shown in equation (5.162) that a bias term arises when estimating modal parameters in the presence of noise. By varying the variables VAR and R(0) in the PC program it is possible to obtain enough random number samples to simulate this bias effect.

It was observed that $M - \hat{M}$ appeared to be a linear function of $\sqrt{\text{VAR}}$ scaled. By plotting the bias against $\sqrt{\text{VAR}}$ the slope of the resulting line may be obtained and then used to obtain an unbiased estimate of the modal parameters.

The example in appendix A was run with variances of 0.0, 0.0002, 0.0004, and 0.0008, and with $R(0)$ values of 0.25, 0.5, and 0.75 for each value of VAR. The modal parameter estimates for S1 are shown in table B.5, B.6, and B.7.

In figure B.9 the mean of $M - \hat{M}$ is plotted against $\sqrt{\text{VAR}}$ and it can be seen that the bias effect appears to be linear in $\sqrt{\text{VAR}}$ and offer a promising technique to obtain unbiased parameter estimates.

<table>
<thead>
<tr>
<th>Modal parameter estimates, VAR=0.0002</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_1 - \hat{M}_1$</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>-0.062660</td>
</tr>
<tr>
<td>-0.066270</td>
</tr>
<tr>
<td>-0.068750</td>
</tr>
</tbody>
</table>

Table B.5: Modal parameter estimates in the presence of additive measurement noise; VAR=0.0002.
Figure B.9: Graph indicating the linear relationship between modal parameter estimate biases and the noise standard deviation.
### Modal parameter estimates, VAR=0.0004

<table>
<thead>
<tr>
<th>$M_1 - \bar{M}_1$</th>
<th>$M_2 - \bar{M}_2$</th>
<th>$M_3 - \bar{M}_3$</th>
<th>$R(0)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.088690</td>
<td>0.160400</td>
<td>-0.071286</td>
<td>0.25</td>
</tr>
<tr>
<td>-0.091520</td>
<td>0.172300</td>
<td>-0.078662</td>
<td>0.50</td>
</tr>
<tr>
<td>-0.095560</td>
<td>0.175720</td>
<td>-0.075221</td>
<td>0.75</td>
</tr>
</tbody>
</table>

Table B.6: Modal parameter estimates in the presence of additive measurement noise; VAR=0.0004.

### Modal parameter estimates, VAR=0.0008

<table>
<thead>
<tr>
<th>$M_1 - \bar{M}_1$</th>
<th>$M_2 - \bar{M}_2$</th>
<th>$M_3 - \bar{M}_3$</th>
<th>$R(0)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.12559</td>
<td>0.227460</td>
<td>-0.101229</td>
<td>0.25</td>
</tr>
<tr>
<td>-0.12528</td>
<td>0.235980</td>
<td>-0.107764</td>
<td>0.50</td>
</tr>
<tr>
<td>-0.13193</td>
<td>0.242560</td>
<td>-0.103764</td>
<td>0.75</td>
</tr>
</tbody>
</table>

Table B.7: Modal parameter estimates in the presence of additive measurement noise; VAR=0.0008.
Appendix C

Listing of Program Modal

Below is a listing of the program Modal and the subroutine Vandm, both written by the author.

All the program code used was written in FORTRAN and was compiled using Microsoft FORTRAN77 V3.31, August 1985. During the compilation of the program, the following object (*.OBJ) files and libraries (*.LIB) were linked: The object files MODAL, DSLIMP, FINV, VANDM, and FFT, and the libraries FORTRAN, 8087, and MATH.

C.1 Program Modal Listing.

PROGRAM MODAL
C
C This program will calculate the modal parameters of a 2 subsystem system. The first subsystem consists of N1 modes and 3 states while the second subsystem has N2 modes and 2 states.
C
C The output of this program is stored in the file MODAL.LIS.
C
C Variable Legend:
Appendix C. Listing of Program Modal

C ---------------
C S1, S2- the complex exponents.
C C1, C2- the complex coefficients; these are updated
C for every interval.
C CO1, CO2- the original complex coefficients.
C INNER1, INNER2- the values of the inner products of
C the exponential terms and the intervals.
C W- the complex random number ~N(0,1).
C EPS- the tolerance of the subroutine DSLIMP, used to
C for the modal parameters.
C INNY1, INNY2- the values of the inner products of
C y(t) and the interval, w.
C INNYP1, INNYP2- the values of the inner products of
C y'(t) and w.
C INYPP1- the values of the inner products of y''(t) and
C w.
C A1, A2- the matrix of the values of the delayed output
C and w.
C M1, M2, M3, M4, M5- the modal parameters.
C T- the sampling period.
C CO- the values of Co for each interval; simulates the
C change in input.
C A- dummy variable for reading in a real value and making
C it complex.
C MO- the modal constants.
C G1, G2- the values of the unit step response.
Appendix C. Listing of Program Modal

C GP1, GP2- the values of the periodic unit step response.
C GG1, GG2- temporary storage for the value of the modal
C function of the unit step response, G(x,n).
C ADD1, ADD2- dummy variables.
C GAM1, GAM4- the gamma parameters.
C SIGMA- the half period of the periodic signals.
C RS1, RS2- the values of the periodic unit square wave
C response.
C R, RR- used to initialize the random number generator.
C VAR- the variance of the noise.
C I, J, K- loop counters.
C T2- the durations of the half intervals of w.
C N1, N2- the order of the system and the subsystem.
C T1- duration of each interval, w.
C N3- the number of values of the unit step response
C desired.
C N4- the number of distinct values in the periodic responses.
C
COMPLEX S1(10),S2(10),INNER1(10),INNER2(10),TEMP(5),C1(10),C2(10)
COMPLEX CO1(10),CO2(10),W,CC(10),D1(10),D2(10),D3(10),AC(10)
DOUBLE PRECISION EPS
REAL INNY1(10),INNY2(10),A1(10,10),A2(10,10),M1(10),M2(10)
REAL M3(10),M4(10),M5(10),INNYP1(10),INNYP2(10),INYPP1(10)
REAL T,CO(10),A,MO(10),G1(50),G2(50),GP1(10),GP2(10),ADD1
REAL ADD2,GG1,GG2,GAM1(11),GAM4(11),SIGMA,RS1(10),RS2(10)
REAL R,RR,VAR,TNT
Appendix C. Listing of Program Modal

INTEGER I, J, K, T21(10), T22(10), N1, N2, N3, N4, NC, NS, N, T1

OPEN(3, FILE='MODAL.LIS', STATUS='OLD')

EPS = 1.0D-14

WRITE(*,100)
100 FORMAT(IX,'Modal parameter estimation program.')

WRITE(*,101)
101 FORMAT(IX,'First, sample data must be generated.

WRITE(*,102)
102 FORMAT(IX,'Each of the 2 outputs is assumed to be a sum of complex exponentials.

WRITE(*,202)
202 FORMAT(IX,'of complex exponentials.

WRITE(*,103)
103 FORMAT(IX,'Subsystems; SI has 3 states and S2 has 2 states.

WRITE(*,203)
203 FORMAT(IX,'Enter # of modes of subsystem SI.

READ(*,*)N1

WRITE(*,204)
204 FORMAT(IX,'Enter # of modes of subsystem S2.

READ(*,*)N2

WRITE(*,290)
290 FORMAT(IX,'Enter the number of uncoupled modes of SI.

READ(*,*)NS

WRITE(*,104)
104 FORMAT(IX,'All complex entries must be of the form (x,y).

WRITE(*,105)
105 FORMAT(IX,'Subsystem S1, uncoupled modes only!')
Appendix C. Listing of Program Modal

C
C Enter the data.
C
DO 1 I=1,NS
  WRITE(*,106)I
106  FORMAT(1X,'Enter real coefficient',I3)
  READ(*,*)A
  CO1(I)=CMPLX(A,0.)
DO 2 I=1,NS
  WRITE(*,107)I
107  FORMAT(1X,'Enter complex exponential.',I3)
  READ(*,*)S1(I)
WRITE(*,108)
108  FORMAT(1X,'Subsystem S2: Enter coupled modes before uncoupled!')
DO 3 I=1,N2
  WRITE(*,106)I
  READ(*,*)A
  CO2(I)=CMPLX(A,0.)
DO 903 I=1,N2
  WRITE(*,107)I
903  READ(*,*)S2(I)
WRITE(*,330)
330  FORMAT(1X,'Enter the roots of the char. equation, r(i).')
NC=N1-NS
DO 331 I=1,NC+1
  WRITE(*,332)I
Appendix C. Listing of Program Modal

332 FORMAT(1X,'Enter complex root, R(',I2,').')
331 READ(*,*)AC(I)
D0 800 J=1,NC
800 S1(NS+J)=S2(J)
D0 805 J=1,NC
   D2(J)=(1.,0.)
D0 801 K=1,NS
801 D2(J)=(S2(J)-S1(K))*D2(J)
805 CONTINUE
D0 806 J=1,NC
   D1(J)=AC(NC+1)
D0 802 K=1,NC
802 D1(J)=(S2(J)-AC(K))*D1(J)
806 CONTINUE
D0 803 J=1,NC
803 D3(J)=D1(J)/D2(J)
D0 804 I=1,NC
804 CO1(NS+I)=-D3(I)*CO2(I)
WRITE(*,109)
109 FORMAT(1X,'To identify the modal parameters of S1, S2 you must')
WRITE(*,110)N1
110 FORMAT(1X,'specify ',12,' intervals. Specify the length of')
WRITE(*,210)
210 FORMAT(1X,'each half-cycle interval as a multiple of T.')
D0 5 I=1,N1
   WRITE(*,111)I
Appendix C. Listing of Program Modal

111 FORMAT(1X,'Enter CO value for interval ',I2)
5 READ(*,*)CO(I)
WRITE(*,58)
58 FORMAT(1X,'Subsystem 1')
DO 55 I=1,N1
211 FORMAT(I3)
WRITE(*,112)I
112 FORMAT(1X,'Enter the half-cycle duration of interval ',I2)
55 READ(*,211)T21(I)
WRITE(*,57)
57 FORMAT(1X,'Subsystem 2')
DO 56 I=1,N2
   WRITE(*,112)I
56 READ(*,211)T22(I)
WRITE(*,113)
113 FORMAT(1X,'Enter the sampling period, T, in seconds.')
READ(*,*)T
C
C Write the system parameters to the output file, MODAL.LIS.
C
WRITE(3,300)
300 FORMAT(1X,'Modal Parameter Estimation Program.',/)
WRITE(3,301)
301 FORMAT(1X,'Data generation parameters:',/)
WRITE(3,302)N1
302 FORMAT(1X,'Subsystem 1 has 3 states and ',I2,' modes.')
WRITE(3,303)N2
303 FORMAT(1X,'Subsystem 2 has 2 states and ',I2,' modes.',/)
WRITE(3,304)(I,C01(I),I=1,N1)
304 FORMAT(2(2X,'C1(',I2,')=',.2E14.6))
WRITE(3,305)(I,C02(I),I=1,N2)
305 FORMAT(2(2X,'C2(',I2,')=',.2E14.6))
WRITE(3,306)(I,S1(I),I=1,N1)
306 FORMAT(2(2X,'S1(',I2,')=',.2E14.6))
WRITE(3,307)(I,S2(I),I=1,N2)
307 FORMAT(2(2X,'S2(',I2,')=',.2E14.6))
WRITE(3,700)(I,AC(I),I=1,NC+1)
700 FORMAT(2(2X,'R(',I2,')=',.2E14.6))
WRITE(3,308)(I,C0(I),I=1,N1)
308 FORMAT(5X,'CO  for  interval ',I2,'=',.E14.6)
WRITE(3,58)
WRITE(3,309)(I,T21(I),I=1,N1)
309 FORMAT(5X,'Interval #',I2,' has a half cycle of ',I3,*'T s.')
WRITE(3,57)
WRITE(3,309)(I,T22(I),I=1,N2)
WRITE(3,310)T
310 FORMAT(/,1X,'The sampling period is ',E10.3,' seconds.')
WRITE(*,312)
C
C    Random number generator initialization.
C
312 FORMAT(1X,'Enter the noise variance.')
READ(*,*)VAR
WRITE(3,316) VAR
316 FORMAT(1X,'The additive noise \( N(0,'',F7.4,'') \)')
WRITE(*,313)
313 FORMAT(1X,'Random number generator initialization.')
WRITE(*,315)
315 FORMAT(1X,'Enter R(0); (between 0 and 1)')
READ(*,314)R
314 FORMAT(F12.9)
RR=9821.*R+0.211327
R=RR-AINT(RR)
C
C Save the original coefficients since the values will get
C updated every interval.
C
DO 29 I=1,N1
   C1(I)=C01(I)
   IF(I.LE.N2) C2(I)=C02(I)
29 CONTINUE
C
C Calculate \( \langle w, \exp(st) \rangle \).
C
DO 6 I=1,N1
   DO 7 J=1,N1
      INNER1(J)=EX(T,S1,T21,J,I,N1)
      IF(J.GT.N2.OR.I.GT.N2)GO TO 7
INNER2(J) = EX(T, S2, T22, J, I, N2)

CONTINUE

C

C Calculate \( <w, y(t)>, \langle w, y'(t) \rangle \), and \( <w, y''(t) \rangle \).

C

DO 18 J = 1, 5

18 TEMP(J) = (0., 0.)

DO 8 K = 1, N1

TEMP(1) = TEMP(1) - C1(K) * INNER1(K)
TEMP(2) = TEMP(2) - C1(K) * INNER1(K) * S1(K)
TEMP(3) = TEMP(3) - C1(K) * INNER1(K) * S1(K)**2
IF(K.GT.N2.0R.I.GT.N2) GO TO 8
TEMP(4) = TEMP(4) - C2(K) * INNER2(K)
TEMP(5) = TEMP(5) - C2(K) * INNER2(K) * S2(K)

CONTINUE

CALL RANDOM(R, W)
INNY1(I) = REAL(TEMP(1) + W*SQRT(VAR))
CALL RANDOM(R, W)
INNYP1(I) = REAL(TEMP(2) + W*SQRT(VAR))
CALL RANDOM(R, W)
INYP1(I) = REAL(TEMP(3) + W*SQRT(VAR))
CALL RANDOM(R, W)
IF(I.LE.N2) INNY2(I) = REAL(TEMP(4) + W*SQRT(VAR))
CALL RANDOM(R, W)
IF(I.LE.N2) INNYP2(I) = REAL(TEMP(5) + W*SQRT(VAR))
C Calculate the \(<w, y(t-iT)\> \) terms and store them in \(A\)

C

DO 9 K=1,N1
    TEMP(1)=(0.,0.)
    TEMP(4)=(0.,0.)
    DO 10 J=1,N1
      TEMP(1)=TEMP(1)+C1(J)*CEXP(-FLOAT(K)*S1(J)*T)*INNER1(J)
      IF(I.GT.N2.OR.K.GT.N2.OR.J.GT.N2)GO TO 10
      TEMP(4)=TEMP(4)+C2(J)*CEXP(-FLOAT(K)*S2(J)*T)*INNER2(J)
10    CONTINUE
    CALL RANDOM(R,W)
    A1(I,K)=REAL(TEMP(1)+W*SQRT(VAR))
    CALL RANDOM(R,W)
    IF(K.LE.N2.AND.I.LE.N2) A2(I,K)=REAL(TEMP(4)+W*SQRT(VAR))
9    CONTINUE

C

C Update the coefficients for the next interval.

C

NT=N1+2*T21(I)
N=N2+2*T22(I)
IF(N.GT.NT) NT=N
IF(I.EQ.1) T1=NT
TNT=NT*T
    CALL UPDATE(S2,C2,CC,CO,TNT,N2,0,I)
    DO 420 J=1,NC
420    CC(J)=-D3(J)*C2(J)
CALL UPDATE(S1,C1,CC,CO,TNT,NS,NC,I)

DO 421 J=1,NC

421 C1(NS+J)=CC(J)

K=I+1

IF(K.LE.N1) THEN

WRITE(3,311) K

311 FORMAT(IX,'New C parameters for interval ',I3)

WRITE(3,304)(J,C1(J),J=1,N1)

WRITE(3,305)(J,C2(J),J=1,N2)

END IF

6 CONTINUE

C

C Solve the matrix equations for the modal parameters and
C check to see that the modal identity adds up to zero.
C

CALL DSLIMP(A1,INNY1,M1,N1,EPS)
CALL CHECK(N1,A1,M1,1,INNY1,EPS)
CALL CONST(M0(1),M1,N1,T,T1,1,CO,CO1,S1,0)
CALL DSLIMP(A2,INNY2,M4,N2,EPS)
CALL CHECK(N2,A2,M4,4,INNY2,EPS)
CALL CONST(M0(4),M4,N2,T,T1,4,CO,CO2,S2,0)

C

C Update A matrix for further calculations.
C

DO 11 I=1,N1

11 DO 12 J=N1,2,-1
A1(I,J) = A1(I, J-1)

12  IF(I.LE.N2.AND.J.LE.N2) A2(I,J)=A2(I,J-1)
    IF(I.LE.N2) A2(I,1) = -INNY2(I)

11  A1(I,1) = -INNY1(I)

C

C Solve for the modal parameters of the second and third
C states and check the modal identities.

C

CALL DSLIMP(A1,INNYP1,M2,N1,EPS)
CALL CHECK(N1,A1,M2,2,INNYP1,EPS)
CALL CONST(MO(2),M2,N1,T,T1,2,C0,C01,S1,1)
CALL DSLIMP(A1,INNYP1,M3,N1,EPS)
CALL CHECK(N1,A1,M3,3,INNYP1,EPS)
CALL CONST(MO(3),M3,N1,T,T1,3,C0,C01,S1,2)
CALL DSLIMP(A2,INNYP2,M5,N2,EPS)
CALL CHECK(N2,A2,M5,5,INNYP2,EPS)
CALL CONST(MO(5),M5,N2,T,T1,5,C0,C02,S2,1)

C

C Unit step response.

C

WRITE(*,400)
400 FORMAT(IX,'Unit step response generation.')
WRITE(*,401)
401 FORMAT(IX,'Enter the number of data points desired.')
READ(*,*) N3
DO 19 I = 1, N3
Appendix C. Listing of Program Modal

GG1=-M0(1)
GG2=-M0(4)
DEL=(I-1)*T
ADD1=Y(C0(1),CO1,S1,N1,DEL)
ADD2=Y(C0(1),CO2,S2,N2,DEL)
DO 21 K=1,N1
  DEL=(I-1-K)*T
  ADD1=ADD1+(M1(K)*Y(CO(1),CO1,S1,N1,DEL))
  IF(K.LE.N2)ADD2=ADD2+(M4(K)*Y(CO(1),CO2,S2,N2,DEL))
21  CONTINUE
GG1=GG1+ADD1
GG2=GG2+ADD2
G1(I)=GG1
G2(I)=GG2
J=MINO(N1+1,I)
DO 22 K=1,J-1
  G1(I)=G1(I)-(M1(K)*G1(I-K))
  IF(K.LE.N2)G2(I)=G2(I)-(M4(K)*G2(I-K))
22  CONTINUE
19  CONTINUE
C
C     Print out
C
WRITE(*,402)N3
WRITE(3,402)N3
402 FORMAT(1X,'The first ',13,' values of the step response are:')
Appendix C. Listing of Program Modal

WRITE(*,404)
WRITE(3,404)
404 FORMAT(1X,'Subsystem 1')
WRITE(*,403)(I,G1(I),I=1,N3)
WRITE(3,403)(I,G1(I),I=1,N3)
403 FORMAT(2(2X,'G(',12,')='E14.6))
WRITE(*,405)
WRITE(3,405)
405 FORMAT(1X,'Subsystem 2')
WRITE(*,403)(I,G2(I),I=1,N3)
WRITE(3,403)(I,G2(I),I=1,N3)
PAUSE

C

C Periodic unit step response.
C

DO 23 I=1,N1+1
  GP1(I)=-0.5*M0(1)+G1(I)
  GP2(I)=-0.5*M0(4)+G2(I)
DO 24 J=1,I-1
  GP1(I)=GP1(I)+(M1(J)*G1(I-J))
  IF(J.LE.N2)GP2(I)=GP2(I)+(M4(J)*G2(I-J))
24  CONTINUE
23  CONTINUE

C

C Print out periodic step response.
C

Appendix C. Listing of Program Modal

\[ N4 = N1 + 1 \]

WRITE(*,406)N4
WRITE(3,406)N4

406 FORMAT(1X,'The ',I3,' values of the periodic step response are')
WRITE(*,404)
WRITE(3,404)

WRITE(*,407)(I,GP1(I),I=1,N1+1)
WRITE(3,407)(I,GP1(I),I=1,N1+1)

407 FORMAT(2(2X,'GP(',12.')='E14.6))
WRITE(*,405)
WRITE(3,405)

WRITE(*,407)(I,GP2(I),I=1,N2+1)
WRITE(3,407)(I,GP2(I),I=1,N2+1)

PAUSE

C

C Square wave response.

C

WRITE(*,500)
WRITE(3,500)

500 FORMAT(1X,'The Periodic Square Wave Response is given by:')
WRITE(*,404)
WRITE(3,404)

C

C Figure out the gamma parameters.

C

SIGMA=0.5
CALL INCOEF(M1,N1,GAM1,SIGMA)
WRITE(*,405)
WRITE(3,405)
CALL INCOEF(M4,N2,GAM4,SIGMA)

C
C Calculate the periodic square wave response.
C
DO 25 I=1,N1+1
   RS1(I)=GAM1(1)*GP1(I)
   DO 26 J=1,N1
      IF(I.GT.J) RS1(I)=RS1(I)+GAM1(J+1)*GP1(I-J)
   26 IF(I.LE.J) RS1(I)=RS1(I)-GAM1(J+1)*GP1(N1+1+I-J)
   IF(I.GT.N2+1) GOTO 25
   RS2(I)=GAM4(1)*GP2(I)
   DO 27 J=1,N2
      IF(I.GT.J) RS2(I)=RS2(I)+GAM4(J+1)*GP2(I-J)
   27 IF(I.LE.J) RS2(I)=RS2(I)-GAM4(J+1)*GP2(N2+1+I-J)
25 CONTINUE

C
C Print out the periodic square wave response.
C
WRITE(*,408)N4
WRITE(3,408)N4
408 FORMAT(IX,'The ',I3,,' values of the per. square wave resp. are')
WRITE(*,404)
WRITE(3,404)
Appendix C. Listing of Program Modal

WRITE(*,409)(I,RS1(I),I=1,N1+1)
WRITE(3,409)(I,RS1(I),I=1,N1+1)
409 FORMAT(2(2X,'RS(',I2,')=','E14.6))
WRITE(*,405)
WRITE(3,405)
WRITE(*,409)(I,RS2(I),I=1,N2+1)
WRITE(3,409)(I,RS2(I),I=1,N2+1)
PAUSE
C
C Calculate the Fourier coefficients
C
WRITE(3,410)
WRITE(*,410)
410 FORMAT(IX,'FFT coefficients of Gp(t).')
WRITE(3,404)
WRITE(*,404)
WRITE(*,404)
CALL SETFFT(GP1,N1+1,T)
WRITE(3,405)
WRITE(*,405)
WRITE(*,405)
CALL SETFFT(GP2,N2+1,T)
STOP
END
C
C
SUBROUTINE INCOEF(M,N,GAM,SIGMA)
C
This subroutine sets up a matrix of scaled modal parameters and inverts it to find the gamma parameters that relate the unit periodic square wave response to the periodic unit step response.

Variables.

\begin{itemize}
  \item \textbf{M-} the modal parameters.
  \item \textbf{GAM-} the gamma parameters.
  \item \textbf{MAT-} the scaled modal matrix.
  \item \textbf{MAT2-} the inverted matrix.
  \item \textbf{SIGMA-} \( \sigma=(N+1)T \).
  \item \textbf{COND-} checks the success of the matrix inversion.
  \item \textbf{I,J-} counters.
  \item \textbf{N-} the order of the system.
\end{itemize}

\begin{verbatim}
REAL M(10),GAM(11),MAT(11,11),MAT2(11,11)
REAL SIGMA,COND
INTEGER N,I,J

C

C Set up the scaled modal matrix.

C
DO 1 I=1,N+1
  DO 2 J=1,N+1
    IF(I.EQ.J) MAT(I,J)=SIGMA
    IF(I.LT.J) MAT(I,J)=-M(N+1+I-J)*SIGMA
\end{verbatim}
Appendix C. Listing of Program Modal

IF(I.GT.J) MAT(I,J)=M(I-J)*SIGMA
2  CONTINUE
1  CONTINUE

C
C  Invert the matrix.
C
CALL FINV(N+11, MAT, 11, MAT2, COND)
IF(COND.EQ.0.0) WRITE(3,100)
100 FORMAT(IX,'Inversion has failed!!!!!!')
C
C  Extract the gamma parameters from the inverted matrix.
C
DO 3 I=1,N+1
3  GAM(I)=MAT2(I,1)
WRITE(*,101)(I,GAM(I),I=1,N+1)
WRITE(3,101)(I,GAM(I),I=1,N+1)
101 FORMAT(2(2X.'GAMMA(',12,')=',E14.6))
RETURN
END

C
C
SUBROUTINE CHECK(N,A,MOD,M,INNY,EPS)

C
C  This subroutine evaluates the modal identity and sees that
C  it adds up to zero. In this way you can be sure that the
C  parameters are correct.
Appendix C. Listing of Program Modal

C
C Variables.
C ---------
C A- the matrix of delayed output inner product terms.
C MOD- the modal parameters.
C INNY- the $\langle w, y(t) \rangle$, $\langle w, y'(t) \rangle$, or $\langle w, y''(t) \rangle$ terms.
C CHECK- the calculated value of the modal identity.
C EPS- the degree of accuracy of the solution of the modal parameters.
C I- a counter
C N- the degree of the system.
C M- an indicator of what state of what subsystem.
C

REAL A(10,10), MOD(10), INNY(IO), CHECK
DOUBLE PRECISION EPS
INTEGER I, M, N
WRITE(3,100)M
WRITE(3,101)(I, MOD(I), I=1,N)
WRITE(3,102)
WRITE(*,100)M
100 FORMAT(IX,'The modal parameters, M',I2,' are:')
WRITE(*,101)(I, MOD(I), I=1,N)
101 FORMAT(2(2X,'(',12,')='E14.6))
WRITE(*,102)
102 FORMAT(IX,'Checking!')
WRITE(*,600)EPS
WRITE(3,600)EPS
600 FORMAT(1X,'EPS adds up to ',E14.6)
C
C Evaluate the modal identity.
C
DO 2 J=1,N
  CHECK=INNY(J)
  DO 1 I=1,N
  1 CHECK=CHECK-(MOD(I)*A(J,I))
  WRITE(3,103)CHECK
  WRITE(*,103)CHECK
103 FORMAT(1X,'The modal identity adds up to ',E14.6,/)
2 CONTINUE
RETURN
END
C
C
FUNCTION EX(T,S,T2,J,I,N)
C
C This function evaluates the inner product, <w,exp(st)>
C when w(t) is a sal(1,x) function of duration 2*T2(I).
C
C Variables.
C  ---------
C S- the array of complex exponentials
C EX- dummy variable
Appendix C. Listing of Program Modal

C T- sampling period
C MID- the midpoint of w(t)
C BEGIN- the beginning of w(t)
C END- the endpoint of w(t)
C T2- array of the half-cycle durations of w(t)
C I,J,N- counters

COMPLEX S(5).EX
REAL T,MID,BEGIN,END
INTEGER T2(5),I,J,N
BEGIN=FLOAT(N)
MID=FLOAT(N+T2(I))
END=FLOAT(N+2*T2(I))
EX=2.*CEXP(MID*S(J)*T)-CEXP(BEGIN*S(J)*T)
EX=(EX-CEXP(END*S(J)*T))/S(J)
RETURN
END

SUBROUTINE UPDATE(S,C,CC,CO,TNT,N,NC,M)

C
C This subroutine will update the C coefficients after
C an input change.
C
C Variables.
C
---------
Appendix C. Listing of Program Modal

C S- complex exponentials
C B- accumulating variable array
C X- dummy variable for storing values of powers of S
C C- coefficients
C CO- CO coefficients
C T- sampling period
C I,J- counters
C N- order of system
C T2- half-cycle duration of intervals
C M- interval number
C T1- time elapsed since the beginning of the first interval

COMPLEX S(10),B(10),X,C(10),CC(10)
REAL CO(10),TNT
INTEGER I,J,N,M,NC
INTEGER CI,N3

C
C Set up the Vandermonde system to be solved for the
C new coefficients.
C
DO 1 I=1,N
   B(I)=(0.,0.)
   IF(I.EQ.1) B(I)=CO(M)
   DO 2 J=1,N+NC
      X=S(J)**(I-1)
      IF(AIMAG(S(J)).EQ.0.) X=CMPLX(REAL(X),0.)
Appendix C. Listing of Program Modal

2 \[ B(I) = B(I) + X * C(J) * \text{CEXP}(S(J) * T) \]

IF(NC.EQ.0) GO TO 1

DO 4 J=1, NC
   X=S(N+J)**(I-1)
   IF(AIMAG(S(N+J)).EQ.0.) X=CMPLX(REAL(X),0.)
   4 B(I)=B(I)-CC(J)*X

1  IF(I.EQ.1) B(I)=B(I)-C0(M+1)

CALL VANDM(S,B,N)

C
C Eliminate any unwanted imaginary parts of the new coefficients.
C
DO 3 I=1,N
   IF(AIMAG(S(I)).EQ.0.) B(I)=CMPLX(REAL(B(I)),0.)
3  C(I)=B(I)

RETURN
END

C
C SUBROUTINE CONST(MO,MOD,N,T,T1,I,CO,C,S,DERIV)
C
C This subroutine calculates the modal offset constant, MO.
C
C Variables.
C
----------
Appendix C. Listing of Program Modal

C    C- coefficients
C    S- complex exponentials
C    MO- the modal constant term, MO
C    MOD- array of modal parameters
C    T- the sampling period
C    CO- CO constants
C    N- the order of the system
C    T1- interval duration
C    I,J- counters
C    DERIV- the state of the system
C    DEL- the delay of each term
C
COMPLEX C(10),S(10)
REAL MO,MOD(10),T,CO(10)
INTEGER N,T1,I,J,DERIV,DEL
MO=WOY(CO,N,T1,T,C,S,0,DERIV)
DO 1 J=1,N
     DEL=J
     IF(I.NE.1.AND.I.NE.4) DEL=J-1
1       MO=MO+MOD(J)*WOY(CO,N,T1,T,C,S,DEL,0)
MO=MO/(FLOAT(T1-N)*T)
WRITE(*,100)I,MO
WRITE(3,100)I,MO
100 FORMAT(ix,'The M',i2,' constant is ',e14.6,'./)
PAUSE
RETURN
FUNCTION WOY(CO,N,T1,T,C,S,DEL,DERIV)

C

This function calculates the \( <w, y(t-itd)> \) terms for
the MO constant calculation.

C

COMPLEX C(10),S(10),WHY,X
REAL CO(10),T,WOY
INTEGER N,T1,DEL,I,DERIV
WHY=CO(1)*FLOAT(T1-N)*T
IF(DERIV.NE.0) WHY=(0.,0.)
DO 1 I=1,N
   X=C(I)*(S(I)**DERIV)*WO(S(I),N,T1,T)*CEXP(-FLOAT(DEL)*S(I)*T)
1   WHY=WHY+X
WOY=REAL(WHY)
RETURN
END
C
C
FUNCTION WO(S,N,T1,T)
C
This function calculates the \( <w, \exp(st)> \) terms for
the MO term calculation.
Appendix C. Listing of Program Modal

COMPLEX S,WO
REAL T
INTEGER N,T1
WO=(CEXP(FLOAT(T1)*S*T)-CEXP(FLOAT(N)*S*T))/S
RETURN
END

C
C
FUNCTION Y(CO,C,S,N,TAU)
C
C   This function evaluates y(t).
C
COMPLEX C(10),S(10),YC
REAL TAU,CO
INTEGER N,I
YC=CMPLX(CO,0.)
DO 1 I=1,N
  1   YC=YC+(C(I)*CEXP(S(I)*TAU))
Y=REAL(YC)
IF(TAU.LT.0.) RETURN
Y=2.*Y
RETURN
END
C
C
SUBROUTINE RANDOM(R,W)
C
C This subroutine generates a complex random number
C where its real and imaginary parts both ~N(0,1).
C
REAL R,R1,Z,Y
COMPLEX W
R1=9821.*R+0.211327
R1=R1-AINT(R1)
Z=SQRT(-2.* ALOG(R))*C0S(2.*3.14*R1)
Y=SQRT(-2.*ALOG(R))*SIN(2.*3.14*R1)
W=CMPLX(Z,Y)
R=R1
RETURN
END

C
C

SUBROUTINE SETFFT(DATA,N,T)
C
C This subroutine prepares data values for use by
C a FFT routine. This subroutine was adapted from
C a book by Marple.
C
COMPLEX X(64),W(64)
REAL DATA(10),T
INTEGER NF,NEXP,I,J
DO 4 I=1,5
Appendix C. Listing of Program Modal

4 IF((2*N).LE.(2**I)) GO TO 5
5 NEXP=I
DO 1 I=0,2,2
   DO 2 J=1,N
   2 X((I*N)+J)=CMPLX(DATA(J),0.)
   DO 3 J=1,N
   3 X(((I+1)*N)+J)=CMPLX(-DATA(J),0.)
1 CONTINUE
NF=2**NEXP
WRITE(*,102)(I,X(I),I=1,NF)
102 FORMAT(2(2X,'X(',12,')=',2E14.6))
CALL PREFFT(NF,O,NEXP,W)
CALL FFT(NF,O,T,NEXP,W,X)
WRITE(3,100)
WRITE(*,100)
100 FORMAT(IX,'Fourier Coefficients: Modal function. ')
WRITE(3,101)(I,X(I),I=1,NF)
WRITE(*,101)(I,X(I),I=1,NF)
101 FORMAT(2(2X,'(',12,')=',2E14.6))
RETURN
END

C.2 Subroutine Vandm Listing.

SUBROUTINE VANDM(X,B,N)
C
Appendix C. Listing of Program Modal

C Solves the system of linear equations Vz=b
C where V is a Van der Monde matrix.
C
COMPLEX X(10),B(10)
INTEGER I,K,N
DO 1 K=1,N-1
   DO 2 I=N,K+1,-1
2   B(I)=B(I)-X(K)*B(I-1)
1 CONTINUE
DO 3 K=N-1,1,-1
   DO 4 I=K+1,N
4   B(I)=B(I)/(X(I)-X(I-K))
   DO 5 I=K,N-1
5   B(I)=B(I)-B(I+1)
3 CONTINUE
RETURN
END
Appendix D

Listings of Other Subroutines

D.1 Listing of the Subroutines DSLIMP and FINV

The following two subroutines were taken from the University of British Columbia, MTS-G mainframe system and are described [14]. Some changes were made to both subroutines.

D.1.1 Listing of Subroutine DSLIMP

```
SUBROUTINE DSLIMP(AA,BB,XX,N,EPS)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION A(10,10),T(10,10),B(10),X(10),RZ(10),IPS(10)
REAL AA(10,10),BB(10),XX(10)
DOUBLE PRECISION EPS
DO 13 I=1,N
   DO 11 J=1,N
11   A(I,J)=DBLE(AA(I,J))
13   B(I)=DBLE(BB(I))
L=10
LT=1
ITMAX=10
IF(LT.NE.1) GO TO 10
```
CALL LRD(N,N,L,A,IPS,L,T)
CALL DETM(N,IPS,L,T,DET,JEXP)
10 CALL DBS(N,1,L,B,X,IPS,L,T)
XNORM=O.DO
DO 1 I=1,N
1  XNORM=DMAX1(XNORM,DABS(X(I)))
IF(XNORM.LE.O.DO) RETURN
EPS=EPS*XNORM
ZX=1.D+60
LD=0
DO 2 LL=1,ITMAX
   DO 3 I=1,N
      DSUMM=0.0
      DO 4 K=1,N
         DA=A(I,K)
         DX=X(K)
      4    DSUMM=DSUMM+DA*DX
      DSUMM=B(I)-DSUMM
   3    RZ(I)=DSUMM
   CALL DBS(N,1,L,RZ,RZ,IPS,L,T)
   ZNORM=O.DO
   DO 5 I=1,N
   5    ZNORM=DMAX1(ZNORM,DABS(RZ(I)))
      X(I)=X(I)+RZ(I)
   DO 12 I=1,N
   12   XX(I)=SNGL(X(I))
Appendix D. Listings of Other Subroutines

IF(ZNORM.GT.ZX) GO TO 50
IF((ZNORM-EPS).LT.O.DO) GO TO 60
ZX=ZNORM
GO TO 2
50 IF(ZNORM.GT.10.D0*ZX) GO TO 70
LD=LD+1
IF(LD.GE.3) GO TO 70
2 CONTINUE
LL=LL-1
WRITE(*,250)
WRITE(3,250)
GO TO 71
70 WRITE(*,247)
WRITE(3,247)
71 EPS=-ZNORM
NITER=LL
RETURN
60 EPS=ZNORM
NITER=LL
RETURN
250 FORMAT(/'0***ITERATIVE IMPROVEMENT DID NOT CONVERGE'/)
247 FORMAT(/'0***ITERATIVE IMPROVEMENT IS DIVERGING'/)
END
SUBROUTINE LRD(N,N1,NDIMA,A,IP,NDIMT,T)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION A(IO,IO),T(10,10),IP(10)
DO 7 J=1,N
    DO 7 I=1,N
7    T(I,J)=A(I,J)
IP(N)=1
DO 6 K=1,N
    IF(K.EQ.N) GO TO 5
    KP1=K+1
    M=K
    DO 1 I=KP1,N
        IF(DABS(T(I,K)).GT.DABS(T(M,K))) M=I
1    CONTINUE
    IP(K)=M
    IF(M.NE.K) IP(N)=-IP(N)
    TEMP=T(M,K)
    T(M,K)=T(K,K)
    T(K,K)=TEMP
    IF(TEMP.EQ.O.DO) GO TO 5
    DO 2 I=KP1,N
2    T(I,K)=-T(I,K)/TEMP
    DO 4 J=KP1,N
4    TEMP=T(M,J)
        T(M,J)=T(K,J)
        T(K,J)=TEMP
        IF(TEMP.EQ.O.DO) GO TO 4
    DO 3 I=KP1,N
3    T(I,J)=T(I,J)+T(I,K)*TEMP
4 CONTINUE
5 IF(T(K,K).EQ.0.DO) IP(N)=0
6 CONTINUE
RETURN
END

SUBROUTINE DETM(N,IP,NDIMT,T,DET,JEXP)
IMPLICIT REAL*8 (A-H,O-Z)
DIMENSION T(IO.IO),IP(10)
DET=IP(N)
JEXP=0
DO 1 I=1,N
   TEMP=T(I,I)
   IF(DABS(DET).LE.1.D15) GO TO 2
   DET=DET*1.D-15
   JEXP=JEXP+15
   GO TO 3
2 IF(DABS(DET).GE.1D-15) GO TO 3
   DET=DET*1.D+15
   JEXP=JEXP-15
3 DET=DET*TEMP
1 CONTINUE
RETURN
END

SUBROUTINE DBS(N,NSOL,NDIMBX,B,X,IP,NDIMT,T)
IMPLICIT REAL*8(A-H,O-Z)
DIMENSION T(IO.IO),X(IO),B(IO)
INTEGER IP(IO), K, N, NM1, KP1, KB

NM1 = N - 1

DO 2 I = 1, N

2 X(I) = B(I)

IF(N .EQ. 1) GO TO 9

DO 10 K = 1, NM1

KP1 = K + 1

M = IP(K)

TEMP = X(M)

X(M) = X(K)

X(K) = TEMP

DO 7 I = KP1, N

7 X(I) = X(I) + T(I, K) * TEMP

10 CONTINUE

DO 8 KB = 1, NM1

KM1 = N - KB

K = KM1 + 1

X(K) = X(K) / T(K, K)

TEMP = -X(K)

DO 8 I = 1, KM1

8 X(I) = X(I) + T(I, K) * TEMP

9 X(1) = X(1) / T(1, 1)

RETURN

END
Appendix D. Listings of Other Subroutines

D.1.2 Listing of Subroutine FINV

SUBROUTINE FINV(N,NDIMT,T1,NDIMA,A,COND)
C
C Calculates the inverse of a matrix.
C
DIMENSION A(11,11),IP(11),T1(11,11)
REAL*8 DSQRT,CSUMA,CSUMB,TA
DO 30 J=1,N
  DO 30 I=1,N
 30 A(I,J)=T1(I,J)
IEXP=0
IF(N.EQ.1) GO TO 1991
DEt=1.0
KSW=1
GO TO 260
G0 TO 260
45 CSUMA=CSUMB
DO 199 K=1,N
  AMAX=ABS(A(K,K))
  IMAX=K
  IF(K.EQ.N) GO TO 65
 50 KP=K+1
  DO 60 I=KP,N
    AIK=ABS(A(I,K))
    IF(AIK.LE.AMAX) GO TO 60
 55 AMAX=AIK
IMAX=I

60 CONTINUE

65 IF(A(MAX.EQ.0.) GO TO 300

IP(K)=IMAX

IF(K.EQ.IMAX) GO TO 100

DET=-DET

100 DET=DET*A(IMAX,K)

QDET=ABS(DET)

IF(QDET.LT.1.E15) GO TO 701

DET=DET*1.E-15

IEXP=IEXP+15

701 IF(QDET.GT.1.E-15) GO TO 750

DET=DET*1.E15

IEXP=IEXP-15

750 T=1./A(IMAX,K)

A(IMAX,K)=A(K,K)

A(K,K)=-1.0

DO 1999 I=1,N

A(I,K)=-A(I.K)*T

1999 CONTINUE

DO 144 J=1,N

IF(J.EQ.K) GO TO 144

TEMP=A(IMAX,J)

IF(K.EQ.IMAX) GO TO 140

A(IMAX,J)=A(K,J)

75 A(K,J)=TEMP
Appendix D. Listings of Other Subroutines

140 A(K,J)=TEMP*T

DO 109 I=1,N
   IF(I.EQ.K) GO TO 109
   A(I,J)=A(I,J)+TEMP*A(I,K)
109 CONTINUE

144 CONTINUE

199 CONTINUE

NM1=N-1

DO 250 KK=1,NM1
   K=N-KK
210 J=IP(K)
   IF(J.EQ.K) GO TO 250
220 DO 225 I=1,N
      T=A(I,J)
      A(I,J)=A(I,K)
      A(I,K)=T
225 CONTINUE

250 CONTINUE

KSW=2

260 CSUMB=0.0D0

DO 271 J=1,N
   DO 270 I=1,N
      TA=A(I,J)
      CSUMB=CSUMB+TA*TA
270 CONTINUE
271 CONTINUE

GO TO (45,275), KSW
Appendix D. Listings of Other Subroutines

275 FN=N
COND=DSQRT(CSUMA*CSUMB)/FN
RETURN
300 WRITE(*,310) K,AMAX
310 FORMAT(IX,'STEP',I3,'PIVOT =',E18.8,'INVERSION STOPPED',/)
DET=0.0
IEXP=0
COND=0.0
RETURN
1991 IF(A(1,1).EQ.0.) GO TO 1992
DET=A(1,1)
A(1,1)=1./A(1,1)
COND=1.
RETURN
1992 K=1
AMAX=0.
GO TO 300
END

D.2 Listing of Subroutine FFT

The subroutine FFT was obtained from [12]. The subroutine as cited here combines the two subroutines PREFFT and FFT.

C These two programs set up the complex exponential table (PREFFT) and compute the discrete-time Fourier series of an array of complex data samples using a decimation-in-frequency fast Fourier transform (FFT)
C algorithm.

SUBROUTINE PREFFT (N, MODE, NEXP, W)

C Input Parameters:
C
C N - Number of data samples to be processed (integer-must be a
C     power of two)
C MODE - Set to 0 for discrete-time Fourier series (Eq. 2.C.1) or
C     1 for inverse (Eq. 2.C.2)
C
C Output Parameters:
C
C NEXP - Indicates power-of-2 exponent such that $N=2^{**NEXP}$.
C     Will be set to -1 to indicate error condition if $N$
C     is not a power of 2 (this integer used by sub. FFT)
C W - Complex exponential array
C
C Notes:
C
C External array W must be dimensioned .GE. N by calling program.
C
COMPLEX W(1), C1, C2
NEXP=1
5 NT=2**NEXP
IF (NT .GE. N) GO TO 10
NEXP=NEXP+1
GO TO 5
10 IF (NT .EQ. N) GO TO 15
NEXP=-1
RETURN
15 S=8.*ATAN(1.)/FLOAT(NT)
C1=CMPLX(COS(S),-SIN(S))
IF (MODE .NE. 0) C1=CONJG(C1)
C2=(1.,0.)
DO 20 K=1,NT
  W(K)=C2
20 C2=C2*C1
RETURN
END

SUBROUTINE FFT (N,MODE,T,NEXP,W,X)
C
C Input Parameters:
C
C N,MODE,NEXP,W - See parameter list for subroutine PREFFT
C T            - Sample interval in seconds
C X            - Array of N complex data samples, X(1) to X(N)
C
C Output Parameters:
C
Appendix D. Listings of Other Subroutines

C X - N complex transform values replace original data samples
C indexed from k=1 to k=N, representing the frequencies
C (k-1)/NT hertz
C
C Notes:
C
C External array X must be dimensioned .GE. N by calling program.
C
COMPLEX X(1),W(1),C1,C2
MM=1
LL=N
DO 70 K=1,NEXP
   NN=LL/2
   JJ=MM+1
   DO 40 I=1,N,LL
      KK=I+NN
      C1=X(I)+X(KK)
      X(KK)=X(I)-X(KK)
   40 X(I)=C1
   IF (NN .EQ. 1) GO TO 70
   DO 60 J=2,NN
      C2=W(JJ)
      DO 50 I=J,N,LL
         KK=I+NN
         C1=X(I)+X(KK)
         X(KK)=(X(I)-X(KK))*C2
   50    
   60    
    70    
Appendix D. Listings of Other Subroutines

50     \( X(I) = C_1 \)
60     \( JJ = JJ + MM \)

    \( LL = NN \)
    \( MM = MM \times 2 \)

70     CONTINUE

\( NV2 = N/2 \)
\( NM1 = N - 1 \)
\( J = 1 \)

DO 90 I = 1, NM1

    IF (I .GE. J) GO TO 80

    \( C_1 = X(J) \)
    \( X(J) = X(I) \)
    \( X(I) = C_1 \)

80     \( K = NV2 \)

85     IF (K .GE. J) GO TO 90

    \( J = J - K \)
    \( K = K/2 \)

    GO TO 85

90     \( J = J + K \)

    IF (MODE .EQ. 0) \( S = T \)
    IF (MODE .NE. 0) \( S = 1/(T \times FLOAT(N)) \)

DO 100 I = 1, N

100    \( X(I) = X(I) \times S \)

RETURN

END