ROBUST NONLINEAR SYSTEM IDENTIFICATION USING CORRELATION TECHNIQUES

A thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Engineering

by

ANIL GOPINATHAN
B.Tech., University of Kerala, 1991

1999
Wright State University
WRIGHT STATE UNIVERSITY

SCHOOL OF GRADUATE STUDIES

December 17, 1999

I HEREBY RECOMMEND THAT THE THESIS PREPARED UNDER MY SUPERVISION BY Anil Gopinathan ENTITLED Robust Nonlinear System Identification Using Correlation Techniques BE ACCEPTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF Master of Science in Engineering

__________________________________________
Joseph Slater, Ph.D.
Thesis Director

__________________________________________
Richard Bethke, Ph.D.
Department Chair

Committee on
Final Examination

__________________________________________
Joseph Slater, Ph.D.

__________________________________________
Mitch Wolff, Ph.D.

__________________________________________
Pradeep Misra, Ph.D.

__________________________________________
Joseph F. Thomas, Jr., Ph.D.
Dean, School of Graduate Studies
ABSTRACT


The fundamental challenge in identification of nonlinear dynamic systems is determining the appropriate form of the model. A robust technique which eliminates this problem is evaluated here. The technique is based on the Minimum Model Error (MME) optimal estimation approach. A study on the applicability of this method for systems with Single Degree of Freedom, Two Degrees of Freedom and Continuous system is done. The most significant feature of this approach is the ability to identify nonlinear dynamic systems without prior assumptions regarding the form of the non-linearities, in contrast to most other methods which usually require detailed assumptions of the non-linearities. Model form is determined via statistical correlation of MME optimal state estimates with the MME optimal model error estimates. The three example illustrations indicate that the method is robust with respect to prior ignorance of the model, and with respect to measurement noise, and measurement record length. The robustness of the identification technique is demonstrated by the ability of it to identify the true system model even in the presence of noise to the extent of 15%.

A modification to the existing technique has been developed whereby the system identification is possible even when only the acceleration data is available. It is demonstrated that this modified technique evaluates the states and the model errors from the acceleration data and determines the model form of the system. The robustness of this is demonstrated by the ability of the method to identify the system model even in the presence of noise to
the extent of 15\%. The derivation of the algorithm is mentioned and its application to three
different systems is demonstrated.

To demonstrate the ability of the system to identify the nonlinear model, the problem of
two dimensional plates undergoing limit cycle oscillations (subsequent to the occurrence
of a linear, aeroelastic instability) is investigated. The states of the system were obtained
from an earlier analytical solution and these states were used to determine the non-linear
model of the system. It is demonstrated that the non-linear model obtained depends on the
linear model assumed in MME. In order to get a model that makes more physical sense
pertaining to the system under investigation, it is always better to start with the true linear
part of the system model. The system model identified using two different linear models is
demonstrated. It is found that both model equations satisfy the system but the model, which
had assumed the true linear part of the system to start with made more physical sense with
respect to the non-linearity identified.

Further the ability of the MME to be able to identify the mathematical model of the
Panel Flutter problem is demonstrated when only the acceleration data is available. It
is found that the model obtained thus was in close agreement with that identified when
both position and velocity states were assumed available. The robustness of the method
is demonstrated when it identifies the model even in the presence of 10\% noise in the
acceleration signal.
List of Symbols

Chapter 3

\n
\( n \) \hspace{1cm} \text{system dimension} \\
\( A \) \hspace{1cm} \text{true system state matrix, } n \times n \\
\( A_i \) \hspace{1cm} \text{assumed model state matrix, } n \times n \\
\( B y(t) \) \hspace{1cm} \text{vector of forcing functions, } n \times 1 \\
\( d(t) \) \hspace{1cm} \text{dynamic model error or correction term} \\
\( f(x(t), \hat{x}, t) \) \hspace{1cm} \text{vector containing nonlinear terms} \\
\( g_k(x(t_k), t_k) \) \hspace{1cm} \text{accurate model of measurement process} \\
\( I \) \hspace{1cm} \text{identity matrix} \\
\( J \) \hspace{1cm} \text{cost functional} \\
\( M \) \hspace{1cm} \text{number of measurements} \\
\( p \) \hspace{1cm} \text{number of states in measurement vector} \\
\( R_k \) \hspace{1cm} \text{measurement covariance matrix, } n \times n \\
\( t_k \) \hspace{1cm} \text{time of } k^{\text{th}} \text{ sample} \\
\( \xi_k \) \hspace{1cm} \text{zero mean gaussian noise at time } t_k \\
\( W \) \hspace{1cm} \text{variable weighting matrix, } n \times n \\
\( \bar{x}(t) \) \hspace{1cm} \text{true model state vector, } n \times 1 \\
\( \hat{x}(t) \) \hspace{1cm} \text{estimated state vector from corrected model} \\
\( \tilde{y}(t_k) \) \hspace{1cm} \text{measurement vector at time } t_k, p \times 1 \\
\( \lambda(t) \) \hspace{1cm} \text{costate vector, } n \times 1 \\

Chapter 4, 5, 6

\( B_0 \) \hspace{1cm} \text{matrix indicating types of initial condition, } 2n \times 2n \\
\( B_f \) \hspace{1cm} \text{matrix indicating types of final condition, } 2n \times 2n \\
\( \xi \) \hspace{1cm} \text{vector containing states and costates, } 2nk \times 1 \\
\( C \) \hspace{1cm} \text{dependent variable state matrix, } 2n \times 2n \\
\( D_f \) \hspace{1cm} \text{matrix for the jumps in the dependent variables, } 2n \times 2n
vector of forcing functions, $2n \times 1$

$I$ identity matrix, $n \times n$

$L$ integration operator

$M$ number of measurements

$t_j$ time at interval $j$

$t_j^-$ time $t_j$ before using the measurements

$t_j^+$ time $t_j$ after using the measurements

$\mathbf{y}_j(t_j^+) = 2n \times 1$, forcing solution vector at $t_j^+$

$\mathbf{y}_j(t_j^-) = 2n \times 1$, forcing solution vector from $t_{j-1}^-$ to $t_j$

$\mathbf{Y}_j(t_j^-)$ 2n × 1, fundamental solution matrix at $t_{j-1}^-$

$\mathbf{Y}_j(t_j)$ 2n × 1, fundamental solution vector from $t_{j-1}$ to $t_j$

$y(t)$ vector of dependent variables, states and costates, $2n \times 1$

$\dot{y}(t)$ derivative of vector of dependent variables

$\mathbf{Y}_j$ vector of jumps in the dependent variables at $t_j$, $2n \times 1$

$\beta$ vector containing values of the initial and final conditions in the states or costates

$\mathbf{D}_{1\times 1}$ vector of dynamic error terms evaluated $l$ times

$f_j$ linear / nonlinear function of the estimated states

$l$ number of correction samples employed in least-squares

$M_{l\times q}$ matrix of linear and nonlinear functions evaluated $l$ times

$\mathbf{P}_{q\times 1}$ vector of $q$ coefficients

$\alpha$ arbitrary coefficient

$\beta$ arbitrary coefficient

$\gamma$ arbitrary coefficient

$\Phi$ least squares error function

$C(x, y)$ cross-correlation coefficient between discrete variables $x$ and $y$

$f$ linear / nonlinear function of the estimated states

$K$ arbitrary constant

$n$ number of data points

$\bar{x}_i$ expected value of $x_i$
\(x_i\) a discrete variable
\(y_i\) a discrete variable
\(\eta\) noise
\(\sigma_i\) standard deviation of variable \(i\)
\(x\) position
\(\dot{x}\) velocity
\(\ddot{x}\) acceleration

Chapter 7

\(C\) system effective damping matrix
\(D\) plate stiffness
\(E\) Modulus of Elasticity
\(h\) plate thickness
\(I\) identity matrix
\(K\) system effective damping matrix
\(l\) plate length
\(M\) Mach speed
\(N_x\) in-plane force due to vertical deflection
\(N_{x0}\) initial in-plane forces
\(p - p_{\infty}\) aerodynamic pressure
\(\Delta p\) static pressure differential
\(P\) static pressure vector
\(S\) snapshot matrix
\(t\) time variable
\(U\) fluid velocity
\(V\) eigenvector matrix
\(w\) plate deflection
\(W\) Non-dimensionalized deflection \((w/h)\)
\(x\) Cartesian coordinate
\(\beta\) \(\sqrt{M^2 - 1}\)
\begin{itemize}
  \item $\nu$: poisson’s ratio
  \item $\xi$: non-dimensionalized coordinate $(x/a)$
  \item $\rho_f$: fluid mass density
  \item $\rho_s$: plate mass density
  \item $\tau$: non-dimensionalized time variable
\end{itemize}
# Contents

1 Introduction .................................................. 1

2 Background .................................................. 4
   2.1 Thesis Overview ........................................... 9

3 Identification Algorithm ................................... 12
   3.1 MME Algorithm ............................................ 12
   3.2 Multiple Shooting ....................................... 18

4 Correlation and Coefficient Identification ................. 25
   4.1 Least Squares Algorithm ................................. 25
   4.2 Correlation ................................................ 27

5 Conservative Single-Degree-of-Freedom Nonlinear Systems 32
   5.1 Single Degree of Freedom System ....................... 32
      5.1.1 Simulation ........................................... 34
   5.2 Position and Velocity Measurement ..................... 36
      5.2.1 Noiseless Measurements ............................ 36
      5.2.2 Measurements with noise .......................... 44
   5.3 Acceleration Measurement ................................. 58
      5.3.1 Multiple Shooting Algorithm development ........ 58
      5.3.2 Noiseless Measurements ............................ 63
      5.3.3 Measurements with noise .......................... 68
   5.4 Conclusion ................................................ 81
6 Conservative Two-Degree-of-Freedom Nonlinear Systems 82
   6.1 Two Degree of Freedom System .............................. 82
       6.1.1 Simulation ........................................... 85
   6.2 Position and Velocity Measurement .......................... 87
       6.2.1 Noiseless Measurements .............................. 87
       6.2.2 Measurements with 5% Noise ........................ 92
       6.2.3 Measurements with 10% Noise ........................ 96
   6.3 Acceleration Measurement .................................... 101
       6.3.1 Multiple Shooting algorithm development .............. 101
       6.3.2 Measurements with noise ............................. 104

7 Validation 123
   7.1 System Definition and Equation of Motion .................... 123
   7.2 Data Generation ............................................ 126
   7.3 System Identification ....................................... 126
   7.4 Correlation and Coefficient Identification ................. 128
   7.5 System Identification with True Linear Model ............... 137
       7.5.1 Position and Velocity Measurement .................... 137
       7.5.2 Acceleration Measurement ............................. 141
   7.6 Conclusion .................................................. 145

8 Conclusions and Future work 146
   8.1 Summary and Conclusions .................................... 146
   8.2 Recommendations for Future work ............................ 147

Bibliography 149

A MME evaluation of SDOF System, Accel. Data Only 153

B Subroutine for Numerical Integration of One Degree of Freedom System 161

C Numerical Integration of 2DOF System, 2 Nonlinearities 162
<table>
<thead>
<tr>
<th></th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>MME Evaluation of Two Degree of Freedom System Acceleration Data Only</td>
<td>163</td>
</tr>
<tr>
<td>E</td>
<td>Numerical Integration of Nonlinear Two DOF System</td>
<td>172</td>
</tr>
<tr>
<td>F</td>
<td>2DOF ID Code Using Position and Velocity</td>
<td>173</td>
</tr>
<tr>
<td>G</td>
<td>Program for Reducing the 64 Node Data to a SDOF Data</td>
<td>181</td>
</tr>
<tr>
<td>H</td>
<td>Program for Minimum Model Error Identification</td>
<td>183</td>
</tr>
<tr>
<td>I</td>
<td>Program for Nonlinear Function and Coefficient Identification</td>
<td>188</td>
</tr>
<tr>
<td>J</td>
<td>Code for Numerical Integration of the Identified Model</td>
<td>190</td>
</tr>
<tr>
<td>K</td>
<td>MME Code for Panel Flutter, Acceleration Data Only</td>
<td>191</td>
</tr>
</tbody>
</table>
List of Figures

5.1 Single DOF Nonlinear Spring-Mass System ........................................... 33
5.2 True Position (solid) and MME Estimated (+) without noise .................. 38
5.3 True Velocity (solid) and MME Estimated (+) without noise .................. 38
5.4 True Acceleration (solid) and MME Estimated (+) without noise ............. 39
5.5 True Model Error \( d_2 \) (solid) and MME Estimated (+) without noise ....... 39
5.6 True Position (solid) and MME Estimated (+) with 1% noise ................ 48
5.7 True Velocity (solid) and MME Estimated (+) with 1% noise ................ 48
5.8 True Acceleration (solid) and MME Estimated (+) with 1% noise ............ 49
5.9 True Model Error \( d_2 \) (solid) and MME Estimated (+) with 1% noise ....... 49
5.10 True Position (solid) and MME Estimated (+) with 5% noise ................. 52
5.11 True Velocity (solid) and MME Estimated (+) with 5% noise ................. 52
5.12 True Acceleration (solid) and MME Estimated (+) with 5% noise .......... 53
5.13 True Model Error \( d_2 \) (solid) and MME Estimated (+) with 5% noise ....... 53
5.14 True Position (solid) and MME Estimated (+) with 10% noise ............... 56
5.15 True Velocity (solid) and MME Estimated (+) with 10% noise ............... 56
5.16 True Acceleration (solid) and MME Estimated (+) with 10% noise ......... 57
5.17 True Model Error \( d_2 \) (solid) and MME Estimated (+) with 10% noise ....... 57
5.18 True Position (solid) and MME Estimated (+) with no noise ................. 66
5.19 True Velocity (solid) and MME Estimated (+) with no noise ................. 66
5.20 True Acceleration (solid) and MME Estimated (+) with no noise .......... 67
5.21 True Model Error \( d_2 \) (solid) and MME Estimated (+) with no noise ....... 67
5.22 True Position (solid) and MME Estimated (+) with 1% noise ............... 71
5.23 True Velocity (solid) and MME Estimated (+) with 1% noise ............... 71
5.24 True Acceleration (solid) and MME Estimated (+) with 1% noise . . . . . . . 72
5.25 True Model Error $d_2$ (solid) and MME Estimated (+) with 1% noise . . . . 72
5.26 True Position (solid) and MME Estimated (+) with 5% noise . . . . . . . 75
5.27 True Velocity (solid) and MME Estimated (+) with 5% noise . . . . . . . 75
5.28 True Acceleration (solid) and MME Estimated (+) with 5% noise . . . . . 75
5.29 True Model Error $d_2$ (solid) and MME Estimated (+) with 5% noise . . . . 76
5.30 True Position (solid) and MME Estimated (+) with 10% noise . . . . . . . 76
5.31 True Velocity (solid) and MME Estimated (+) with 10% noise . . . . . . . 79
5.32 True Acceleration (solid) and MME Estimated (+) with 10% noise . . . . . 79
5.33 True Model Error $d_2$ (solid) and MME Estimated (+) with 10% noise . . . . 80

6.1 Two DOF Nonlinear Spring-Mass System . . . . . . . . . . . . . . . . . . . . . . . . 83
6.2 True Position (solid) and MME Estimated (+) with no noise . . . . . . . . . 90
6.3 True Velocity (solid) and MME Estimated (+) with no noise . . . . . . . . . 90
6.4 True Model Error-$d_3$ (solid) and MME Estimated (+) with no noise . . . . 91
6.5 True Model Error-$d_4$ (solid) and MME Estimated (+) with no noise . . . . 91
6.6 True Position (x) (solid - no noise, 'o' - with 5% noise) and MME Estimated (+) 95
6.7 True Velocity ($\dot{x}$) (solid - no noise, 'o' - with 5% noise) and MME Estimated (+) 95
6.8 True Model Error-$d_3$ (solid) and MME Estimated (+) . . . . . . . . . . . . . 96
6.9 True Model Error-$d_4$ (solid) and MME Estimated (+) . . . . . . . . . . . . . 96
6.10 True Position (x) (solid - no noise, 'o' - with 10% noise) and MME Estimated (+) 99
6.11 True Velocity ($\dot{x}$) (solid - no noise, 'o' - with 10% noise) and MME Estimated (+) 99
6.12 True Model Error $d_3$ (solid) and MME Estimated (+) . . . . . . . . . . . . . 100
6.13 True Model Error $d_4$ (solid) and MME Estimated (+) . . . . . . . . . . . . . 100
6.14 True Position($x_1$) (solid) and MME Estimated (+) . . . . . . . . . . . . . 107
6.15 True Position($x_2$) (solid) and MME Estimated (+) . . . . . . . . . . . . . 107
6.16 True Velocity ($\dot{x}_1$) (solid) and MME Estimated (+) . . . . . . . . . . . . . 108
6.17 True Velocity ($\dot{x}_2$) (solid) and MME Estimated (+) . . . . . . . . . . . . . 108
6.18 True Acceleration($\ddot{x}_1$) (solid - no noise, 'o' - with 5% noise) and MME Estimated (+) 109
6.19 True Acceleration ($\ddot{x}_2$) (solid - no noise, 'o' - with 5% noise) and MME Estimated (+) 109
6.20 True Model Error($d_3$) (solid) and MME Estimated (+) .......................... 110
6.21 True Model Error($d_4$) (solid) and MME Estimated (+) .......................... 110
6.22 True Position($x_1$) (solid) and MME Estimated (+) .............................. 113
6.23 True Position($x_2$) (solid) and MME Estimated (+) .............................. 113
6.24 True Velocity ($\dot{x}_1$) (solid) and MME Estimated (+) ....................... 114
6.25 True Velocity ($\dot{x}_2$) (solid) and MME Estimated (+) ....................... 114
6.26 True Acceleration ($\ddot{x}_1$) (solid - no noise, ‘o’ - with 10% noise) and MME Estimated (+) 115
6.27 True Acceleration ($\ddot{x}_2$) (solid - no noise, ‘o’ - with 10% noise) and MME Estimated (+) 115
6.28 True Model Error($d_3$) (solid) and MME Estimated (+) .......................... 116
6.29 True Model Error($d_4$) (solid) and MME Estimated (+) .......................... 116
6.30 True Position($x_1$) (solid) and MME Estimated (+) .............................. 119
6.31 True Position($x_2$) (solid) and MME Estimated (+) .............................. 119
6.32 True Velocity ($\dot{x}_1$) (solid) and MME Estimated (+) ....................... 120
6.33 True Velocity ($\dot{x}_2$) (solid) and MME Estimated (+) ....................... 120
6.34 True Acceleration ($\ddot{x}_1$) (solid - no noise, ‘o’ - with 15% noise) and MME Estimated (+) 121
6.35 True Acceleration ($\ddot{x}_2$) (solid - no noise, ‘o’ - with 15% noise) and MME Estimated (+) 121
6.36 True Model Error ($d_3$) (solid) and MME Estimated (+) ....................... 122
6.37 True Model Error ($d_4$) (solid) and MME Estimated (+) ....................... 122
7.1 Simply supported panel subject to supersonic airflow .............................. 124
7.2 Steady State Panel deflection at t=8 secs. ......................................... 127
7.3 Position profile of Node 32 with time ............................................. 127
7.4 Velocity profile of Node 32 with time ............................................. 128
7.5 Steady State generalized first modal position data .................................. 129
7.6 Steady State generalized first modal velocity data .................................. 129
7.7 Position data used for MME ............................................................... 130
7.8 Velocity data used for MME ............................................................... 130
7.9 Model Errors identified by the MME, $d_2$(solid) and $d_1$(+) .................. 131
7.10 The Model Error term $d_2$(solid) and $d_2a$(+) .................................. 132
7.11 The First Model Error term ($d_2$ (solid)) and Model Error term after the fourth iteration ($d_2d$ (dot))
7.12 MME Estimated position (+) and True position (solid) 136
7.13 MME Estimated velocity (+) and True velocity (solid) 136
7.14 MME Estimated position (+) and True position (solid) 140
7.15 MME Estimated velocity (+) and True velocity (solid) 140
7.16 Acceleration data used for MME with 10% noise 141
7.17 Positions: True (solid) and Mathematical model (+) 144
7.18 Velocity: True (solid) and Mathematical model (+) 144
7.19 Acceleration: True with 10% noise (solid) and Mathematical model (+) 145
List of Tables

5.1 CORRELATION COEFFICIENTS FOR DATA WITH NO NOISE . . . . . . 40
5.2 CORRELATION COEFFICIENTS FOR DATA WITH 1% NOISE . . . . . 46
5.3 CORRELATION COEFFICIENTS FOR DATA WITH 5% NOISE . . . . . 50
5.4 CORRELATION COEFFICIENTS FOR DATA WITH 10% NOISE . . . . 54
5.5 CORRELATION COEFFICIENTS FOR DATA WITH NO NOISE . . . . . 64
5.6 CORRELATION COEFFICIENTS FOR DATA WITH 1% NOISE . . . . . 69
5.7 CORRELATION COEFFICIENTS FOR DATA WITH 5% NOISE . . . . . 73
5.8 CORRELATION COEFFICIENTS FOR DATA WITH 10% NOISE . . . . 77

6.1 CORRELATION COEFFICIENTS FOR DATA WITH NO NOISE . . . . . 88
6.2 CORRELATION COEFFICIENTS FOR DATA WITH 5% NOISE . . . . . 93
6.3 CORRELATION COEFFICIENTS FOR DATA WITH 10% NOISE . . . . 97
6.4 CORRELATION COEFFICIENTS FOR DATA WITH 5% NOISE . . . . . 105
6.5 CORRELATION COEFFICIENTS FOR DATA WITH 10% NOISE . . . . 112
6.6 CORRELATION COEFFICIENTS FOR DATA WITH 15% NOISE . . . . 118

7.1 CORRELATION COEFFICIENTS AFTER FIRST ITERATION . . . . . . 131
7.2 CORRELATION COEFFICIENTS AFTER SECOND ITERATION . . . . . 133
7.3 CORRELATION COEFFICIENTS AFTER THIRD ITERATION . . . . . 134
7.4 CORRELATION COEFFICIENTS AFTER FOURTH ITERATION . . . . . 134
7.5 NONLINEAR FUNCTION COEFFICIENTS AFTER EVERY ITERATION 138
7.6 NONLINEAR FUNCTION COEFFICIENTS AFTER EVERY ITERATION WITHOUT NOISE 142
7.7 NONLINEAR FUNCTION COEFFICIENTS AFTER EVERY ITERATION WITH 10% NOISE 143
Acknowledgement

I would like to thank my advisor, Dr. Joseph Slater, for the opportunity to conduct this research. His insights and enthusiasm toward the research have been both stimulating and encouraging.

I would also like to thank Sean A. Mortara, Dr. Mitch Wolff and Dr. Pradeep Misra for all the help I received during the development of this thesis.

To my parents and my sister, whose unconditional love and support have gone well beyond reasonable expectation and made this possible, I will be forever grateful.
This thesis is dedicated to my father G. Gopinathan Pillai
and mother B. Rajarajeswari
Chapter 1

Introduction

A large number of applications exist in the general area of “post experiment” estimation, wherein estimates of the actual state histories of a dynamic system are obtained using an assumed state dynamic model and sets of discrete measurements. Applications are found throughout engineering, but are especially numerous in such aerospace problems as orbit estimation, attitude estimation, and postflight trajectory estimation.

In general, both the dynamic model and the available measurements are imperfect. The estimation algorithm combines the model-predicted state estimates with the available measurements in such a way as to obtain estimates of the state histories which are of higher accuracy and more complete than either the model predictions or the measurements. If the estimation algorithm optimizes some performance index, typically based on the estimated error in state measurements, then the resulting state estimate is said to be optimal.

The most commonly used estimation approach is the Kalman filter and numerous closely related strategies, by Kalman and Kalman and Bucy. The Kalman algorithms are well suited for real-time estimation because of their sequential processing structure and emphasis on the most recent data. As a result of popularity and familiarity, Kalman filters are
now routinely used for post experiment estimation, normally accomplished via iterative
postprocessing of the filter estimate.

The various filter strategies use similar, although distinct, optimality criteria. The two
most common are “minimum variance” and “maximum likelihood” criteria. In minimum
variance estimation, a function of the trace (typically the trace itself) of the state estimate
covariance matrix is minimized. In maximum likelihood estimation, the most probable
state estimate is found given the measurements. The important feature of these existing
strategies is that they require estimation of the state estimate error covariance. In order
to rigorously estimate the state estimate error, knowledge of both the model error and the
measurement error is required. While measurements errors may be determined in numerous
ways, model errors are generally unknown by definition (i.e., if one were aware of the
model error, one would correct the model).

The widespread existence of nonlinear behaviour in dynamic systems is well docu-
mented, e.g. Thompson and Stewart [1] and Nayfeh and Mook [2]. In particular, virtually
every problem associated with flight trajectory estimation, spacecraft dynamics, large de-
flection etc., is known to exhibit nonlinear behaviour. Many excellent methods for analyz-
ing nonlinear system models have been developed. However, a key practical link is often
overlooked, namely: How does one obtain an accurate mathematical model for the dynam-
ics of a particular complicated nonlinear system? General methods for obtaining accurate
models for real physical systems are not nearly as widespread or well developed as are the
techniques available for analyzing models.

In this thesis a nonlinear identification technique is presented which obtains accurate
models of the nonlinear systems with little or no a priori knowledge of the type of nonlin-
earity involved. Given discrete time-domain measurements and an assumed model, which contains what is known of the system, the Minimum Model Error estimation algorithm (MME) is used to obtain estimates of the dynamic model error and the state trajectory. A correlation technique is employed to select the best functional basis for the dynamic correction from an extensive library. A least squares fit is performed to find the respective coefficients of the model error function(s). Examples are presented using data from analog simulations. The tests demonstrate that the method is robust with respect to the poor ignorance of the nonlinear system model, robust for short measurement record length, and robust regardless of initial conditions. It works well in the presence of noise and is demonstrated for systems up to third order.
Chapter 2

Background

Exact solutions for nonlinear systems are generally not available. Hence recourse is made to approximate analysis including purely analytical techniques, purely numerical techniques, and numerical-perturbation techniques. The purely analytical techniques are applicable to systems with simple geometries, composition, and boundary conditions. Purely numerical techniques may involve the use of finite difference in both space and time, finite differences in time and finite elements in space, and finite elements in both space and time. These purely numerical techniques are especially costly for two and three-dimensional systems.

Numerous methods for the identification of nonlinear systems have been developed in the past (Billings [29] and Natke, Juang and Gawronski [28]). Most of these techniques fall into one of the following categories:

- Describing the nonlinear system using a linear model.

- Representing the nonlinear system in a series expansion, and obtaining the respective coefficients either by using a regression estimation technique, by minimizing a cost
functional, by using correlation techniques, or by some other approach.

- Obtaining a graphical representation of the nonlinear term(s), then finding an analytical model for the nonlinearity.

The first two approaches circumvent the problem of finding the nonlinear function form by either linearizing the problem or selecting a functional form a priori. The preselected analytical form may be parametric or nonparametric. Examples of the third approach are seldom found in literature. In the third technique a criterion is used for selecting the functional form of the nonlinearity, instead of assuming a particular form. The nonlinear identification technique used in this paper is an example of such an approach.

With such diversity of nonlinear identification techniques the choice of a particular algorithm may be based on criteria such as: the degree to which prior assumptions of the model form affect the user’s effort in applying the algorithms; the number of iterations required; the sensitivity to the presence of measurement noise in the data; the number of state measurements needed; whether or not knowledge of the initial conditions is required; the kind of forcing inputs required or permitted (step, white gaussian noise, sinusoidal etc.); the ability to handle hysteretic or discontinuous non-linearities; the degree of a priori knowledge of system properties required; and/or the computational requirements. Most algorithms differ widely in at least some of these comparisons; the choice of a particular technique depends on the needs of the particular application.

Among the methods which linearize the nonlinear system are those presented by Jedner and Unbehauen [7] wherein a nonlinear system often operates in small regions around a number of operating points, is represented by an equal number of linear submodels. It is
assumed that the system operates only at a few points. Although the model may work well for controller design, the points at which the system is operating must be known and the linear models apply only within the operating regions.

The direct equation approach is used by Yasuda, Kawamura and Watanabe [8],[9]. The input and output measurements of a dynamic process are expressed as a Fourier Series. The system nonlinearity is represented as a sum of the polynomials with unknown coefficients. Applying the principle of harmonic balance, the polynomial coefficients as well as the other system parameters are obtained accurately. Knowledge of the nonlinearity is needed to construct the polynomial. Truncation in the Fourier Series expansion of the input or output may lead to error.

The regression estimation approach is used by Billings and Voon [10], and Greblick and Pawlack [11]. In other techniques, as in statistical linearization, a nonlinear relation is replaced by a linear equivalent gain. Broersen [12], extends the technique of statistical linearization by representing the nonlinearity as a linear combination of a number of arbitrary functions. The number and type of the functions selected depends on the desired accuracy as well as some knowledge of system nonlinearity. Although some of the basic properties of the true nonlinear output are preserved, it is limited to only random excitation, and knowledge of all states and forcing terms is required.

In the method of multiple scales (Hanagud, Mayyappa and Craig [13]), a perturbation solution to the nonlinear equation of motion is obtained. An objective function is built employing an integral least squares approach. The minimization of the functional yields the unknown parameters. Data on only one field variable is necessary, and the method is effective in the presence of high noise. The method typically requires some algebraic
manipulation which may be quite involved, and these manipulations are only valid for a particular assumed nonlinear form. If the assumed nonlinear form is changed, the algebra must be repeated.

Other popular series used in nonlinear identification are orthogonal polynomials such as Legendre (Wang and Chan [14]), and Jacobi (Horn and Chou [15]). Horn and Chou expand the variables of the system into a shifted Jacobi series, reducing the nonlinear state equation into a algebraic matrix equation. The unknown parameters of the nonlinear system are then estimated using least squares. Even though the algorithm works well in the presence of noise, the nonlinear form must be known a priori.

Hammond, Lo and Seager-Smith [16] use an identification technique based on optimal control methods employed for linear system deconvolution. The form of the linear model is assumed to be known as well as the input and the output. A cost functional consisting of the weighted sum of the square of the error (between the actual and estimated output) yields an optimal estimated input. The estimated input and the actual input are used to obtain the nonlinearity as a function of the state variables. Although no previous assumption is made of the non-linearities, there is no provision to deal with noise.

All of the techniques outlined above have proven useful in certain applications. However, all of them are subject to one or more of the following shortcomings:

- The form of the nonlinearity (quadratic, cubic, exponential, etc.) must be assumed a priori. This is very serious drawback because the identification algorithm can only attempt to find the best model in the assumed form. If the form is assumed incorrectly, the resulting model may be so poor as to be useless or it may appear to fit the
data well enough that the user erroneously concludes that the correct model has been obtained. Also, for many techniques of this type, the effort required to test a given form is considerable, which greatly diminishes the effectiveness since multiple form tests are less likely to be conducted.

- Techniques which attempt to avoid the problem of a priori model form assumption through the use of series expansion generally eliminate any possibility of understanding the underlying physics. Thus, although a good fit of the data might be achieved using a sufficient number of terms in the series, physical insight is lost. Moreover, large systems and/or particularly complicated behaviour may require that a very large number of terms be used to obtain a given level of accuracy.

- The presence of noise in the measurement data is not rigorously treated, yet noise is generally unavoidable.

- Initial conditions must be known in order to implement the algorithm.

- The algorithm can only be implemented if the data is obtained using very specific system excitations.

The algorithm using Minimum Model Error (MME) developed by Mook and Stry [6], which has been used here to determine the nonlinearity in the system, compares favorably with the other algorithms in most of the categories listed above. It is robust with respect to measurement noise; does not require knowledge of initial conditions; is independent of the forcing (but like all other methods, assumes that it is known); is not computationally prohibitive; and, most importantly, it requires minimal a priori assumptions regarding the
form of the model or the system properties. In fact the use of the correlation technique in the algorithm virtually eliminates the need to ever assume the nonlinear model form.

2.1 Thesis Overview

In Chapter 3 the nonlinear identification technique is explained in detail. The MME algorithm is explained in Section 3.1. Multiple shooting coupled with MME aids the nonlinear system identification technique by facilitating the solution of the state and error trajectories. The multiple shooting algorithm is explained in Section 3.2.

Chapter 4 describes the final step in the identification procedure, to use a least-squares algorithm to fit the model error to the functional forms. The correlation technique used to determine the function having the highest correlation coefficient is explained. The error term is expanded into a combination of the determined functional forms, and the least squares algorithm used to determine the coefficients for these functions is mentioned.

Chapter 5 is a compilation of examples in which the nonlinear identification technique is tested using simulated measurements for a Single Degree of Freedom system. Section 5.1 describes the simulation process. Section 5.2 shows the MME procedure implementation for the system when both the position and velocity are known. The tests demonstrate that the identification technique works well for identification of nonlinear systems in the presence of noise. It is shown that the method does not require an extensive number of measurements. Section 5.3 demonstrates the derivation of the MME algorithm for a system when only the acceleration measurements are available. A couple of examples demonstrate the convergence of the MME to the actual states after a few iterations. The correlation tech-
The correlation technique selects the actual nonlinear feedback error term(s) in the presence of noise and when the dynamic error consists of more than one function.

Chapter 6 is a compilation of examples that demonstrate MME for Two Degree of Freedom Systems. It is shown that the technique is capable of identifying a nonlinear model which accurately reproduces the analog output regardless of initial conditions. Although the true model is unknown, the identified model accurately reproduces the measurements.

Chapter 7 shows the validation of the code for a true nonlinear experimental data where the nonlinearity is not known. The nonlinear panel flutter problem, refer Dowell [26], is considered for the method and code validation. To begin with a simple linear model is assumed and the nonlinear system is identified. The position and velocity vectors are available. The data is used by MME for model error identification and subsequently this model error is used for nonlinear function and coefficient identification yielding the nonlinear terms in the modal equation that cause the unstable linear system to exhibit limit cycle oscillation. This results in a mathematical model for the system. To check if the mathematical model is the true system model, the identified model is numerically integrated and the position and velocity vectors are compared with the true position and velocity vector. It is shown that both are in very close agreement and hence the mathematical model obtained is a very good approximation of the true system model. The linear model is then modified to include the true linear stiffness and damping terms and the system model is identified using MME. A comparison is made between the two identified results.

Suggestion for future work and a brief summary and conclusion appear in Chapter 8. Finally the list of all the program codes used for the nonlinear identification technique are
included in the Appendix.
Chapter 3

Identification Algorithm

The nonlinear identification technique consists of three major parts. The first part is the Minimum Model Error (MME) estimation technique, which, coupled with multiple shooting, computes the state and dynamic error trajectories given an assumed model and measurements of the nonlinear system in question. The second part is the correlation technique, which selects from an extensive function library, the functional form(s) of the dynamic error term. Finally, least-squares is employed to obtain the coefficients of the functional forms.

The MME estimation technique and the multiple shooting algorithm is explained in this Chapter. The correlation technique and the Least squares is explained in Chapter 4.

3.1 MME Algorithm

The MME estimation part of the identification technique is presented here in detail. The aim of MME is to find what corrections must be added to what is already known of the system model such that the model plus the correction yield an accurate representation of
the system behaviour.

The MME algorithm uses discrete time-domain measurements and an assumed model. This model is typically linear and contains what is already known of the system. A correction term, that contains the still unknown (not modeled) dynamics of the nonlinear system, is added to the assumed model and a cost functional formed. Minimization of this cost functional yields the value of the dynamic error term. The assumed model plus the functional form of the correction term yield an accurate model of the nonlinear system.

Consider a forced nonlinear dynamic system which may be modeled in state-space form by the equation

$$\dot{x}(t) = Ax(t) + Bu(t) + f(x(t), \dot{x}(t))$$

where $x(t)$ is a $n \times 1$ state vector consisting of the system states, $A$ is the $n \times n$ state matrix, $Bu(t)$ is an $n \times 1$ vector of known external excitation, and $f(x(t), \dot{x}(t))$ is an $n \times 1$ vector which includes all of the system non-linearities. And the dot represents differentiation with respect to time. Equation (3.1) is denoted as the true or actual system model. It is assumed that a model such as the one presented in Equation (3.1) exists describing any kind of nonlinear dynamic system. State-observable discrete time domain measurements are available for this system in the form

$$\ddot{y}(t_k) = g_k(x(t_k), t_k) + \omega_k, \quad t_0 \leq t_k \leq t_f$$

where $\ddot{y}(t_k)$ is an $p \times 1$ measurement vector at time $t_k$. $p$ is equal to $n$, the system dimension, if all the states are measured. In general, however, the number of observable states is less than $n$. The variable $g_k$ is the accurate model of the measurement process, and $\omega_k$ represents
measurement noise. The variable $v_k$ is assumed to be a zero mean, gaussian distributed process of known covariance $R_k$.

To implement MME an assumed model is built. This model is presumably derived from traditional physics, but it does not fully represent the behaviour of the system, possibly because the model itself is linear while the system in question is nonlinear or because the system is too complicated to be accurately modeled by traditional physics. The assumed linear model is given in state space form as

$$\dot{x}_i(t) = A_i\tilde{x}_i(t) + Bu_i(t)$$  \hspace{1cm} (3.3)

where $\tilde{x}_i$ is the state vector estimate obtained from the linear model and $A_i$ is the linear model $n \times n$ state matrix. The forcing $Bu_i(t)$ is assumed to be known. Although in practice a linear model is most easily obtained, the assumed model in Equation (3.3) may be nonlinear. MME uses the assumed model in Equation (3.3) and the noisy measurements in Equation (3.2) to find the dynamic model error. Note, that because the system in Equation (3.3) lacks some information, the linear state matrix $A_i$ is not necessarily equal to $A$ in Equation (3.1).

The dynamic model error which includes the unknown nonlinear terms of the system, as well as some linear terms depending on the completeness of Equation (3.3). The model error is represented by the addition of a correction term to the assumed linear model as

$$\dot{\hat{x}}(t) = A_i\hat{x}(t) + Bu_i(t) + \Delta(t)$$  \hspace{1cm} (3.4)

where $\Delta(t)$ is the $n \times 1$ model error (correction term) to be estimated later and $\hat{x}(t)$ is the state vector estimate obtained from the corrected linear model. The estimated state vector
\( \hat{x}(t) \) is equal to the true state, \( \bar{x}(t) \) in Equation (3.1) within experimental error, when the assumed model has been fully corrected by \( d(t) \) as shown in Equation (3.4).

A cost functional, \( J \), is built which consists of the weighted integral square of the correction term plus the weighted sum square of the measurement-minus-estimated residuals. The form of \( J \) is

\[
J = \sum_{k=1}^{M} \left( [\hat{y}(t_k) - g_k(\bar{x}(t_k), t_k)]^T R_k^{-1} [\hat{y}(t_k) - g_k(\bar{x}(t_k), t_k)] \right) + \int_{t_0}^{t_f} d(\tau)^T W d(\tau) d\tau
\]

(3.5)

where \( W \) is a weight matrix of to-be-determined value. \( J \) is affected by how well the estimated states match the measured states as well as how much correction \( d(t) \) the assumed model needs. For a given assumed model (not equal to the true model), if \( d(t) \) is small the summation of measurement residuals is small. The value of \( d(t) \) depends on \( W \). Therefore, the value of \( W \) is set to yield the \( d(t) \) needed to correct the assumed model.

The functional \( J \) is minimized with respect to the model error term \( d(t) \) to find the optimum value of the dynamic correction term. The necessary conditions for the minimization lead to the following two-point boundary-value problem (TPBVP), (for details see Geering [17]),

\[
\dot{x}(t) = A(t) \dot{x}(t) + B(t) u(t) + d(t)
\]

(3.6)

\[
\dot{\lambda}(t) = -A(t)^T \lambda(t)
\]

(3.7)

\[
d(t) = -\frac{1}{2} W^{-1} \lambda(t)
\]

(3.8)

\[
\lambda(t^+_{k}) = \lambda(t^-_{k}) + 2H_k R_k^{-1} [\hat{y}(t_k) - g_k(\bar{x}(t_k), t_k)]
\]

(3.9)
\[ H_k = \frac{\delta g}{\delta \lambda} \big|_{\lambda(t_k), t_k} \]  

(3.10)

\[ \lambda(t_0) = \lambda_0 \quad \text{or} \quad \lambda(t_0) = 0 \]  

(3.11)

\[ \lambda(t_f) = \lambda_f \quad \text{or} \quad \lambda(t_f) = 0 \]  

(3.12)

where \( \lambda(t) \) is a vector of co-states (Lagrange Multipliers) and \( H_k \) is the derivative of the measurement function. It equals \( I \), the identity matrix, if the states are measured directly.

The boundary conditions are selected such that either \( \lambda(t_0) \) is specified or \( \lambda(t_0) = 0 \) at the initial time and either \( \lambda(t_f) \) is specified or \( \lambda(t_f) = 0 \) at the final time. If all initial or all final conditions are available, Equations (3.6) and (3.7) are integrated forward if the initial conditions are given, or backwards if the final conditions are available. The jump in Equation (3.9) is calculated, then the correction term obtained at each time step \( t_k \) from Equation (3.8).

Equations (3.6-3.9) define an optimum problem for a given \( W \). In order to determine the optimal solution the value of \( W \) is varied until the “covariance constraint” is satisfied. According to the covariance constraint (see Mook and Junkins [3]) the measurement-minus-estimate error covariance must match the measurement-minus-truth error covariance matrix. In other words we want the error covariance which exists between the truth and the measurements to be approximately equal to the error covariance between the estimated states and the measurements. Mathematically speaking

\[ [\tilde{y}(t_m) - g_m(\bar{x}(t_m), t_m)]^T [\tilde{y}(t_m) - g_m(\bar{x}(t_m), t_m)] \approx R_m, \quad m = 1, 2, \ldots, k \]  

(3.13)

where \( R_m \) is obtained from the measuring equipment manufacturer data or formed by the experimentalist according to the reliability of the measurements. During the minimization
the weight $W$ is varied until the state estimates satisfy the covariance constraint. If the assumed model is identical to the true model no dynamic correction is needed, and the absolute value of $W$ is set very large. When $W = \infty$ then $d(t) = 0$. If the assumed model is significantly different from the true model, i.e., the states in Equation (3.3) do not match the measurements, the value of $W$ is decreased until $d(t)$ is large enough to compensate for what is missing in the assumed model. The correction term or model error is, therefore, the minimum adjustment required for the estimated states to match the measurements with the same error covariance as the truth matches the measurements. Summarizing, the solution of the boundary-value problem Equations (3.6-3.12) is repeated until optimal estimates of the states and of the dynamic model error is produced. A correlation technique identifies the functional form of the dynamic error term and the final step in the procedure is to use a least-squares algorithm to find the respective function coefficients. As shown in Equation (3.9) the TPBVP resulting from the minimization of Equation (3.5) contains jumps in the co-states and, consequently, in the correction term. The size of the jump is directly proportional to the measurement residual at each measurement time. Jumps in the dynamic error term may increase the difficulty in finding its functional form and detract from its accuracy. If the measurements have little noise in relation to the correlation magnitude, the dynamic error estimate is still smooth. Jumps become a problem when measurement noise is high and the identification accuracy may be degraded. Nevertheless, in all cases state estimates are good. A multiple shooting algorithm, as presented by Mook and Lew [19], converts the TPBVP of Equations (3.6-3.12) into a set of linear algebraic equations which may be easily solved using any linear equation solver. By setting the initial and final co-state values to zero, no knowledge of the state initial or final values are necessary. Multiple shooting also
facilitates the analysis of a large number of measurements by processing the solution at the end of every set of time intervals.

### 3.2 Multiple Shooting

The TPBVP necessary conditions in Equations (3.6-3.12) produce jump discontinuities in the co-states at the measurement time $t_j$. These discontinuities reduce numerical stability and increase the sensitivity of the integration to the unknown boundary conditions. These problem are alleviated by using a stable numerical algorithm such as a multiple shooting method to solve the TPBVP. Multiple shooting also helps in other ways by eliminating the need to guess at initial or final conditions and by computing the solution at time multiplies such that the algorithm can handle problems of bigger size. In multiple shooting the TPBVP necessary conditions are converted into a set of algebraic equations which produce an exact solution for linear problems. Since most of the time the nonlinear identification algorithm employs a linear assumed model, using multiple shooting poses no restrictions, at least unless the dynamic error term is composed of more than one nonlinear function. Iterative use of MME with nonlinear assumed models cannot employ multiple shooting. In the multiple shooting approach of Mook and Lew [19], Equations (3.6-3.12) are rewritten as a new system of equations defined as

$$
\dot{y}(t) - C y(t) = F(t), \quad t_0 \leq t \leq t_f
$$

(3.14)
where the state and co-state are chosen as the dependent variables i.e.

$$\mathbf{y}(t) = \begin{pmatrix} \dot{\mathbf{x}}(t) \\ \lambda(t) \end{pmatrix}$$  \hspace{1cm} (3.15)

from Equations (3.6 - 3.8),

$$C = \begin{pmatrix} A_l & -\frac{1}{2}W^{-1} \\ 0 & -A^T_l \end{pmatrix}$$  \hspace{1cm} (3.16)

where $C$ is the $2n \times 2n$ state matrix of the newly defined system in Equation (3.14). The system forcing is accounted in

$$\mathbf{F}(t) = \begin{pmatrix} B_\mathbf{u}(t) \\ 0 \end{pmatrix}$$  \hspace{1cm} (3.17)

where $\mathbf{F}(t)$ is a $2n \times 1$ vector of known forcing functions.

The following equation accounts for the discontinuities at the measurement times,

$$y_{j+1}(t^+_j) = D_j y_j(t^-_j) + Y_j, \quad t_0 \leq t \leq t_f, j = 0, 1, 2, 3, ..., k$$  \hspace{1cm} (3.18)

and from Equation (3.9),

$$D_j = \begin{pmatrix} I & 0 \\ -2R_j^{-1} & I \end{pmatrix}$$  \hspace{1cm} (3.19)

$$Y_j = \begin{pmatrix} 0 \\ 2R_j^{-1} \bar{y}_j \end{pmatrix}$$  \hspace{1cm} (3.20)

where $D_j$ is a $2n \times 2n$ matrix and $Y_j$ is a $2n \times 1$ vector, both accounting for the jumps in the dependent variables. $I$ is an $n \times n$ identity matrix.
The following equation,

\[ \beta = B_0 y(t_0^-) + B_0 y(t_f^+) \]  \hspace{1cm} (3.21)

determines the boundary conditions. For the MME TPBVP if \( x(t_0) \) and \( x(t_f) \) are specified then,

\[
B_0 = \begin{pmatrix}
I & 0 \\
0 & I \\
0 & 0
\end{pmatrix}
\]  \hspace{1cm} (3.22)

\[
B_f = \begin{pmatrix}
0 & 0 \\
0 & I \\
I & 0
\end{pmatrix}
\]  \hspace{1cm} (3.23)

\[
\beta = \begin{pmatrix}
x(t_0) \\
x(t_f)
\end{pmatrix}
\]  \hspace{1cm} (3.24)

else if \( \delta(t_0) = 0 \) and \( \delta(t_f) = 0 \) are specified then,

\[
B_0 = \begin{pmatrix}
0 & I \\
0 & 0
\end{pmatrix}
\]  \hspace{1cm} (3.25)

\[
B_f = \begin{pmatrix}
0 & 0 \\
0 & I
\end{pmatrix}
\]  \hspace{1cm} (3.26)

\[
\beta = \begin{pmatrix}
0 \\
0
\end{pmatrix}
\]  \hspace{1cm} (3.27)

Typically, the initial or final values of only a few states are known, therefore \( \delta(t_0) \) and \( \delta(t_f) \) are specified.
The quantities in Equations (3.14), (3.15) and (3.21) are determined from the MME TP-BVP necessary conditions and are particular for every problem at hand. Equations (3.14), (3.15) and (3.21) are assembled into a set of matrices and solved at every time interval using linear algebra. The state and co-state trajectories are found at once from simple matrix inversions and multiplication as shown later in this section.

The solution of Equations (3.14), (3.15) and (3.21) is found by dividing the time domain into subintervals,

\[ t_0 < t_1 < t_2 < \ldots t_{j-1} < t_j = t_f = t_k \]  

(3.28)

where each node \( t_j \) coincides with a measurement time. An Initial Value Problem (IVP) is then solved in each time interval. The form of the solution \( y_j(t) \) in each time sub-interval is expressed as

\[ y_j(t) = v_j(t) + V_j(t)c_j, \quad t_{j-1} \leq t \leq t_j \]  

(3.29)

where \( v_j(t) \) is a particular solution vector and \( V_j(t) \) is a \( n \times n \) fundamental solution matrix and \( c(t) \) is a \( n \times 1 \) vector of to-be-determined constants. At the beginning of each sub-interval, \( t_{j-1}^+ \), the particular and fundamental solution are taken to be

\[ v_j(t_{j-1}^+) = v_j^0 = 0, \quad V_j(t_{j-1}^+) = V_j^0 = I \]  

(3.30)

Hence the solution at time \( t_{j-1}^+ \) is given as

\[ y(t_{j-1}^+) = v_j(t_{j-1}^+) + V_j(t_{j-1}^+)c_j = v_j^0 + V_j^0c_j \]  

(3.31)

The particular solution and the fundamental solution at the end of each interval is obtained by integrating Equation (3.32) over the time step.

\[ \dot{x}(t) - Cx(t) = F(t), \quad \dot{V}(t) - CV(t) = 0, \]  

(3.32)
Integrating Equation (3.32) with the initial conditions as specified in Equation (3.30) the particular and fundamental solution at the end of each time interval (namely \( v_j(t^-_j) \) and \( V_j(t^-_j) \)) is obtained. The continuity conditions at each time step must be modified to account for jump discontinuities. Consider the time being considered here as \( t_j \). At \( t^-_j \) the solution is represented as

\[
y(t^-_j) = v_j(t^-_j) + V_j(t^-_j) = v_j(t^-_j) + V_j(t^-_j)\xi_j
\] (3.33)

Substituting Equation (3.33) in Equation (3.18) yields

\[
y(t^+_j) = D_j(v_j(t^-_j) + V_j(t^-_j)\xi_j) + Y_j
\] (3.34)

The solution at time step \( t^+_j \) can also be represented in terms of the initial condition at the next time step as (similar to Equation (3.31))

\[
y(t^+_j) = y_{j+1}(t^+_j) = v^0_{j+1} + V^0_{j+1}\xi_{j+1}
\] (3.35)

Comparing Equations (3.34 and 3.35) it requires that

\[
D_j(v_j(t^-_j) + V_j(t^-_j)\xi_j) + Y_j = v^0_{j+1} + V^0_{j+1}\xi_{j+1}
\] (3.36)

The condition Equation (3.36) is imposed at each jump point.

We now evaluate the boundary condition, Equation (3.21). At the initial time, assuming a jump exists at \( t_0 \), we have

\[
y(t^-_0) = v^0_0 + V^0_0c_0
\] (3.37)

\[
y(t^+_0) = D_0(v^0_0 + V^0_0c_0) + Y_0
\] (3.38)
From Equations (3.35 and 3.36) we also have

\[ y(t_0^+) = v_1^0 + V_1^0 c_1 \]  

(3.39)

and

\[ D_0(v_0(t_0^-) + V_0(t_0^-) c_0) + Y_0 = v_1^0 + V_1^0 c_1 \]  

(3.40)

At the final time

\[ y(t_f^+) = D_k(v_k(t_f^-) + V_k(t_f^-) c_k) + Y_k \]  

(3.41)

Substituting Equations (3.37 and 3.41) into Equation (3.21), the boundary conditions may be written as

\[ \beta = B_0[v_0(t_0^-) + V_0(t_0^-) c_0] + B_f[D_k[v_k(t_f^-) + V_k(t_f^-) c_k] + Y_k] \]  

(3.42)

Combining Equations (3.34-3.42) we obtain

\[
\begin{pmatrix}
B_0 V_0(t_0^-) & 0 & 0 & \ldots & 0 & B_f D_k V_k(t_f^-) \\
-D_0 V_0(t_0^-) & V_1(t_0^-) & 0 & \ldots & 0 & 0 \\
0 & -D_1 V_1(t_f^-) & V_2(t_f^-) & \ldots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \ldots & V_{k-1}(t_{k-2}^+) & 0 \\
0 & 0 & 0 & \ldots & -D_{k-1} V_{k-1}(t_{k-1}^-) & V_k(t_{k-1}^+) \\
\end{pmatrix}
\begin{pmatrix}
c_0 \\
c_1 \\
c_2 \\
\vdots \\
\vdots \\
c_{k-1} \\
c_k \\
\end{pmatrix}
\]
\[
\begin{pmatrix}
\beta - B_0 \psi_0(t_0^-) - B_f D_k \psi_k(t_f^-) - B_f Y_k \\
D_0 \psi_0(t_0^-) - \psi_1(t_0^+) + Y_0 \\
D_1 \psi_1(t_1^-) - \psi_2(t_1^+) + Y_1 \\
\cdots \\
\cdots \\
\cdots \\
D_{k-2} \psi_{k-2}(t_{k-2}^-) - \psi_{k-1}(t_{k-2}^+) + Y_{k-2} \\
D_{k-1} \psi_{k-1}(t_{k-1}^-) - \psi_k(t_{k-1}^+) + Y_{k-1}
\end{pmatrix}
\]  

(3.43)

Equation (3.43) represents a system of \(2n \times (k + 1)\) equations for the \(k + 1\) unknown \(2n\) vectors \(c_0, c_1, c_2, \ldots, c_k\). In simpler form Equation (3.43) can be rewritten as

\[
\begin{pmatrix}
G_0(c) \\
G_1(c) \\
\cdots \\
G_{k-2}(c) \\
G_{k-1}(c)
\end{pmatrix} = \begin{pmatrix}
S_0 \\
S_1 \\
\cdots \\
S_{k-2} \\
S_{k-1}
\end{pmatrix}
\]

(3.44)

hence

\[
(c) = (\text{inv}[G])(S)
\]

(3.45)

The vector \(c\) contains the value of the states and the co-states at each time.

Among the multiple shooting advantages are to facilitate the solution of the TPBVP by eliminating the need to know the states boundary conditions, and to increase numerical stability. Implementing multiple shooting also causes a decrease in computer time since the old way of guessing at the boundary conditions to solve the MME TPBVP is eliminated. Multiple shooting, however, restricts the flexibility of MME by limiting the assumed model to a linear system, since the solution in multiple shooting is by means of linear algebra.
Chapter 4

Correlation and Coefficient Identification

4.1 Least Squares Algorithm

The final step in the identification procedure is to use a least-squares algorithm to fit the model error to the functional forms (i.e. perform parameter identification once the true non-linear form has been identified). The error term is expanded into a combination of the functional forms such as

\[ J = \sum_{j=1}^{J_A} g_j p_j |C|^{\theta_j} \]

where \( J_A \), \( K \), \( |C|^{\theta_j} \) are unknown coefficients to be determined by least squares, and \( g_j p_j |C|^{\theta_j} \) are functions which are selected as a result of the correlation test (often, however, only one function is used at a time). Other parameters may be present inside the functions (such as, for example, the coefficients of exponents). Equation 4.1 may be sampled repeatedly.
(using the MME estimates) to obtain

\[ d(t_1) = \alpha f_1(x(t_1)) + \beta f_2(x(t_1)) + \gamma f_3(x(t_1)) + \ldots \]
\[ d(t_2) = \alpha f_1(x(t_2)) + \beta f_2(x(t_2)) + \gamma f_3(x(t_2)) + \ldots \]
\[ d(t_3) = \alpha f_1(x(t_3)) + \beta f_2(x(t_3)) + \gamma f_3(x(t_3)) + \ldots \]
\[ \vdots \]
\[ d(t_l) = \alpha f_1(x(t_l)) + \beta f_2(x(t_l)) + \gamma f_3(x(t_l)) + \ldots \]

(4.2)

or, in the matrix form,

\[ \mathbf{D}_{l \times 1} = M_{l \times p} \mathbf{P}_{p \times 1} \]

(4.3)

where \( \mathbf{D}_{l \times 1} \) is a vector containing \( l \) number of \( d(t) \) samples used for the fit. The matrix \( M_{l \times p} \) contains the \( l \) tested functions evaluated \( l \) times and \( \mathbf{P} = [\alpha \ \beta \ \gamma \ldots]^T \) is the vector of the \( p \) coefficients for each of the tested functions.

The number of tested functions, \( p \), is chosen at the discretion of the user. The number of times the error term is fitted, \( l \), can be up to the data for the last time step \( (t_f) \) available since \( d(t_i) \) is available for all \( t_0 \leq t \leq t_f \). If the number of measurements is sufficiently large, however, \( l \) is chosen as the center eighty percent of the number of measurements, since it is found that the MME converges in this region [24].

Generally, because of the potential jump discontinuities in the model error estimates at the measurement times, it is desirable to pick the least squares sampling times in Equation 4.2 at points other than the measurement times. The least squares estimate is found by
minimizing the following cost functional with respect to $P$:

$$
\Phi = \{D - MP\}^T(D - MP)
$$

The solution is given by

$$
P = (M^T M)^{-1}M^T \hat{D} \tag{4.5}
$$

Among the benefits of this approach is the flexibility in selecting any kind of function to model the dynamic error term. Also, if a function which misrepresents the error term is included in the least-squares, the fit yields a large cost and the algorithm automatically indicates that the function choice is poor. In some instances, however, the dynamic error functional form is unknown and one may find oneself repeatedly guessing the unknown function(s) or using a large number of functions, since any curve may be accurately represented if a sufficient number of analytical terms are employed. This detracts from the algorithms efficiency and can yield models which are too complicated for engineering applications.

### 4.2 Correlation

Correlation is a measure of the relationship that exists between two variables. The more highly correlated two variables are, the more closely will the change in one variable correspond to a change in the other variable. The cross-correlation coefficient between two discrete variables, say $x$ and $y$, is defined as (Newland [18] or Witte [20])

$$
C(x, y) = \frac{\sum_{j=1}^{n}(x_j - \bar{x})(y_j - \bar{y})}{\sigma_x \sigma_y n} \tag{4.6}
$$
where \( n \) is the number of data points and the overbar denotes the mean of those \( n \) points, \( \sigma_x \) is the standard deviation of the variable \( x \) and is defined as
\[
\sigma_x = \sqrt{\frac{\sum_{j=1}^{n}(x_j - \bar{x})}{n}} \tag{4.7}
\]
The cross correlation \( C(x, y) \) is a measure of the linear relationship between variables \( x \) and \( y \). The value of \( C(x, y) \) lies in the range \(-1 < C(x, y) < 1\). If, for instance, changes in the value of \( y \), where the changes in both variables are of the same sign, then the value of \( C(x, y) \) is -1. If changes in the values of \( x \) and \( y \) tend to correspond in sign but are not perfectly predictable, then \( 0 < C(x, y) < 1 \). If changes in the values of \( x \) and \( y \) tend to be opposite sign but are not perfectly predictable, then \(-1 < C(x, y) < 0 \). If there is no linear relationship between the values of \( x \) and \( y \), then \( C(x, y) = 0 \). For example, suppose \( x \) and \( y \) are multiples of each other, \( x = K*y \), where \( K \) is an arbitrary constant of proportionality. Then
\[
C(x, y) = \frac{\sum_{i=1}^{n} K(x_i - \bar{x})^2}{\sum_{j=1}^{n} K(x_j - \bar{x})^2} = 1.0 \tag{4.8}
\]
The true functional form of the model error can be found by calculating the correlation of the MME model error estimates with functions of the MME state estimates. If the functional form of the actual system is used, and if the estimates from the MME are perfect, then \( C(x, y) = 1.0 \). Thus, an algorithm is constructed which performs nonlinear system identification by:

- utilizing the MME to process the available measurement and the initial model in order to produce state estimates and the model error estimates, and
- testing the correlation between the state estimates and the model error estimates using
a “sufficient number” of functional forms so that the actual form is included among
those tested.

The MME method does not require that the correct form of the model be known a-priori.
The correlation tests may be performed using an existing library of nonlinear functional
forms, without input from the user. Thus, if the library is complete (in the sense that it
contains the actual model form), the identification of the nonlinear model is accomplished,
yet at no point in the algorithm is the user required to assume the correct model form.

The success of the algorithm is determined by the ability of the MME to produce ac-
curate state and model error estimates, and by the completeness of the library of nonlinear
functions to be used in the correlation test.

The MME method has been shown to consistently produce state and model error es-
timates of high accuracy in the presence of high measurement noise, low measurement
frequency, and poor initial model [[3] -[5]]. Generally, however, some noise is still present
in both the state estimate and the model error term, although these noise levels are con-
siderably less than the noise in the original data. Let the model error term be given by

\[ x_{\text{correlation}} = x + \xi \]  

where \( \xi \) is the noise. The cross-correlation between the error term and
the test function \( y \) becomes

\[
C(x, y) = \frac{\sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) + \sum_{j=1}^{n} \xi(y_i - \bar{y})}{\sigma_x \sigma_y + \sum_{k=1}^{n} (2\xi(x_i - \bar{x}) + \xi^2)} = 1.0
\]

(4.9)

As long as the noise is negligible, all terms containing \( x_i \) are small and affect the result
only slightly. Thus, the correlation calculated for the actual function is close to, but not
exactly equal to, 1, while the correlation calculated for incorrect terms remains close to 0.
If the level of noise is excessive, say, of comparable magnitude to one or more of the actual
nonlinear model terms, then the ability of the correlation test to distinguish this term from similar terms may be greatly reduced or eliminated. However, even with significant noise, least-squares fit of the terms can correctly select the actual nonlinear function from among those which the correlation test could not distinguish [6].

If the highest cross-correlation absolute value is much less than one, it indicates that more than one function is necessary to model the dynamic error term. Say the dynamic error term is composed of two functions, $x_i = x_{1i} + x_{2i}$, and then cross-correlation has the form

$$C(x, y) = \frac{\sum_{i=1}^{n} (x_{1i} - \bar{x}_{1})(y_i - \bar{y})}{n\sigma_y\sqrt{\sigma_{x1}^2 + \sigma_{x2}^2 + \sum_{k=1}^{n}(x_{1k} - \bar{x}_{1})(x_{2k} - \bar{x}_{2})/n}} + \frac{\sum_{j=1}^{n} (x_{2j} - \bar{x}_{2})(y_j - \bar{y})}{n\sigma_y\sqrt{\sigma_{x1}^2 + \sigma_{x2}^2 + \sum_{k=1}^{n}(x_{1k} - \bar{x}_{1})(x_{2k} - \bar{x}_{2})/n}}$$

(4.10)

The cross correlation is highest for the term which constitutes the largest part of the error. Thus, it is desirable to execute the algorithm iteratively. The library term which constitutes the largest portion of the actual model error is identified first and then added to the MME model. The entire process (including MME) is then repeated, so that new state and model error estimates are obtained (note that the change in state estimates should be minimal, while the change in model error estimates should be a large reduction in magnitude). The largest term remaining in the model error is identified in each pass, then added to the initial MME model.

An alternative to iterative application of the algorithm is to test the correlation of combinations of the library functions. An algorithm can be constructed which tests every possible combination of the functions explicitly contained in the library.
If the actual model error is not present in the library then test cases [6] show that the highest correlation values are calculated for the terms in the series expansion of the actual function. Thus, for example, if the actual model error was of the form \( \sin(x) \), but \( \sin(x) \) was not present in the library, the correlation coefficients are highest for the terms \( x, x^3, x^5 \ldots \) etc.

Summarizing the identification procedure described in Chapter 3 and 4: to identify nonlinear systems, MME requires a number of discrete time-domain measurements and an assumed model. The assumed model contains what is already known of the system. A cost functional is minimized such that the estimated states match the measurements with the same error covariance as the measurements match the truth. This method produces optimal state trajectories of all states and a vector of terms containing the dynamic model error. The model error is correlated with an extensive library of linear and nonlinear functions. The cross-correlation coefficient points out the functions which best represent the model error. A least-squares algorithm is utilized to identify the parameter for the selected functions. The assumed model plus the determined analytical terms yield an accurate model of the nonlinear system.
Chapter 5

Conservative Single-Degree-of-Freedom Nonlinear Systems

In this chapter the ability of MME to identify the form of a conservative nonlinear system having one degree of freedom is described. The MME method for identifying the nonlinearity of the free (undamped and unforced) oscillations of a spring mass system is presented in detail with and without noise in the signal. The ability of MME to identify the model form even when only the acceleration data is available of the system, is demonstrated with and without noise. The Runge-Kutta method is used to simulate the actual response of the system and this data is used in the MME algorithm to determine the unmodeled nonlinearity and then the least squares method is used to identify the coefficient of the nonlinear term.

5.1 Single Degree of Freedom System

Consider the free response of a Duffing oscillator as shown in Figure 5.1. For the purpose of computational simplicity let us assume that the mass and stiffness of the system are such
that the equation of motion for the system is:

$$\ddot{x} + x - \alpha(x^3) = 0$$  \hspace{1cm} (5.1)

The true system can be modeled in state space form as:

$$\begin{pmatrix}
\dot{x} \\
\dot{\dot{x}}
\end{pmatrix} =
\begin{bmatrix}
0 & 1 \\
-1 & 0
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix}
+ 
\begin{pmatrix}
0 \\
\alpha(x^3)
\end{pmatrix}
$$

where \(x\) is the position, \(\dot{x}\) is velocity and (\(\cdot\)) indicates differentiation with respect to time.

The term \(\alpha(x^3)\) denotes the nonlinear term to be identified by the MME-based identification algorithm. Measurements were simulated by solving Equation (5.1) using a Runge Kutta method. The data obtained from this simulation was used as the input to the MME algorithm, instead of an experimental data. The program written to do this simulation and to identify the nonlinearity, is shown in Appendix B.

The assumed model used for the MME analysis consists of the undamped linear oscillator part of the system,

$$\begin{pmatrix}
\dot{x} \\
\dot{\dot{x}}
\end{pmatrix} =
\begin{bmatrix}
0 & 1 \\
-1 & 0
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix}$$
which may be represented as,

\[ \dot{X} = AX \]  \hspace{1cm} (5.2)

where

\[ X = \begin{pmatrix} x \\ \dot{x} \end{pmatrix} \] \hspace{1cm} (5.3)

and

\[ \dot{X} = \begin{pmatrix} \dot{x} \\ \ddot{x} \end{pmatrix} \] \hspace{1cm} (5.4)

5.1.1 Simulation

The noiseless measurements are simulated by integrating Equation (5.1) using the fourth order Runge-Kutta method for a period of 10 seconds with a step size of 0.1 seconds (equivalent to a 10 Hz sampling frequency). Data was generated for values of \( \alpha \) as 0.3 and 0.5. For the sake of easy reference the following nomenclature is used for identifying the measurement in terms of the nonlinearity function present in it. SET 1 denotes the measurement simulated with the nonlinear coefficient as 0.3, with initial conditions of \( x_0 = [5.0 \quad 0]^T \). For 10% noise level the initial conditions are \( x_0 = [4.0 \quad 0]^T \). SET 2 denotes the measurement simulated with the nonlinear coefficient as 0.5, with initial conditions of \( x_0 = [2.0 \quad 0]^T \).
Let $\mathbf{X}$ denote the response vector.

$$
\mathbf{X} = \begin{pmatrix}
  x_1 \\
  x_2 \\
  \vdots \\
  x_n
\end{pmatrix}
$$

where $x_1$ denotes the response at time $t_1$, $x_2$ at $t_2$ and so on.

The next step is to calculate the acceleration using Equation (5.1):

$$
\ddot{\mathbf{X}} = \begin{pmatrix}
  \ddot{x}_1 \\
  \ddot{x}_2 \\
  \vdots \\
  \ddot{x}_n
\end{pmatrix}
$$

where

$$
\ddot{x}_n = -x_n - \alpha(x_n^3)
$$

(5.5)

Now we have a nonlinear system for which the position, velocity and acceleration vectors are available. In the practical world a pure signal without noise is unheard of. In order to simulate a real world problem, noise of three varying magnitudes were added to the signal. The noise levels considered were 1%, 5% and 10%. The noise was generated using the `rand` program for generation of random numbers in Matlab and this was added to the measurement signal. So now measurements are available for the system for two different values of the nonlinear coefficient $\alpha$ with three different noise levels resulting in 6 sets of ‘measurement’ data. Further 2 sets of data are available in the form of noiseless measurements. Hence a total of 8 sets of data are available for the MME analysis.
To validate the MME procedure, at first the position and velocity vectors are taken as the known measured output from the system and these are used as the input into the MME identification algorithm. The nonlinearity in the system is first identified and then, using the correlation procedure, the coefficient of the non-linear term is identified. The position, velocity and the acceleration signals were input into the identification procedure with various levels of noise level and the robustness of the identification procedure with respect to various level of noise disturbance in the signal was evaluated.

In the next example it is assumed that only the acceleration data is available for the system. The acceleration data with various levels of noise in the signal was input to the MME identification algorithm. The first step was to identify the unknown states, namely, the position and the velocity vectors and the model error term. The MME algorithm evaluates the states and the model error simultaneously. Once the model error and the states have been estimated to a considerable degree of accuracy, the correlation procedure is used to identify the non-linear function form and the least squares fit is used to identify the coefficient of the non-linear function.

### 5.2 Position and Velocity Measurement

#### 5.2.1 Noiseless Measurements

Noiseless measurement was first used as the input to the MME algorithm. Both the position and velocity are assumed to be available measurements and this is used to identify the nonlinearity in the system.
The measurement model is given by:

\[
\tilde{y}(t_k) = [x(t_k); \dot{x}(t_k)]^T
\]  

(5.6)

and the assumed model is written as

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} =
\begin{bmatrix}
0 & 1 \\
-1 & 0
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix} +
\begin{pmatrix}
d_1 \\
d_2
\end{pmatrix}
\]

where \(d_1\) and \(d_2\) are the model error terms that are to be identified by the MME algorithm.

The velocity and position vectors are input into the algorithm and then convergence of the program is achieved when the estimated states match the input states within the covariance constraint assumed in the beginning, which in this case is 0.0025 (a random guess). The weighting function for which the states converged is \(W = [400 \quad 0.01]^T\) for the nonlinear coefficient of 0.3 and \(W = [5 \quad 0.001]^T\) for the nonlinear coefficient of 0.5.
Figure 5.2: True Position (solid) and MME Estimated (+) without noise

Figure 5.3: True Velocity (solid) and MME Estimated (+) without noise
Figure 5.4: True Acceleration (solid) and MME Estimated (+) without noise

The optimal states obtained by solving the two-point boundary value problem are shown.

Figure 5.5: True Model Error $d_2$ (solid) and MME Estimated (+) without noise
in Figures 5.2-5.3. The first equation states that \( \dot{x}(t) = \dot{x}(t) + d_1(t) \). This can be true only if \( d_1(t) = 0 \). The values for the vector obtained are \( d_1(t) \approx 0 \). Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to be zero. Figure 5.5 shows the actual model error, \( d_2 \), compared to the estimated model error. This model error term is later used to identify the nonlinear function and identify its coefficient. The two are clearly in close agreement. Let the estimated position vector be \( \hat{x} \) and the estimated velocity vector be \( \hat{x} \). Figures 5.2-5.5 show the comparison of the MME estimated states and model error with the true states and true model error.

Once the states and the model error have been identified the next step is to identify the nonlinear function which closely fits the model error. In order to do so a library of 3 nonlinear functions were used in the correlation algorithm. The Table 5.1 shows the nonlinear functions used and the cross-correlation coefficient, \( C(d(t), f) \), obtained for the functions after the first iteration.

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t), f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( d_2 = -0.3x^3 )</td>
<td>0</td>
<td>( (\dot{x})(\dot{x}) )</td>
<td>-0.0434</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>( \ddot{x} )</td>
<td>-0.1364</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>( \ddot{x} )</td>
<td>-0.9469</td>
<td>( \checkmark )</td>
</tr>
<tr>
<td>2</td>
<td>( d_2 = -0.5x^3 )</td>
<td>0</td>
<td>( (\dot{x})(\dot{x}) )</td>
<td>-0.00034</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>( \ddot{x} )</td>
<td>-0.1025</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>( \ddot{x} )</td>
<td>-0.9762</td>
<td>( \checkmark )</td>
</tr>
</tbody>
</table>

Table 5.1: CORRELATION COEFFICIENTS FOR DATA WITH NO NOISE
The functional form of the nonlinearity was determined solely from the least-squares fit of the functions identified during the correlation tests on the MME state and model error estimates obtained using only the linear model. From the Table 5.1 it is clear that the correlation coefficient is highest for the function \( \hat{x}^3 \) and hence this function is chosen as the first non-linear function for the system. The new equation for the system becomes:

\[
\begin{pmatrix}
\dot{x} \\
\dot{x}
\end{pmatrix} = \begin{bmatrix}
0 & 1 \\
-1 & 0
\end{bmatrix} \begin{pmatrix}
x \\
\dot{x}
\end{pmatrix} + \begin{pmatrix}
0 \\
\alpha(x^3)
\end{pmatrix}
\]

(5.7)

**Least Squares Fit**

The next step is to identify \( \alpha \), the coefficient of the nonlinear function. The measurement Set 2 is considered first to describe the whole procedure in detail. This requires the model error term \( d_1 \) and \( d_2 \), position vector \( \hat{x} \) and the velocity vector \( \dot{x} \).

The model error term can now be written in the form of the function form identified as

\[
d_2(t_i) = \alpha \hat{x}_i^3 \quad i = 1, 2, 3, ..., l
\]

(5.8)

where \( i \) denotes each time instant.

The model error data over the entire time domain can be written in the form of linear
equations as:

\[ d_2(t_1) = \alpha f_1(\hat{x}_3(t_1)) \]
\[ d_2(t_2) = \alpha f_1(\hat{x}_3(t_2)) \]
\[ d_2(t_3) = \alpha f_1(\hat{x}_3(t_3)) \]
\[ \vdots = \vdots \]
\[ d_2(t_l) = \alpha f_1(\hat{x}_3(t_l)) \]  

(5.9)

or, in the matrix form,

\[
\begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_l \end{bmatrix}_{1 \times l} = M_{l \times 1} \begin{bmatrix} P_{1 \times 1} \end{bmatrix}
\]  

(5.10)

where \( D_{1 \times l} \) contains all of the \( l \) number of model error terms \( d_2(t_i) \) used for the fit. The matrix \( M_{l \times 1} \) contains the function \( \hat{x}^3 \) evaluated at each time step and \( P_{1 \times 1} = [\alpha] \) is the coefficient for the tested function \( \hat{x}^3 \). In all the analysis done so far \([24]\) the best convergence was obtained using the mid 80% data points. Hence it is recommended that the only the data in the mid 80% be taken into consideration for the correlation and least squares fit.

From Equation 4.5

\[ \alpha = (M^T M)^{-1} M^T D \]  

(5.11)

The value of \( \alpha \) identified is 0.4946 as compared to the true value of 0.5. The system model is now updated to include this nonlinear function. The updates model of the system becomes

\[
\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ -0.4946(x^3) \end{bmatrix}
\]  

(5.12)
The next step is to update the model error term by removing the contribution of the nonlinear function just identified. The new model error term becomes

\[
\begin{pmatrix}
\dot{d}_2(t) \\
\ddot{d}_2(t)
\end{pmatrix} =
\begin{pmatrix}
d_2(t) \\
\ddot{d}_2(t)
\end{pmatrix} - 
\begin{pmatrix}
-0.4946(x^3(t))
\end{pmatrix}
\tag{5.13}
\]

The updated model error vector is \( \dot{d}_2(t) \). This error term was found to be very close to zero and hence a further correlation identification was not performed. Even if the correlation identification was performed on this error term it would return very low correlation coefficient for any function form used. This goes to show that the model error term has been almost completely accounted for by the \(-0.4946x^3\) term.

The corresponding value for the first set of data was also obtained and found to be \(-0.2908x^3\) as compared to 0.3. Further accuracy to the coefficient of the nonlinear coefficient can be obtained by using this new model as the assumed model in the MME, instead of the linear model and a new set of model error terms can be identified and a correlation and least squares fit may be done on that data and a updated function and coefficient could be identified. The purpose of this example is to demonstrate the convergence of the MME algorithm and since the coefficient has converged to within 2% of the value, hence further iteration is not carried out.

**In summary:**

The system model at the beginning was an undamped linear oscillator of the form,

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{pmatrix} x \\ \dot{x} \end{pmatrix}
\]

The MME method identified the nonlinear function as \( x^3 \) from a library of three func-
tions using cross-correlation techniques and a least squares fit identified the coefficient of
the nonlinear function \( x^3 \) as \(-0.2908\) and \(-0.4946\) respectively for the two sets of data.

Hence the equation of the Nonlinear System for the first set would be

\[
\begin{pmatrix}
\dot{x} \\
\dot{\dot{x}}
\end{pmatrix}
= \begin{pmatrix}
0 & 1 \\
-1 & 0
\end{pmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix}
+ \begin{pmatrix}
0 \\
-0.2908x^3
\end{pmatrix}
\]

and for the second set it would be

\[
\begin{pmatrix}
\dot{x} \\
\dot{\dot{x}}
\end{pmatrix}
= \begin{pmatrix}
0 & 1 \\
-1 & 0
\end{pmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix}
+ \begin{pmatrix}
0 \\
-0.4946x^3
\end{pmatrix}
\]

Hence the equation of motion of the Nonlinear system as identified by MME are:

\[
\ddot{x} + x + 0.2908x^3 = 0 \tag{5.14}
\]

and

\[
\ddot{x} + x + 0.4946x^3 = 0 \tag{5.15}
\]

These equations are in agreement with Equation (5.1) within 3%. This result indicates
that the correlation technique applied to a MME-produced state and model error estimates
enables to identify the model form of the system.

5.2.2 Measurements with noise

Noise of various magnitude was added to noiseless measurements obtained earlier. The
noise levels considered were 1\%, 5\% and 10\%.

The measurement model is given by:

\[
\bar{y}(t_k) = [\bar{x}(t_k); \bar{\dot{x}}(t_k)]^T \tag{5.16}
\]
and when noise is included it becomes

\[
\ddot{x}(t_k) = \dot{x}(t_k) + q_k, \quad \ddot{\dot{x}}(t_k) = \dot{\dot{x}}(t_k) + \ddot{q}_k
\]  

(5.17)

Here \( q_k \) represents measurement noise. It is assumed to be a zero-mean gaussian distribution process of known covariance \( R_k \).

Since the entire process of MME algorithm has been explained in detail in the earlier example, the details have not been explained here. The following plots and tables show the comparison between the assumed model and the true system. The tables showing the cross-correlations and the function coefficients are also attached.

**Measurements with 1\% noise**

The convergence criteria here is taken as \( R_k = 0.0001 \) (\( R_k \)=covariance). Figures 5.6-5.9 show the comparison of the MME estimated states and model error with the true states and true model error. It was found that the values for the vector \( d_1(t) \approx 0 \) as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to be zero. Figure 5.9 shows the actual model error, \( d_2 \), compared to the estimated model error. This model error term would be later on used to identify the nonlinear function and identify the coefficient. The two are clearly in close agreement. The Table 5.2 shows the nonlinear functions used and the cross-correlation coefficient, \( C(d(t), f) \), obtained for the functions after the first iteration.

From the Table 5.2 it is clear that the correlation coefficient is highest for the function \( x^3 \) and hence this function is chosen as the first non-linear function for the system. The
new equation for the system becomes:

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} = \begin{bmatrix}
0 & 1 \\
-1 & 0
\end{bmatrix} \begin{pmatrix}
x \\
\dot{x}
\end{pmatrix} + \begin{pmatrix}
0 \\
\alpha(x^3)
\end{pmatrix}
\]

(5.18)

As before the model error data over the entire time domain can be written in the form of linear equations as:

\[
d_2(t_1) = \alpha f_1(\ddot{x}_3(t_1))
\]

\[
d_2(t_2) = \alpha f_1(\ddot{x}_3(t_2))
\]

\[
d_2(t_3) = \alpha f_1(\ddot{x}_3(t_3))
\]

\[
\vdots = \vdots
\]

\[
d_2(t_i) = \alpha f_1(\ddot{x}_3(t_i))
\]

(5.19)
or, in the matrix form,

\[ D_{t \times 1} = M_{t \times 1} P_{1 \times 1} \] (5.20)

The least squares fit was performed on this data and the value of \( \alpha \) identified are \( \alpha = -0.2937 \) (for Set 1) and \( \alpha = -0.4984 \) (for Set 2). Hence it shows that the MME is able to identify the nonlinearity function within 1.5\% even in the presence of 1\% noise in the signal.
Figure 5.6: True Position (solid) and MME Estimated (+) with 1% noise

Figure 5.7: True Velocity (solid) and MME Estimated (+) with 1% noise
The convergence criteria here is taken as $R_k = 0.0025$. Figures 5.10-5.13 show the comparison of the MME estimated states and model error with the true states and true model.
error. It was noted that the values for the vector \( \mathbf{d}_1(t) \approx 0 \) as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to be zero. Figure 5.13 shows the actual model error, \( \hat{d}_2 \), compared to the estimated model error. This model error term would be later on used to identify the nonlinear function and identify the coefficient. The two are clearly in close agreement. Let the estimated position vector be \( \hat{x} \) and the estimated velocity vector be \( \hat{\dot{x}} \). The Table 5.3 shows the non-linear functions used and the cross-correlation coefficient, \( C(d(t),f) \), obtained for the functions after the first iteration. From the Table 5.3 it is clear that the correlation coefficient is highest for the function \( \hat{x}^3 \) and hence this function is chosen as the first non-linear function for the system.

The least squares fit was performed on this data value of \( \alpha \) identified from the program are \( \alpha = -0.2963 \) (for Set 1) and \( \alpha = -0.4967 \) (for Set 2). Hence it shows that the MME is able to identify the nonlinearity function within a reasonable degree of accuracy even in

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( d_2 = -0.3x^3 )</td>
<td>5%</td>
<td>( \ddot{x} )</td>
<td>0.0478</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>( \frac{\ddot{x}^3}{\dot{x}} )</td>
<td>-0.1276</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>( \frac{\ddot{x}^3}{\dot{x}} )</td>
<td>-0.9586</td>
<td>√</td>
</tr>
<tr>
<td>2</td>
<td>( d_2 = -0.5x^3 )</td>
<td>5%</td>
<td>( \ddot{x} )</td>
<td>0.0244</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>( \frac{\ddot{x}^3}{\dot{x}} )</td>
<td>-0.1513</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>( \frac{\ddot{x}^3}{\dot{x}} )</td>
<td>-0.9521</td>
<td>√</td>
</tr>
</tbody>
</table>

Table 5.3: CORRELATION COEFFICIENTS FOR DATA WITH 5% NOISE
the presence of 5% noise in the signal.
Figure 5.10: True Position (solid) and MME Estimated (+) with 5% noise

Figure 5.11: True Velocity (solid) and MME Estimated (+) with 5% noise
Figure 5.12: True Acceleration (solid) and MME Estimated (+) with 5% noise

Measurements with 10% noise

The convergence criteria here is taken as $R_k = 0.01$. The value of $R_k$ is chosen randomly and depends on how close the MME data has to converge to the true data. Figures 5.14-5.17

Figure 5.13: True Model Error $d_2$ (solid) and MME Estimated (+) with 5% noise
show the comparison of the MME estimated states and model error with the true states and true model error. It was noted that the values for the vector $d_n(t) \approx 0$ as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Figure 5.17 shows the actual model error, $d_o$, compared to the estimated model error. This model error term is used later to identify the non-linear function and identify the coefficient. The two are clearly in close agreement. Let the estimated position vector be $\hat{x}$ and the estimated velocity vector be $\hat{x}$. Table 5.4 shows the nonlinear functions used and the cross-correlation coefficient, $C(d(t),f)$, obtained for the functions after the first iteration.

From the Table 5.4 it is clear that the correlation coefficient is highest for the function $\hat{x}^3$ and hence this function is chosen as the first non-linear function for the system. The new equation for the system becomes:

The least squares fit was performed on this data value of $\alpha$ identified from the program are $\alpha = -0.2949$ (for Set 1) and $\alpha = -0.4725$ (for Set 2). Hence it shows that the MME
is able to identify the nonlinearity function within a reasonable degree of accuracy even in
the presence of 10\% noise in the signal.
Figure 5.14: True Position (solid) and MME Estimated (+) with 10% noise

Figure 5.15: True Velocity (solid) and MME Estimated (+) with 10% noise
Figure 5.16: True Acceleration (solid) and MME Estimated (+) with 10% noise

Figure 5.17: True Model Error $d_2$ (solid) and MME Estimated (+) with 10% noise
5.3 Acceleration Measurement

The measurement model is defined as:

\[
\bar{y}(t_k) = \bar{x} = [0 \ 1] [\ddot{x}(t_k) \ \dddot{x}(t_k)]^T \tag{5.21}
\]

Let \( \dddot{x} \) denote the response acceleration vector.

\[
\dddot{x} = \begin{pmatrix}
\dddot{x}_1 \\
\dddot{x}_2 \\
\vdots \\
\dddot{x}_n
\end{pmatrix}
\]

where \( \dddot{x}_1 \) denotes the response at time \( t_1 \), \( \dddot{x}_2 \) at \( t_2 \) and so on. Consider this acceleration data to be the only available measurement. So unlike the previous example there is no position and velocity data available to check the convergence criteria. Hence the MME procedure has to be modified suitably to check for the acceleration convergence rather than position and velocity.

5.3.1 Multiple Shooting Algorithm development

The TBVP with the jump discontinuities can be expressed in the form of linear equations of the form, as was the case earlier in Equation (3.14):

\[
\dot{y} - C(t)y = f(t), \quad t_0 \leq t \leq t_f, t \neq \tau_i, i = 0, 1, 2, 3, \ldots, M \tag{5.22}
\]

\[
\beta = B_0y(t^-_0) + B_0y(t^+_0) \tag{5.23}
\]

\[
y(\tau^{+}_i) = D_i y(\tau^{-}_i) + Y_i, \quad t_0 \leq t \leq t_f, t \neq \tau_i, i = 0, 1, 2, 3, \ldots, M \tag{5.24}
\]
where \( y(t) \) is the vector of dependent variables; \( B_0, B_f, \) and \( \beta \) account for the boundary conditions; and \( D_i \) and \( Y_i \) account for the jump discontinuities.

Converting the necessary conditions given in Equations (3.6 - 3.10) into the form of equations given above, the whole system of equations can be written in matrix form as:

\[
y(t) = \begin{pmatrix} \dot{x}(t) \\ \Lambda(t) \end{pmatrix}
\]

from Equations 3.6 - 3.8,

\[
C = \begin{pmatrix} A & -\frac{1}{2} W^{-1} \\ 0 & -A^T \end{pmatrix}
\]

where \( C \) is the \( 2n \times 2n \) state matrix of the newly defined system in Equation (5.22). The system forcing is accounted in

\[
F(t) = \begin{pmatrix} B y(t) \\ 0 \end{pmatrix}
\]

where \( F(t) \) is a \( 2n \times 1 \) vector of known forcing functions.

The following equation accounts for the discontinuities at the measurement times,

\[
y_{i+1}(t^+_i) = D_i y_i(t^-_i) + Y_i, \quad t_0 \leq t \leq t_f, i = 0, 1, 2, 3, \ldots, M
\]

To explain the convergence criteria in detail let us start from the basic state space equation.

\[
\begin{pmatrix} \dot{x} \\ \ddot{x} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \begin{pmatrix} \dot{x} \\ \ddot{x} \end{pmatrix} + \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}
\]

The first convergence criteria is that the velocity vector \( \dot{x} \) on the left hand side of the Equation (5.29) should be equal to the velocity vector \( \dot{x} \) obtained by MME which is on
the right hand side of the equation. This would ensure that \( d_1 \) tends to zero. This is done by minimizing \( R_k \) where

\[
R_k = \frac{1}{n} \sum_{i=1}^{n} (\dot{x} - \dot{x})^2
\]  

(5.30)

that is

\[
R_k = \frac{1}{n} \sum_{i=1}^{n} \left( \dot{x} - [1 \ 0] \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \right)^2
\]  

(5.31)

Let

\[
g_1 = [1 \ 0] \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}
\]  

(5.32)

or

\[
g_1 = \dot{x} + d_1
\]  

(5.33)

but from Equation (3.8),

\[
d_1 = \frac{-\lambda_1}{2W_1}
\]  

(5.34)

hence

\[
g_1 = \dot{x} - \frac{\lambda_1}{2W_1}
\]  

(5.35)

Rewriting the convergence criteria Equation (3.9) in terms of the Lagrange multipliers (\( \lambda \)), we have

\[
\lambda(t_k^+) = \lambda(t_k^-) + 2R_k^{-1}[\bar{y}(t_k) - g_1]
\]  

(5.36)
or

$$\Delta_1(t^+_k) = \Delta_1(t^-_k) + 2R_k^{-1} \left[ [\hat{x} - \frac{\Delta_1}{2W_2}] - \bar{y}(t_k) \right] \tag{5.37}$$

Here $\bar{y}(t_k)$ is the velocity vector itself. Hence

$$\Delta_1(t^+_k) = \Delta_1(t^-_k) - 2R_k^{-1} \left[ \frac{\Delta_1}{2W_2} \right] \tag{5.38}$$

Now consider the second convergence criteria: that the acceleration vector ($\ddot{x}$) on the left hand side of the Equation (5.29) should be equal to the acceleration vector ($\ddot{x}$) obtained by the MME on the right hand side of the equation. This would ensure that $d_2$ converges to the model error term.

$$R_k = \frac{1}{n} \sum_{i=1}^{n} (\ddot{x} - \ddot{x})^2 \tag{5.39}$$

that is

$$R_k = \frac{1}{n} \sum_{i=1}^{n} \left( \ddot{x} - [0 \quad 1] \left[ \begin{array}{cc} 0 & 1 \\ -1 & 0 \end{array} \right] \left( \begin{array}{c} \hat{x} \\ \ddot{x} \end{array} \right) + \left( \begin{array}{c} d_1 \\ d_2 \end{array} \right) \right) \right)^2 \tag{5.40}$$

Let

$$g_2 = [0 \quad 1] \left[ \begin{array}{cc} 0 & 1 \\ -1 & 0 \end{array} \right] \left( \begin{array}{c} \hat{x} \\ \ddot{x} \end{array} \right) + \left( \begin{array}{c} d_1 \\ d_2 \end{array} \right) \tag{5.41}$$

or

$$g_2 = -\ddot{x} + d_2 \tag{5.42}$$

but from Equation (3.8),

$$d_2 = \frac{-\lambda_2}{2W_2} \tag{5.43}$$
hence

\[ g_2 = -\hat{x} - \frac{\lambda_2}{2W_1} \]  

(5.44)

Rewriting the convergence criteria given by Equation (5.30) in terms of the Lagrange multipliers, we have

\[ \Lambda(t_k^+) = \Lambda(t_k^-) + 2R_k^{-1}[\overline{y}(t_k) - g_2] \]  

(5.45)

or

\[ \Lambda_0(t_k^+) = \Lambda_0(t_k^-) + 2R_k^{-1}[\overline{y}(t_k) - [-\hat{x} - \frac{\lambda_0}{2W_2}]] \]  

(5.46)

The constraint Equations (5.38 and 5.46) can be substituted in Equation (3.18) and the resulting equation can be written in matrix form as,

\[ y_{i+1}(t_i^+) = D_i y_i(t_i^-) + Y_i, \quad t_0 \leq t \leq t_f, i = 0, 1, 2, ..., M \]  

(5.47)

where

\[
\begin{pmatrix}
\hat{x}_{i+1}(t_i^+) \\
\hat{x}_{i+1}(t_i^+) \\
\lambda_{1(i+1)}(t_i^+) \\
\lambda_{2(i+1)}(t_i^+)
\end{pmatrix} =
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & (1 - R_k^{-1}\frac{1}{W_1}) & 0 \\
2R_k^{-1} & 0 & 0 & (1 + R_k^{-1}\frac{1}{W_2})
\end{pmatrix}
\begin{pmatrix}
\hat{x}_i(t_i^-) \\
\hat{x}_i(t_i^-) \\
\lambda_{1(i)}(t_i^-) \\
\lambda_{2(i)}(t_i^-)
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
0 \\
2R_k^{-1}\overline{y}_i
\end{pmatrix}
\]  

(5.48)
The boundary conditions remain the same as the earlier

\[ \beta = B_0 y(t_0^-) + B_0 y(t_f^+) \]  \hspace{1cm} (5.49)

\[ B_0 = \begin{pmatrix} 0 & I \\ 0 & 0 \end{pmatrix} \]  \hspace{1cm} (5.50)

\[ B_f = \begin{pmatrix} 0 & 0 \\ 0 & I \end{pmatrix} \]  \hspace{1cm} (5.51)

\[ \beta = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \]  \hspace{1cm} (5.52)

### 5.3.2 Noiseless Measurements

New set of noiseless measurement were generated and used as the input to the MME program. The acceleration data for the system is assumed to be available and this is used to identify the nonlinearity in the system. The initial conditions are same as for Set 1, \( x_0 = [2.0 \ 0]^T \) and the measurements are sampled at 5 Hz over a 30-second interval.

Again two sets of data were generated with two different values of \( \alpha \) namely 0.3 and 0.5. For the sake of easy reference the following nomenclature is used for identifying the measurement in terms of the nonlinearity function present in it. Set 1 denotes the measurement is simulated with the nonlinear coefficient as 0.3. Set 2 denotes the measurement is simulated with the nonlinear coefficient as 0.5.

The Table 5.5 shows the nonlinear functions used and the cross-correlation coefficient, \( C(d(t), f) \), obtained for the functions after the first iteration.
Table 5.5: CORRELATION COEFFICIENTS FOR DATA WITH NO NOISE

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(d_2 = -0.3x^3)</td>
<td>0</td>
<td>((\dot{x})(\dot{x}))</td>
<td>-0.1673</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>(\dot{x}^2)</td>
<td>-0.1289</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>(\dot{x}^3)</td>
<td>-0.9455</td>
<td>(\sqrt{\ })</td>
</tr>
<tr>
<td>2</td>
<td>(d_2 = -0.5x^3)</td>
<td>0</td>
<td>((\dot{x})(\dot{x}))</td>
<td>0.0309</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>(\dot{x}^2)</td>
<td>-0.2807</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>(\dot{x}^3)</td>
<td>-0.9287</td>
<td>(\sqrt{\ })</td>
</tr>
</tbody>
</table>

From the Table 5.5 it is clear that the correlation coefficient is highest for the function \(\dot{x}^3\) and hence this function is chosen as the first non-linear function for the system. The new equation for the system becomes:

\[
\begin{pmatrix}
    \ddot{x} \\
    \dot{x}
\end{pmatrix} =
\begin{bmatrix}
    0 & 1 \\
    -1 & 0
\end{bmatrix}
\begin{pmatrix}
    x \\
    \dot{x}
\end{pmatrix} +
\begin{pmatrix}
    0 \\
    \alpha(x^3)
\end{pmatrix}
\]

(5.53)

The model error data over the entire time domain can be written in the form of linear equations as:

\[
d_2(t_1) = \alpha f_1(\dot{x}(t_1))
\]

\[
d_2(t_2) = \alpha f_1(\dot{x}(t_2))
\]

\[
d_2(t_3) = \alpha f_1(\dot{x}(t_3))
\]

\[
\vdots = \vdots
\]

\[
d_2(t_i) = \alpha f_1(\dot{x}(t_i))
\]

(5.54)
or, in the matrix form,

\[ D_{I\times1} = M_{I\times1} P_{1\times1} \]  \hspace{1cm} \text{(5.55)}

The least squares fit was performed on this data value of \( \alpha \) identified from the program
are \( \alpha = -0.2921 \) (for Set 1) and \( \alpha = -0.4951 \) (for Set 2). The validity of the result is
confirmed since the state estimates match the input states within the covariance constraint.
Hence it shows that the MME is able to identify the nonlinearity function within a reason-
able degree of accuracy using only the acceleration data.
Figure 5.18: True Position (solid) and MME Estimated (+) with no noise

Figure 5.19: True Velocity (solid) and MME Estimated (+) with no noise
Figure 5.20: True Acceleration (solid) and MME Estimated (+) with no noise

Figure 5.21: True Model Error $d_2$ (solid) and MME Estimated (+) with no noise
5.3.3 Measurements with noise

Noise of various magnitudes were added to noiseless measurements generated earlier. The noise levels considered were 1%, 5% and 10%.

The measurement model is given by:

\[ \bar{y}(t_k) = [0 \quad 1] [\hat{x}(t_k); \hat{\nu}(t_k)]^T \]  \hspace{1cm} (5.56)

where

\[ \hat{\nu}(t_k) = \nu(t_k) + q_k \]  \hspace{1cm} (5.57)

Here \( q_k \) represents measurement noise. It is assumed to be a zero-mean gaussian distribution process of known covariance \( R_k \). The results obtained from the program are explained below.

**Measurements with 1% noise**

The convergence criteria here is taken as \( R_k = 0.0001 \). Figures 5.22-5.25 show the comparison of the MME estimated states and model error with the true states and true model error. It was found that the values for the vector \( d_{\nu}(t) \approx 0 \) as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to be zero. Figure 5.25 shows the actual model error, \( d_{\omega} \), compared to the estimated model error. Let the estimated position vector be \( \hat{x} \) and the estimated velocity vector be \( \hat{\dot{x}} \). The Table 5.6 shows the nonlinear functions used and the cross-correlation coefficient, \( C(d(t), f) \), obtained for the functions after the first iteration.
From Table 5.6 it is clear that the correlation coefficient is highest for the function $\dot{x}^3$ and hence this function is chosen as the first non-linear function for the system. The new equation for the system becomes:

$$
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} =
\begin{bmatrix}
0 & 1 \\
-1 & 0
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix}
+ 
\begin{pmatrix}
0 \\
\alpha(x^3)
\end{pmatrix}
$$

(5.58)

The model error data over the entire time domain can be written in the form of linear equations as:

$$
d_2(t_1) = \alpha f_1(\dot{x}(t_1))
$$

$$
d_2(t_2) = \alpha f_1(\dot{x}(t_2))
$$

$$
d_2(t_3) = \alpha f_1(\dot{x}(t_3))
$$

$$
\vdots = \vdots
$$

$$
d_2(t_t) = \alpha f_1(\dot{x}(t_t))
$$

(5.59)
or, in the matrix form,

\[ D_{1 \times 1} = M_{1 \times 1} P_{1 \times 1} \]  

(5.60)

The least squares fit was performed on this data value of \( \alpha \) identified from the program are \( \alpha = -0.2920 \) (for Set 1) and \( \alpha = -0.4914 \) (for Set 2). The validity of the result is confirmed since the state estimates match the input states to an accuracy within the covariance constraint. Hence it shows that the MME is able to identify the nonlinearity function within a reasonable degree of accuracy even in the presence of 1% noise in the signal.
Figure 5.22: True Position (solid) and MME Estimated (+) with 1% noise

Figure 5.23: True Velocity (solid) and MME Estimated (+) with 1% noise
Figure 5.24: True Acceleration (solid) and MME Estimated (+) with 1% noise

Figure 5.25: True Model Error $d_2$ (solid) and MME Estimated (+) with 1% noise
Table 5.7: CORRELATION COEFFICIENTS FOR DATA WITH 5% NOISE

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( d_2 = -0.3x^3 )</td>
<td>5%</td>
<td>(( \dot{x} ))(( \dot{x} ))</td>
<td>0.1630</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>( \ddot{x}^2 )</td>
<td>-0.1347</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>( \dot{x}^3 )</td>
<td>-0.9410</td>
<td>( \sqrt{\text{ }} )</td>
</tr>
<tr>
<td>2</td>
<td>( d_2 = -0.5x^3 )</td>
<td>5%</td>
<td>(( \dot{x} ))(( \dot{x} ))</td>
<td>0.0554</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>( \ddot{x}^2 )</td>
<td>-0.2060</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>( \dot{x}^3 )</td>
<td>-0.9330</td>
<td>( \sqrt{\text{ }} )</td>
</tr>
</tbody>
</table>

Measurements with 5% noise

The convergence criteria here is taken as \( R_k = 0.0025 \). Figures 5.26-5.29 show the comparison of the MME estimated states and model error with the true states and true model error. It was noted that the values for the vector \( \mathbf{d}_\mathbf{1}(t) \approx 0 \) as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to zero Figure 5.29 shows the actual model error, \( \mathbf{d}_\mathbf{a} \), compared to the estimated model error.

The Table 5.7 shows the nonlinear functions used and the cross-correlation coefficient, C(d(t),f), obtained for the functions after the first iteration.

From the Table 5.7 it is clear that the correlation coefficient is highest for the function \( \dot{x}^3 \) and hence this function is chosen as the first non-linear function for the system. The
new equation for the system becomes:

\[
\begin{pmatrix}
\dot{x} \\
\dot{x}
\end{pmatrix}
= \begin{bmatrix}
0 & 1 \\
-1 & 0
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix}
+ \begin{pmatrix}
0 \\
\alpha(x^3)
\end{pmatrix}
\]

(5.61)

The least squares fit was performed on this data value of \(\alpha\) identified from the program are \(\alpha = -0.2988\) (for Set 1) and \(\alpha = -0.5026\) (for Set 2). Hence it shows that the MME is able to identify the nonlinearity function within a reasonable degree of accuracy even in the presence of 5% noise in the acceleration signal.
Figure 5.26: True Position (solid) and MME Estimated (+) with 5% noise

Figure 5.27: True Velocity (solid) and MME Estimated (+) with 5% noise
Figure 5.28: True Acceleration (solid) and MME Estimated (+) with 5% noise

Figure 5.29: True Model Error $d_2$ (solid) and MME Estimated (+) with 5% noise
Table 5.8: CORRELATION COEFFICIENTS FOR DATA WITH 10% NOISE

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(d_2 = -0.3x^3)</td>
<td>10%</td>
<td>((\dot{x})(\dot{x}))</td>
<td>-0.2050</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>(\dot{x}^3)</td>
<td>-0.1563</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>(\dot{x}^2)</td>
<td>-0.9241</td>
<td>(\checkmark)</td>
</tr>
<tr>
<td>2</td>
<td>(d_2 = -0.5x^3)</td>
<td>10%</td>
<td>((\dot{x})(\dot{x}))</td>
<td>0.0912</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>(\dot{x}^2)</td>
<td>0.2096</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>(\dot{x}^3)</td>
<td>-0.9304</td>
<td>(\checkmark)</td>
</tr>
</tbody>
</table>

**Measurements with 10% noise**

The convergence criteria here is taken to be \(R_k = 0.01\). Figures 5.30-5.33 show the comparison of the MME estimated states and model error with the true states and true model error. It was noted that the values for the vector \(d_{\alpha}(t) \approx 0\) as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to zero Figure 5.33 shows the actual model error, \(d_{\alpha}\), compared to the estimated model error. Let the estimated position vector be \(\hat{x}\) and the estimated velocity vector be \(\dot{\hat{x}}\). The Table 5.8 shows the nonlinear functions used and the cross-correlation coefficient, \(C(d(t),f)\), obtained for the functions after the first iteration. From the Table 5.8 it is clear that the correlation coefficient is highest for the function \(\dot{x}^3\) and hence this function is chosen as the first nonlinear function for the system.

The least squares fit was performed on this data value of \(\alpha\) identified from the program.
are $\alpha = -0.2892$ (for Set 1) and $\alpha = -0.5252$ (for Set 2). Hence it shows that the MME is able to identify the nonlinearity function within a reasonable degree of accuracy even in the presence of 10% noise in the acceleration signal within 5%.
Figure 5.30: True Position (solid) and MME Estimated (+) with 10% noise

Figure 5.31: True Velocity (solid) and MME Estimated (+) with 10% noise
Figure 5.32: True Acceleration (solid) and MME Estimated (+) with 10% noise

Figure 5.33: True Model Error $d_2$ (solid) and MME Estimated (+) with 10% noise
5.4 Conclusion

This chapter shows that the MME is able to identify the mathematical model of the SDOF system even in the presence of substantial amount of noise in the signal. It also shows that the MME is able to identify the system model even when only the acceleration data is available for the system. This was verified for the signal having up to 10% noise.
Chapter 6

Conservative Two-Degree-of-Freedom Nonlinear Systems

In this chapter an example of a conservative, nonlinear system having two degrees of freedom is described. The MME method for identifying the nonlinearity of the free (undamped and unforced) oscillations is presented. The Range-Kutta method is used to find the actual response of the system and this data is used in the MME technique to determine the nonlinear function of the system and then the least squares method is used to identify the coefficient of the nonlinear term.

6.1 Two Degree of Freedom System

Consider the free response of a conservative system having a two degrees of freedom as shown in Figure 6.1. The system is a two-mass and 3-spring system. Spring $S_1$ is a nonlinear spring whereas springs $S_2$ and $S_3$ are linear springs. Langrange’s Equations are used to derive the equation of motion of the system. The potential energy for each spring system can be written as:
Figure 6.1: Two DOF Nonlinear Spring-Mass System

Potential Energy for Spring \( S_1 \): \[ V_1 = \frac{1}{2} x_1^2 + \frac{1}{8} x_1^4 \]

Potential Energy for Spring \( S_2 \): \[ V_2 = \frac{1}{2} (x_2 - x_1)^2 \]

Potential Energy for Spring \( S_3 \): \[ V_3 = \frac{1}{2} (x_2)^2 \]

Similarly the kinetic energies for the two-mass systems can be given as (assuming \( M_1 = 1 \) and \( M_2 = 1 \)):

Kinetic Energy for Mass \( M_1 \): \[ T_1 = \frac{1}{2} \dot{x}_1^2 \]

Kinetic Energy for Spring \( M_2 \): \[ T_2 = \frac{1}{2} (\dot{x}_2)^2 \]

Applying Lagrange’s equation:

\[ \ddot{x}_1 + 2x_1 + \frac{x_1^3}{2} - x_2 = 0 \] \hspace{1cm} (6.1)

\[ \ddot{x}_2 + 2x_2 - x_1 = 0 \] \hspace{1cm} (6.2)
In state space form the above equation can be written as:

\[
\begin{pmatrix}
    \dot{x}_1 \\
    \dot{x}_2 \\
    \ddot{x}_1 \\
    \ddot{x}_2 \\
\end{pmatrix} = 
\begin{pmatrix}
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1 \\
    -2 & 1 & 0 & 0 \\
    1 & -2 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
    x_1 \\
    x_2 \\
    \dot{x}_1 \\
    \dot{x}_2 \\
\end{pmatrix} + 
\begin{pmatrix}
    0 \\
    0 \\
    -\frac{x_1^3}{2} \\
    0 \\
\end{pmatrix}
\]

(6.3)

where the dot indicates differentiation with respect to time. For simplicity, the system is unforced. The term \( \frac{x_1^3}{2} \) denotes the nonlinear term to be identified by the MME-based identification algorithm. Measurements were simulated by solving Equation (6.1) using Runge Kutta integration. The data obtained from this method was used as the input to the MME algorithm instead of experimental data to demonstrate the ability of the identification algorithm to identify the model with no prior knowledge of the nonlinear function. The program written to perform this simulation and to identify the nonlinearity is presented in the Appendix F. Three nonlinear functions were chosen to check the convergence of the method. The functions used were \( \cos(x), x^2 \) and \( x^3 \).

In addition to the above model, another nonlinear system with two non-linearities was also considered. This included the non linear term \( (\dot{x}_1^2x_1) \) in addition to the cubic nonlinearity. The equation for such a system is given below

\[
\begin{pmatrix}
    \dot{x}_1 \\
    \dot{x}_2 \\
    \ddot{x}_1 \\
    \ddot{x}_2 \\
\end{pmatrix} = 
\begin{pmatrix}
    0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 1 \\
    -2 & 1 & 0 & 0 \\
    1 & -2 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
    x_1 \\
    x_2 \\
    \dot{x}_1 \\
    \dot{x}_2 \\
\end{pmatrix} + 
\begin{pmatrix}
    0 \\
    0 \\
    -\frac{x_1^3}{2} - 0.1\dot{x}_1^2x_1 \\
    0 \\
\end{pmatrix}
\]
In both cases the assumed model used for MME analysis consisted of the undamped linear oscillator part of the system:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix}
= \begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix}
\] (6.4)

6.1.1 Simulation

The noiseless measurements are simulated by integrating Equation (6.1) using Runge-Kutta integration for a period of 10 seconds with a step size of 0.1 seconds (equivalent to a 10 Hz sampling frequency). Hence the position, velocity and acceleration vectors for the system are available.

The system of equations can be represented as,

\[
\dot{\mathbf{X}} = A\mathbf{X}
\] (6.5)

Let \(\mathbf{X}\) denote the response vector and velocity vectors for the system and \(\dot{\mathbf{X}}\) denotes the velocity and acceleration vector. Hence

\[
\dot{\mathbf{X}} = \begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix}
\]
and

\[
\begin{pmatrix}
\hat{x}_1 \\
\hat{x}_2 \\
\hat{\dot{x}}_1 \\
\hat{\dot{x}}_2
\end{pmatrix}
\]

In order to simulate a real world problem, noise of three varying magnitudes were added to the signal. The noise levels considered were 1%, 5% and 10%. The noise was generated using the `rand` program for generation of random numbers in Matlab and this was added to the measurement signal. As in the previous chapter, at first the position and velocity vectors are taken as the known measured output from the system and these are used as the input into MME identification algorithm. The nonlinearity in the system is first identified and then using the correlation procedure, the coefficient of the non-linear term is identified. The position, velocity and the acceleration were input into the identification procedure with various levels of noise level and the robustness of the identification procedure, to various level of noise disturbance in the signal, is evaluated.

Later it is assumed that only the acceleration data is available for the system. The acceleration data with various levels of noise in the signal was input to MME identification algorithm. The first step was to identify the unknown states, namely, the position and the velocity vectors and the model error term was to be identified. The MME algorithm evaluates the states and the model error simultaneously. Once the model error and the stated have been estimated to a considerable degree of accuracy, the correlation procedure is used to identify the non linear function form and the least squares fit is used to identify the coefficient of the nonlinear function.
6.2 Position and Velocity Measurement

6.2.1 Noiseless Measurements

Noiseless measurement was first used as the input to the MME program. Both the position and velocity are assumed to be available measurements and this is used to identify the nonlinearity in the system.

The measurement model is given by:

\[ \hat{Y}(t_k) = [\hat{X}(t_k)] \]  \hspace{1cm} (6.6)

and the assumed model can be written as,

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix}
+ \begin{pmatrix}
d_1 \\
d_2 \\
d_3 \\
d_4
\end{pmatrix} \hspace{1cm} (6.7)
\]

where \(d_1, d_2, d_3\) and \(d_4\) are the model error terms that are to be identified by the MME algorithm.

The convergence criteria here is taken as \(R_k = 0.0025\). Figures 6.2-6.5 show the comparison of the MME estimated states and model error with the true states and true model error. It was found that the values for the vectors \(d_1(t) \approx 0, d_2(t) \approx 0\) and \(d_4(t) \approx 0\) as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence these model error vectors are taken to be zero. Figure 6.4 shows the actual model error, \(d_3\), compared to the estimated model error. This model error term would be later on
Table 6.1: CORRELATION COEFFICIENTS FOR DATA WITH NO NOISE

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>0</td>
<td>(\hat{x}_1)(\hat{x}_1)^2</td>
<td>-0.4586</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>\hat{x}_1^3</td>
<td>-0.9601</td>
<td>✓</td>
</tr>
<tr>
<td>2</td>
<td>(d_3 = -0.5x_1^3)</td>
<td>0</td>
<td>(\hat{x}_1)(\hat{x}_1)^2</td>
<td>-0.6245</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>0.1(x_1)(\dot{x}_1)^2</td>
<td>0</td>
<td>\hat{x}_1^3</td>
<td>-0.9306</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>(d_4 = 0)</td>
<td>0</td>
<td>\dot{x}_2^2</td>
<td>0.0563</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td>\dot{x}_2</td>
<td>-0.2155</td>
<td></td>
</tr>
</tbody>
</table>

used to identify the nonlinear function and identify the coefficient. The two are clearly in close agreement. Figure 6.5 shows the actual model error, \(d_4\), compared to the estimated model error. This model error term would be later on used to identify the nonlinear function and identify the coefficient. The two are clearly in close agreement. The Table 6.1 shows the nonlinear functions used and the cross-correlation coefficient, C(d(t),f), obtained for the functions after the first iteration. From the Table 6.1 it is clear that the correlation coefficient is highest for the function \(\hat{x}_1^3\) in Set 1 and hence this function is chosen as the first non-linear function for the system.
The updated equation for the system (SET 1) can now be written as:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
0.4823x_1^3 \\
0
\end{pmatrix}
\]

For Set 2, the correlation coefficient is highest for the function \( \dot{x}^3 \) and it is also high for the function \((\dot{x}_1)(\dot{x}_1)^2\) as expected. This goes to show that both functions have contribution to the model error term. The procedure is to first select the function having the highest correlation and run MME to get the coefficient of this non-linear function, then update the model error by removing the contribution of this term and then rerun MME to identify the coefficient of the second function. The Table 6.1 shows the results of the first MME run only, which shows the contribution of two functions in the model error term. The updated equation for Set 2 becomes

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
\alpha_2x_1^3 + \alpha_3(\dot{x}_1)^2x_1 \\
0
\end{pmatrix}
\]

The least squares fit was performed on this data value of \( \alpha \) identified from the program are \( \alpha_1 = -0.4892 \) and \( \alpha_3 = -0.1038 \). The actual values were \( \alpha_1 = -0.5 \) and \( \alpha_3 = -0.1 \) (for Set2). The validity of the result is confirmed since the state estimates match the input states to an accuracy within the covariance constraint. Figures 6.2-6.5 compare the true states and true model error with MME estimated states and model error.
Figure 6.2: True Position (solid) and MME Estimated (+) with no noise

Figure 6.3: True Velocity (solid) and MME Estimated (+) with no noise
Figure 6.4: True Model Error-\(d_3\) (solid) and MME Estimated (+) with no noise

Figure 6.5: True Model Error-\(d_4\) (solid) and MME Estimated (+) with no noise
6.2.2 Measurements with 5% Noise

5% noise was added into the signal and MME was run with this data.

The measurement model is given by:

\[
\hat{Y}(t_k) = [\hat{X}(t_k)] + Q_k
\]  

(6.8)

Here \(Q_k\) represents measurement noise. It is assumed to be a zero-mean gaussian distribution process of known covariance \(R_k\), and the assumed model can be written as,

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} +
\begin{pmatrix}
d_1 \\
d_2 \\
d_3 \\
d_4
\end{pmatrix}
\]  

(6.9)

The Table 6.2 shows the non-linear functions used and the cross-correlation coefficient, \(C(d(t), f)\), obtained for the functions after the first iteration.

From Table 6.2 it is clear that the correlation coefficient is highest for the function \(\dot{x}^3\) in Set 1 and hence this function is chosen as the first non-linear function for the system.

The updated equation for the system (SET 1) can now be written as:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
\alpha_1 x_1^3 \\
0
\end{pmatrix}
\]

For Set 2, the correlation coefficient is highest for the function \(\dot{x}^3\) and it is also high for the function \((\dot{x}_1)(\dot{x}_1)^2\) as expected. This goes to show that both functions have contribution
to the model error term. The procedure is to first select the function having the highest
correlation and run MME to get the coefficient of this non linear function, then update the
model error by removing the contribution of this term and then rerun MME to identify
the coefficient of the second function. Table 6.2 shows the results of the first MME run
only, which shows the contribution of two functions in the model error term. The updated
equation for Set 2 becomes

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
\alpha_2 x_1^3 + \alpha_3 (\dot{x}_1)^2 x_1 \\
0
\end{pmatrix}
\]

The least squares fit was performed on this data and the value of \( \alpha \) identified from the
program are \( \alpha_1 = -0.4955 \) (for Set1), \( \alpha_2 = -0.5178 \) and \( \alpha_3 = -0.0912 \) (for Set2). The
validity of the result is confirmed since the state estimates match the input states to an accuracy within the covariance constraint. The Figures 6.6-6.9 compare the true states and true model error with the MME estimated states and model error.
Figure 6.6: True Position ($x$) (solid - no noise, 'o' - with 5% noise) and MME Estimated (+)

Figure 6.7: True Velocity ($\dot{x}$)(solid - no noise, 'o' - with 5% noise) and MME Estimated (+)
6.2.3 Measurements with 10% Noise

10% noise was added into the signal and MME was run with this data.
The measurement model is given by :

$$ \tilde{Y}(t_k) = [\tilde{X}(t_k)] + Q_k $$

(6.10)

Here $Q_k$ represents measurement noise. It is assumed to be a zero-mean gaussian distribution process of known covariance $R_k$. and the assumed model can be written as,

$$
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{bmatrix} = 
\begin{bmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{bmatrix} + 
\begin{bmatrix}
d_1 \\
d_2 \\
d_3 \\
d_4
\end{bmatrix}

(6.11)

The Table 6.3 shows the non-linear functions used and the cross-correlation coefficient, $C(d(t),f)$, obtained for the functions after the first iteration.

From Table 6.3 it is clear that the correlation coefficient is highest for the function $\hat{x}^3$. 

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$d_3 = -0.5x_1^3$</td>
<td>10%</td>
<td>$(\hat{x}_1)(\hat{x}_1)^2$</td>
<td>-0.5142</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>$\hat{x}_1^3$</td>
<td>-0.8779</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td></td>
<td>$d_4 = 0$</td>
<td>10%</td>
<td>$\hat{x}_2\hat{x}_2$</td>
<td>-0.0689</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>$d_3 = -0.5x_1^3$</td>
<td>10%</td>
<td>$(\hat{x}_1)(\hat{x}_1)^2$</td>
<td>-0.7514</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td></td>
<td>$0.1(x_1)(\hat{x}_1)^2$</td>
<td>10%</td>
<td>$\hat{x}_1^3$</td>
<td>-0.7696</td>
<td>$\checkmark$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>$\hat{x}_2\hat{x}_2$</td>
<td>-0.1418</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$d_4 = 0$</td>
<td>10%</td>
<td>$\hat{x}_2\hat{x}_2$</td>
<td>-0.1652</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.3: CORRELATION COEFFICIENTS FOR DATA WITH 10% NOISE
in Set 1 and hence this function is chosen as the first non linear function for the system.

The updated equation for the system (SET 1) can now be written as:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
\alpha_1 x_1^3 \\
0
\end{pmatrix}
\]

For Set 2, the correlation coefficient is highest for the function \(\dot{x}_1^3\) and it is also high for the function \((\dot{x}_1)^2\) as expected. This goes to show that both functions have contribution to the model error term. The procedure is to first select the function having the highest correlation and run MME to get the coefficient of this non linear function, then update the model error by removing the contribution of this term and then rerun MME to identify the coefficient of the second function. The Table 6.3 shows the results of the first MME run only, which clearly shows the contribution of two functions in the model error term. The updated equation for Set 2 becomes:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix} =
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
\alpha_2 x_1^3 + \alpha_3 (\dot{x}_1)^2 x_1 \\
0
\end{pmatrix}
\]

The least squares fit was performed on this data and the value of \(\alpha\) identified from the program are \(\alpha_1 = -0.4883\) (for Set1), \(\alpha_2 = -0.4836\) and \(\alpha_3 = -0.0876\) (for Set2). The validity of the result is confirmed since the state estimates match the input states to an accuracy within the covariance constraint. The Figures (6.10-6.13) compare the true states and true model error with the MME estimated states and model error.
Figure 6.10: True Position (x) (solid - no noise, 'o' - with 10% noise) and MME Estimated (+)

Figure 6.11: True Velocity (\(\dot{x}\)) (solid - no noise, 'o' - with 10% noise) and MME Estimated (+)
Figure 6.12: True Model Error $d_3$ (solid) and MME Estimated (+)

Figure 6.13: True Model Error $d_4$ (solid) and MME Estimated (+)
6.3 Acceleration Measurement

The measurement model is defined as:

\[
\hat{y}(t_k) = \hat{X} = \begin{bmatrix} 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{X}(t_k); \ddot{X}(t_k) \end{bmatrix}^T
\]  

(6.12)

Let \( \hat{X} \) denote the response acceleration vector.

\[
\hat{X} = \begin{pmatrix} \ddot{x}_1 \\ \ddot{x}_2 \\ \vdots \\ \ddot{x}_n \end{pmatrix}
\]

where \( \ddot{x}_1 \) denotes the response at time \( t_1 \), \( \ddot{x}_2 \) at \( t_2 \) and so on. This acceleration data is the available measurement. Since the algorithm development has been explained in detail for a single degree of freedom system in the previous chapter. Hence only the final transformation equations for the constraint evaluation are given here.

6.3.1 Multiple Shooting algorithm development

The system equation is given as

\[
\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \ddot{x}_1 \\ \ddot{x}_2 \end{pmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -2 & 1 & 0 & 0 \\ 1 & -2 & 0 & 0 \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} + \begin{pmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \end{pmatrix}
\]  

(6.13)

where

\[
d_1 = -\frac{\lambda_1}{2W_1}
\]  

(6.14)
\[ d_2 = -\frac{\lambda_2}{2W_2} \quad (6.15) \]

\[ d_3 = -\frac{\lambda_3}{2W_3} \quad (6.16) \]

\[ d_4 = -\frac{\lambda_4}{2W_4} \quad (6.17) \]

Rewriting the convergence criteria given in Equation (3.9) in terms of the Lagrange multipliers, we have

\[ \Delta_{k+1}(t^+_k) = \Delta_k(t^-_k) + 2R_k^{-1}[\tilde{y}(t_k) - \tilde{G}] \quad (6.18) \]

The two velocity convergence criteria can be written as

\[ \Delta_{1(k+1)}(t^+_k) = \Delta_{1k}(t^-_k) + 2R_k^{-1}\left[-\frac{\lambda_{1k}(t^-_k)}{2W_1}\right] \quad (6.19) \]

and

\[ \Delta_{2(k+1)}(t^+_k) = \Delta_{2k}(t^-_k) + 2R_k^{-1}\left[-\frac{\lambda_{2k}(t^-_k)}{2W_2}\right] \quad (6.20) \]

The two acceleration convergence criteria can be written as

\[ \Delta_{3(k+1)}(t^+_k) = \Delta_{3k}(t^-_k) + 2R_k^{-1}[\tilde{y}_{k}^{-\prime} - (-2\ddot{x}_1 + \ddot{x}_2 - \frac{\lambda_{3k}(t^-_k)}{2W_3})] \quad (6.21) \]

and

\[ \Delta_{4(k+1)}(t^+_k) = \Delta_{4k}(t^-_k) + 2R_k^{-1}[\tilde{y}_{k}^{-\prime} - (\ddot{x}_1 - 2\ddot{x}_2 - \frac{\lambda_{4k}(t^-_k)}{2W_4})] \quad (6.22) \]
The state and lagrange multiplier updates can be written in the matrix form as:

\[
\begin{pmatrix}
\dot{x}_{1(i+1)}(t_i^+) \\
\dot{x}_{2(i+1)}(t_i^+) \\
\dot{x}_{1(i+1)}(t_i^+) \\
\dot{x}_{2(i+1)}(t_i^+) \\
\lambda_{1(i+1)}(t_i^+) \\
\lambda_{2(i+1)}(t_i^+) \\
\lambda_{3(i+1)}(t_i^+) \\
\lambda_{4(i+1)}(t_i^+)
\end{pmatrix} =
\begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
4R_k^{-1} & -2R_k^{-1} & 0 & 0 & 0 & 0 & 0 & 0 \\
-2R_k^{-1} & 4R_k^{-1} & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
\dot{x}_{1(i)}(t_i^+) \\
\dot{x}_{2(i)}(t_i^+) \\
\dot{x}_{1(i)}(t_i^+) \\
\dot{x}_{2(i)}(t_i^+) \\
\lambda_{1(i)}(t_i^+) \\
\lambda_{2(i)}(t_i^+) \\
\lambda_{3(i)}(t_i^+) \\
\lambda_{4(i)}(t_i^+)
\end{pmatrix} +
\begin{pmatrix}
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
2R_k^{-1}\bar{y}_{1i}
\end{pmatrix}
\]

(6.23)

The boundary conditions remain the same as the earlier

\[
\beta = B_0y(t_o^-) + B_0y(t_f^+)
\]

(6.24)
6.3.2 Measurements with noise

Noise of various magnitude was added to noiseless measurements obtained earlier. The noise levels considered were 5%, 10% and 15%.

The measurement model is given by:

\[ B_0 = \begin{pmatrix} 0 & I \\ 0 & 0 \end{pmatrix} \]  
(6.25)

\[ B_f = \begin{pmatrix} 0 & 0 \\ 0 & I \end{pmatrix} \]  
(6.26)

\[ \beta = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \]  
(6.27)

\[
\bar{y}(t_k) = [0 \ 1] [\tilde{x}(t_k) \ \tilde{z}(t_k)]^T
\]
(6.28)

where

\[
\tilde{z}(t_k) = \bar{z}(t_k) + q_k
\]
(6.29)

Here \( q_k \) represents measurement noise. It is assumed to be a zero-mean gaussian distribution process of known covariance \( R_k \). The results obtained from the program are explained below.

Measurements with 5% noise

The convergence criteria here is taken as \( R_k = 0.0001 \). Figures 6.14-6.21 show the comparison of the MME estimated states and model error with the true states and true model...
Table 6.4: CORRELATION COEFFICIENTS FOR DATA WITH 5% NOISE

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$d_3 = -0.2\dot{x}_1^2x_1$</td>
<td>5%</td>
<td>$\dot{x}_1^3$</td>
<td>-0.3898</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$\dot{x}_1\dot{x}_1$</td>
<td>-0.6726</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$\dot{x}_1^3$</td>
<td>0.0500</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$\dot{x}_1^2$</td>
<td>0.0224</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$\dot{x}_1\dot{x}_1$</td>
<td>-0.7810</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>$d_4 = 0$</td>
<td>5%</td>
<td>$\dot{x}_2^3$</td>
<td>-0.2152</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$\dot{x}_2\dot{x}_2$</td>
<td>0.0256</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$\dot{x}_2^3$</td>
<td>0.0697</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$\dot{x}_2^2$</td>
<td>-0.0574</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>5%</td>
<td>$\dot{x}_2\dot{x}_2$</td>
<td>0.0627</td>
<td></td>
</tr>
</tbody>
</table>

eff. It was found that the values for the vector $d_1$, $d_2$ and $d_4$ are very close to zero as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to be zero. Figure 6.20 shows the actual model error, $d_3$, compared to the estimated model error. Let the estimated position vector be $\hat{x}$ and the estimated velocity vector be $\dot{\hat{x}}$.

The Table 6.4 shows the nonlinear functions used and the cross-correlation coefficient, C(d(t),f), obtained for the functions after the first iteration. From the Table 6.4 it is clear that the correlation coefficient is highest for the function $\dot{x}_1^2x_1$ and hence this function is chosen as the first nonlinear function for the system. The coefficient of $\dot{x}_1^2x_1$ is identified...
as $-0.189$. The identified equation for the system is:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix}
= \begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix}
+ \begin{pmatrix}
0 \\
0 \\
-0.189\dot{x}_1^2 x_1 \\
0
\end{pmatrix}
\quad (6.30)
\]
Figure 6.14: True Position($x_1$) (solid) and MME Estimated (+)

Figure 6.15: True Position($x_2$) (solid) and MME Estimated (+)
Figure 6.16: True Velocity ($\dot{x}_1$) (solid) and MME Estimated (+)

Figure 6.17: True Velocity ($\dot{x}_2$) (solid) and MME Estimated (+)
Figure 6.18: True Acceleration ($\ddot{x}_1$) (solid - no noise, ‘o’ - with 5% noise) and MME Estimated (+)

Figure 6.19: True Acceleration ($\ddot{x}_2$) (solid - no noise, ‘o’ - with 5% noise) and MME Estimated (+)
Figure 6.20: True Model Error($d_3$) (solid) and MME Estimated (+)

Figure 6.21: True Model Error($d_4$) (solid) and MME Estimated (+)
Measurements with 10% noise

The convergence criteria here is taken as $R_k = 0.0025$. Figures 6.22-6.29 show the comparison of the MME estimated states and model error with the true states and true model error. It was found that the values for the vector $d_1, d_2$ and $d_4$ are very close to zero as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to be zero. Figure 6.28 shows the actual model error, $d_3$, compared to the estimated model error. Let the estimated position vector be $\hat{x}$ and the estimated velocity vector be $\hat{\dot{x}}$.

The Table 6.5 shows the nonlinear functions used and the cross-correlation coefficient, $C(d(t),f)$, obtained for the functions after the first iteration. From the Table 6.5 it is clear that the correlation coefficient is highest for the function $x_1\dot{x}_1^2$ and hence this function is chosen as the first non linear function for the system. The coefficient of $x_1\dot{x}_1^2$ is identified as $-0.1808$. The identified equation is:

$$
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix} = 
\begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix} + 
\begin{pmatrix}
0 \\
0 \\
-0.1808x_1\dot{x}_1^2 \\
0
\end{pmatrix}
$$

(6.31)

If the identified system is the true model then both the acceleration data should match within a very close tolerance. As shown in Figure 6.26-6.27 the two curves are in close agreement. Hence it shows that the identified model is the correct form of the system.
\[ d_3 = -0.2x_1 \dot{x}_1^2 \]

<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( d_3 = -0.2x_1 \dot{x}_1^2 )</td>
<td>10%</td>
<td>( \dot{x}_1^3 )</td>
<td>-0.4770</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>( \dot{x}_1 \dot{x}_1 )</td>
<td>0.3221</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>( \dot{x}_1^3 )</td>
<td>0.1011</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>( \dot{x}_1^2 )</td>
<td>-0.0282</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>( \dot{x}_1 \dot{x}_1 )</td>
<td>-0.6938</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>( d_4 = 0 )</td>
<td>10%</td>
<td>( \dot{x}_2^3 )</td>
<td>-0.0218</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>( \dot{x}_2 \dot{x}_2 )</td>
<td>-0.1702</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>( \dot{x}_2^3 )</td>
<td>-0.1762</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>( \dot{x}_2^2 )</td>
<td>0.1657</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>10%</td>
<td>( \dot{x}_2 \dot{x}_2 )</td>
<td>-0.0983</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.5: CORRELATION COEFFICIENTS FOR DATA WITH 10\% NOISE
Figure 6.22: True Position($x_1$) (solid) and MME Estimated (+)

Figure 6.23: True Position($x_2$) (solid) and MME Estimated (+)
Figure 6.24: True Velocity (\( \dot{x}_1 \)) (solid) and MME Estimated (+)

Figure 6.25: True Velocity (\( \dot{x}_2 \)) (solid) and MME Estimated (+)
Figure 6.26: True Acceleration ($\vec{x}_1$) (solid - no noise, 'o' - with 10% noise) and MME Estimated (+)

Figure 6.27: True Acceleration ($\vec{x}_2$) (solid - no noise, 'o' - with 10% noise) and MME Estimated (+)
Figure 6.28: True Model Error($d_3$) (solid) and MME Estimated (+)

Figure 6.29: True Model Error($d_4$) (solid) and MME Estimated (+)
Measurements with 15% noise

The convergence criteria here is taken as $R_k = 0.0025$. Figures 6.30-6.37 show the comparison of MME estimated states and model error with the true states and true model error. It was found that the values for the vector $d_1$, $d_2$ and $d_4$ are very close to zero as expected. Hence we would not bother to do a correlation and least squares fit on this data since we know that the coefficients are bound to be very small and tending to zero. Hence this model error vector is taken to be zero. Figure 6.37 shows the actual model error, $d_3$, compared to the estimated model error. Let the estimated position vector be $\hat{x}$ and the estimated velocity vector be $\dot{\hat{x}}$.

The Table 6.6 shows the nonlinear functions used and the cross-correlation coefficient, $C(d(t),f)$, obtained for the functions after the first iteration. From the Table 6.5 it is clear that the correlation coefficient is highest for the function $\dot{x}_1^2x_1$ and hence this function is chosen as the first non-linear function for the system. The coefficient of $\dot{x}_1^2x_1$ is identified as $-0.2061$. The identified equation is:

\[
\begin{pmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\ddot{x}_1 \\
\ddot{x}_2
\end{pmatrix}
= \begin{pmatrix}
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
-2 & 1 & 0 & 0 \\
1 & -2 & 0 & 0
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
\dot{x}_1 \\
\dot{x}_2
\end{pmatrix}
+ \begin{pmatrix}
0 \\
0 \\
-0.2061\dot{x}_1^2x_1 \\
0
\end{pmatrix}
\]

(6.32)

If the identified system is the true model then both the acceleration data should match within a very close tolerance. As shown in Figures 6.34-6.35, the two curves are in close agreement. Hence it shows that the identified model is the correct form of the system.
<table>
<thead>
<tr>
<th>SET</th>
<th>TRUE MODEL ERROR</th>
<th>NOISE</th>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$d_3 = -0.2\hat{x}_1^2x_1$</td>
<td>15%</td>
<td>$\hat{x}_1^3$</td>
<td>-0.4493</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_1\hat{x}_1$</td>
<td>-0.3754</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_1^3$</td>
<td>0.0348</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_1^2$</td>
<td>0.0940</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_1\hat{x}_1$</td>
<td>-0.6322</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_2^3$</td>
<td>-0.0096</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_2\hat{x}_2$</td>
<td>0.1423</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_2^3$</td>
<td>0.2763</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_2^2$</td>
<td>-0.1820</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>15%</td>
<td>$\hat{x}_2\hat{x}_2$</td>
<td>0.0564</td>
<td></td>
</tr>
</tbody>
</table>

Table 6.6: CORRELATION COEFFICIENTS FOR DATA WITH 15% NOISE
Figure 6.30: True Position($x_1$) (solid) and MME Estimated (+)

Figure 6.31: True Position($x_2$) (solid) and MME Estimated (+)
Figure 6.32: True Velocity ($\dot{x}_1$) (solid) and MME Estimated (+)

Figure 6.33: True Velocity ($\dot{x}_2$) (solid) and MME Estimated (+)
Figure 6.34: True Acceleration ($\ddot{x}_1$) (solid - no noise, 'o' - with 15% noise) and MME Estimated (+)

Figure 6.35: True Acceleration ($\ddot{x}_2$) (solid - no noise, 'o' - with 15% noise) and MME Estimated (+)
Figure 6.36: True Model Error ($d_3$) (solid) and MME Estimated (+)

Figure 6.37: True Model Error ($d_4$) (solid) and MME Estimated (+)
Chapter 7

Validation

The ability of MME to identify the model form of a non-linear panel flutter problem is demonstrated in this chapter. The simulated data for the problem is used as the input to the MME. The results are checked for validity of the identification procedure by comparing the data obtained by the numerical integration of the identified model with the simulated data. The MME is evaluated using two different linear models and the obtained non-linear models are compared for validity.

7.1 System Definition and Equation of Motion

The geometry of the panel flutter problem is depicted in Figure 7.1. The governing differential equation for the system makes use of Von Karman’s large deflection theory and is defined by Dowell [26] as

\[
D \frac{\partial^4 w}{\partial x^4} - (N_x + N_{x0}) \frac{\partial^2 w}{\partial x^2} + \rho_s h \frac{\partial^2 w}{\partial t^2} + (p - p_\infty) = \Delta p
\]

(7.1)
where

\[ N_x = \frac{Eh}{2l} \int_0^l \left( \frac{\partial w}{\partial x} \right)^2 dx \quad (7.2) \]

\( N_{x0} \) is external in-plane loading on the panel, \( \Delta p \) is the static pressure differential across the panel and \( p - p_\infty \) is the aerodynamic pressure loading given by quasi-steady, supersonic theory:

\[ p - p_\infty = \frac{\rho_f U^2}{\beta} \left[ \frac{\partial w}{\partial x} + \left( \frac{M^2 - 2}{M^2 - 1} \right) \frac{1}{U_0} \frac{\partial w}{\partial t} \right] \quad (7.3) \]

Note that the aerodynamic load is a function of \( w(x) \) and \( \dot{w} \) which adds stiffness and damping respectively to the system. This equation is non-dimensionalized into the form

\[ W''' - 6(1 - \nu^2) \left[ \int_0^l (W')^2 dx \right] W'' - R_x W'' + \frac{\partial^2 W}{\partial \tau^2} + \lambda \left[ W' + \left( \frac{M^2 - 2}{M^2 - 1} \right) \left( \frac{\mu}{\beta \lambda} \right)^2 \frac{\partial W}{\partial \tau} \right] = P \quad (7.4) \]
given the following non-dimensionalized parameters:

\[ \xi \equiv \frac{x}{l}, \quad \tau \equiv t \sqrt{\frac{D}{\rho_s hl^4}}, \quad \theta \equiv \frac{\partial}{\partial \xi} \]

\[ W \equiv \frac{w}{h}, \quad \lambda \equiv \frac{\rho_f U^2 l^3}{\beta D}, \quad \mu \equiv \frac{\rho_f l}{\rho_s h} \quad (7.5) \]

\[ R_x \equiv \frac{N_{x0} l^2}{D}, \quad P \equiv \frac{\Delta p l^4}{Dh} \quad (7.6) \]

Here the spatial derivatives of \( W \) are discretized through the use of finite differences to obtain a series of nonlinear second order differential equations. Equation (7.4) then becomes:

\[ \ddot{W} + [C_A] \dot{W} + [K_s - K_N + K_A]W = P \quad (7.7) \]

where \( C_A \) is a matrix representing the aerodynamic damping coefficient at each node of the finite difference model. The matrices \( K_s, K_N, \) and \( K_A \) represent the linear, nonlinear, and aerodynamic stiffening parameters respectively (from hereon the summation of the stiffness matrices is denoted as \( K \)). The nonlinear integral is approximated by:

\[ \int_0^t (W')^2 d\xi = \frac{\Delta \xi}{2} \left( -3W_1 + 4W_2 - W_3 \right)^2 \]

\[ + \Delta \xi \sum_{i=2}^{i=m-1} \left( \frac{W_{i+1} - W_{i-1}}{2 \Delta \xi} \right)^2 \]

\[ + \frac{\Delta \xi}{2} \left( 3W_m - 4W_{m-1} + W_{m-2} \right)^2 \quad (7.8) \]

\[ \int_0^t (W')^2 d\xi = \frac{\Delta \xi}{2} \left( -3W_1 + 4W_2 - W_3 \right)^2 \]

\[ + \Delta \xi \sum_{i=2}^{i=m-1} \left( \frac{W_{i+1} - W_{i-1}}{2 \Delta \xi} \right)^2 \]

\[ + \frac{\Delta \xi}{2} \left( 3W_m - 4W_{m-1} + W_{m-2} \right)^2 \quad (7.9) \]

\[ + \frac{\Delta \xi}{2} \left( 3W_m - 4W_{m-1} + W_{m-2} \right)^2 \quad (7.10) \]

Equation 7.7 can also be put into state space form:

\[ \begin{pmatrix} \dot{W} \\ \ddot{W} \end{pmatrix} = \begin{bmatrix} 0 & I \\ -K & -C_A \end{bmatrix} \begin{pmatrix} W \\ \dot{W} \end{pmatrix} + \begin{pmatrix} 0 \\ P \end{pmatrix} \]

The discretized system can now be numerically time integrated. The nonlinear stiffness term is the only part of the equation that must be re-evaluated at each time step. The data obtained by the numerical integration of the above equation was used as the input for system identification.
7.2 Data Generation

The code used for simulation was obtained from Mortara [27]. The integration time steps were 0.001 secs and the initial amplitude was 0.25 and the initial velocity was 0 for the center node. The position and velocity plot for node number 32 is shown in Figure 7.3-7.4. From the figures it can be noted that the initial displacements consists of more than one mode. After 6 seconds the vibration settles into the first nonlinear mode. The first step is to reduce the system to an equivalent one degree of freedom system. In order to do that position and velocity vectors are dot multiplied with one of the steady state position vectors. In this case the vector at time step 8000 (at 8 second time step) is used, Figure 7.2. This would give the generalized first modal position and velocity vector for an equivalent one degree of freedom linear system. The generalized position and velocity vector for the one degree of freedom system is shown in Figure 7.5-7.6.

7.3 System Identification

The velocity and position vectors between the times 9.5 secs and 9.9 secs as shown in Figure (7.7-7.8) was used for system identification. To create a real world situation 10% noise was added into the signal. The linear system to start with was assumed to be a single degree of freedom model as shown in Equation (7.11)

\[
\begin{bmatrix}
    \dot{x} \\
    \ddot{x}
\end{bmatrix} = 
\begin{bmatrix}
    0 & 1 \\
    -1 & 0
\end{bmatrix}
\begin{bmatrix}
    x \\
    \dot{x}
\end{bmatrix} + 
\begin{bmatrix}
    d_1 \\
    d_2
\end{bmatrix}
\] (7.11)

The system identification was done using the program in Appendix H. The convergence covariance constraint was set at 0.0025. Figure 7.9 show the model error terms \(d_1\) and \(d_2\).
Figure 7.2: Steady State Panel deflection at t=8 secs.

Figure 7.3: Position profile of Node 32 with time
We note that as expected the model error term \( d_1 \) is close to zero as compared to \( d_2 \).

### 7.4 Correlation and Coefficient Identification

The model error term \( d_2 \) is used to identify the nonlinearity in the system. To start with, six nonlinear functions were considered for identifying the correlation coefficient. The Table 7.1 shows the list of functions used and the correlation coefficient identified after the first iteration.

From the Table 7.1 it is noted that the function \( \hat{x} \) has the highest correlation coefficient of -0.9848 and hence this function is taken as the first function whose coefficient has to be identified. The coefficient identification is done subsequently and the coefficient is

![Figure 7.4: Velocity profile of Node 32 with time](image-url)
-1250.6165. Hence the new model equation is given by Equation (7.12).

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix}
= \begin{bmatrix}
0 & 1 \\
-1251.6165 & 0 
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix}
+ \begin{pmatrix}
0 \\
\dot{d}_{2a}
\end{pmatrix}
\]

(7.12)
Figure 7.7: Position data used for MME

Figure 7.8: Velocity data used for MME
Figure 7.9: Model Errors identified by the MME, d2(solid) and d1(+)

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
<th>COEFFICIENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\dot{x}^3$</td>
<td>$-0.9122$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(\dot{x}^2)(\dot{x})$</td>
<td>$-0.1000$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(\dot{x})^2\ddot{x}$</td>
<td>$-0.3745$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$(\dot{x})^3$</td>
<td>$-0.4564$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\ddot{x}$</td>
<td>$-0.4014$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\dot{x}$</td>
<td>$-0.9848$</td>
<td>$\sqrt{}$</td>
<td>$-1250.6165$</td>
</tr>
</tbody>
</table>

Table 7.1: CORRELATION COEFFICIENTS AFTER FIRST ITERATION
In Equation (7.12) the term $d_{2a}$ is the new model error term calculated as shown in Eqn.

$$d_{2a} = d_2 + 1250.6165\dot{x}$$  \hspace{1cm} (7.13)

The Figure 7.10 shows the model error term before and after the first iteration. As can be seen the model error has subsequently reduced drastically. The next step is to use the new model error term $d_{2a}$ for the identification of the new nonlinear function. The Table 7.2 shows the resulting correlation coefficients after the second iteration. From the Table 7.2 we note that the function $(\dot{x})^2\ddot{x}$ has the highest correlation coefficient of 0.7217 and hence this function is taken as the second function whose coefficient has to be identified. The coefficient identification is done subsequently and the coefficient is 0.03539. Hence the new model equation is given by Equation (7.14)

$$\begin{pmatrix} \dot{x} \\ \ddot{x} \end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -1251.6165 & 0 \end{bmatrix} \begin{pmatrix} x \\ \dot{x} \end{pmatrix} + \begin{pmatrix} 0 \\ 0.03539(\dot{x})^2\ddot{x} \end{pmatrix} + \begin{pmatrix} 0 \\ d_{2b} \end{pmatrix}$$  \hspace{1cm} (7.14)

![Figure 7.10: The Model Error term $d_2$(solid) and $d_{2a}$(+)](image)
The new model error \( (d_{2b}) \) is used for a new function identification and then the coefficient is identified. This repetitive procedure is continued until the model error term is very small, denoting that all the model error has been accounted for by the nonlinear functions in the identified model. The Tables (7.3-7.4) show the correlation coefficients along with the coefficient identified after the third and fourth iteration. The model error term after the fourth iteration is plotted against the first model error term. This is shown in Figure 7.11.

As can be seen, the model error term is almost equal to zero. This could be a sufficient indication to stop the iteration process and compare the position and velocity of the identified model with the input position and velocity. If the identified model is the correct model of the system then the two vectors should be in close agreement. After the fourth iteration

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
<th>COEFFICIENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \dot{x}^3 )</td>
<td>-0.6367</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( (\dot{x}^2)(\ddot{x}) )</td>
<td>-0.1576</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( (\dddot{x})^2\dot{x} )</td>
<td>0.7217</td>
<td>( \sqrt{\cdot} )</td>
<td>0.03539</td>
</tr>
<tr>
<td>( (\dddot{x})^3 )</td>
<td>-0.0784</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \dot{x} )</td>
<td>-0.1136</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( \ddot{x} )</td>
<td>-0.2199</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.2: CORRELATION COEFFICIENTS AFTER SECOND ITERATION
Table 7.3: CORRELATION COEFFICIENTS AFTER THIRD ITERATION

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
<th>COEFFICIENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \dot{x}^3 )</td>
<td>-0.8833</td>
<td>( \checkmark )</td>
<td>-19.7449</td>
</tr>
<tr>
<td>((\dot{x}^2)(\dot{x}))</td>
<td>-0.2870</td>
<td></td>
<td></td>
</tr>
<tr>
<td>((\dot{x})^2 \dot{x})</td>
<td>0.2871</td>
<td></td>
<td></td>
</tr>
<tr>
<td>((\ddot{x})^3)</td>
<td>-0.4168</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\ddot{x})</td>
<td>-0.4292</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\dddot{x})</td>
<td>-0.6449</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.4: CORRELATION COEFFICIENTS AFTER FOURTH ITERATION

<table>
<thead>
<tr>
<th>FUNCTION</th>
<th>C(d(t),f)</th>
<th>SELECTION</th>
<th>COEFFICIENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \dot{x}^3 )</td>
<td>-0.5052</td>
<td></td>
<td></td>
</tr>
<tr>
<td>((\dot{x}^2)(\dot{x}))</td>
<td>-0.4915</td>
<td></td>
<td></td>
</tr>
<tr>
<td>((\dot{x})^2 \dot{x})</td>
<td>0.5358</td>
<td>( \checkmark )</td>
<td>0.01463</td>
</tr>
<tr>
<td>((\ddot{x})^3)</td>
<td>-0.4461</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\ddot{x})</td>
<td>-0.5185</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(\dddot{x})</td>
<td>-0.1928</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The identified system model is given by Equation (7.15)

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} =
\begin{bmatrix}
0 & 1 \\
-1251.6165 & 0
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix} +
\begin{pmatrix}
0 \\
-19.7449x^3 + 0.050024(\dot{x})^2x
\end{pmatrix}
\]

(7.15)

The Equation (7.15) is numerically integrated using Runge Kutta fourth order method. The Figures (7.12-7.13) show the comparison between the position and velocity profiles of the identified system model and the true position and velocity. As can be seen the two are in close agreement. Hence the model representing the non-linear panel flutter problem is given by Equation (7.16).

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} =
\begin{bmatrix}
0 & 1 \\
-1251.6165 & 0
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix} +
\begin{pmatrix}
0 \\
-19.7449x^3 + 0.050024\dot{x}^2x
\end{pmatrix}
\]

(7.16)
Figure 7.12: MME Estimated position (+) and True position (solid)

Figure 7.13: MME Estimated velocity (+) and True velocity (solid)
7.5 System Identification with True Linear Model

7.5.1 Position and Velocity Measurement

As shown in Equation (7.7), the panel flutter system consists of a linear stiffness matrix $(K_S + K_A)$ and a linear damping matrix $(C_A)$ along with the nonlinear terms. These matrices may be used to determine the eigenvalue of the system and subsequently the natural frequency and the damping coefficient of the system can be evaluated. The linear model used for the MME is now modified to include the first natural frequency and the damping coefficient.

\[
A = \begin{bmatrix}
0 & I \\
-(K_S + K_A) & -C_A \\
\end{bmatrix}
\]  \hspace{1cm} (7.17)

From this linear model, the positive eigenvalue was obtained as $\lambda = 5.481988 \pm 34.248854i$. Hence the magnitude of the first eigenvalue and the first natural frequency is given as :

\[
\text{mag}(\lambda) = (5.481988)^2 + (34.248854)^2 = 1203.03 \quad (7.18)
\]

\[
\omega = \sqrt{1203.03} = 34.6844 \text{ rad/sec} \quad (7.19)
\]

and the damping coefficient is given as :

\[
\zeta = \frac{-5.481988}{34.6844} = -0.158 \quad (7.20)
\]

The new linear model can be written as:

\[
A = \begin{bmatrix}
0 & I \\
-(\omega)^2 & -2\zeta\omega \\
\end{bmatrix}
\]  \hspace{1cm} (7.21)
substituting the values of $\xi$ and $\omega$ it becomes

$$A = \begin{bmatrix} 0 & I \\ -1203.03 & 10.964 \end{bmatrix}$$  \quad (7.22)$$

The linear model used in MME maybe written as

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1203.03 & 10.964 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} d_1 \\ d_2 \end{bmatrix}$$  \quad (7.23)$$

Using MME the model error terms are evaluated and the nonlinear functions are evaluated along with its coefficients. The Table 7.5 shows the functions and the coefficients identified after each iteration. After the sixth iteration the identified system model is given by Equation (7.24)

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -1203.03 & 10.964 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \end{bmatrix} +$$
\[
\begin{pmatrix}
0 \\
-0.00303791(\dot{x})^3 - 35.2936x^3 + 0.034815(\dot{x})^2x - 2.2070299\dot{x}x^2
\end{pmatrix}
\] (7.24)

Equation (7.24) was numerically integrated using Runge Kutta fourth order method and Figures (7.14-7.15) show the comparison between the position and velocity profiles of the identified system model and the true position and velocity. As can be seen the two are in close agreement. Hence the model representing the non-linear panel flutter problem is given by Equation (7.24).
Figure 7.14: MME Estimated position (+) and True position (solid)

Figure 7.15: MME Estimated velocity (+) and True velocity (solid)
7.5.2 Acceleration Measurement

In this section it is considered that only acceleration data is available as the measured data. The MME is used to first identify the states and the Model Error and these are further used to identify the non-linear model form of the system. The linear model considered to start with is the same as in the previous section. The linear model form is:

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} = \begin{bmatrix} 0 & 1 \\ -1203.03 & 10.964 \end{bmatrix} \begin{pmatrix} x \\ \dot{x} \end{pmatrix} + \begin{pmatrix} d_1 \\ d_2 \end{pmatrix}
\]

(7.25)

Two cases are considered for the MME simulation. In the first case the acceleration data is used without any noise for the simulation and in the second case 10% noise is added to the acceleration signal. The acceleration data with noise used for the MME is shown in Figure (7.16). The program used for this data is in Appendix K. The Table 7.6 shows the function coefficients identified after each iteration when no noise was present in the signal and Table
Table 7.6: NONLINEAR FUNCTION COEFFICIENTS AFTER EVERY ITERATION WITHOUT NOISE IN THE SIGNAL

7.7 shows the corresponding data when 10% noise is added. In the case where no noise was present, the identified model after the sixth iteration is given by Equation (7.26)

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} =
\begin{bmatrix}
0 & 1 \\
-1203.03 & 10.964
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix} +
\begin{pmatrix}
0 \\
-0.002716(\dot{x})^3 - 37.9518x^3 + 0.02939(\dot{x})^2 x - 3.0006\dot{x}x^2
\end{pmatrix}
\]  
(7.26)

whereas when 10% noise was present the identified model is given by Equation (7.27).

\[
\begin{pmatrix}
\dot{x} \\
\ddot{x}
\end{pmatrix} =
\begin{bmatrix}
0 & 1 \\
-1203.03 & 10.964
\end{bmatrix}
\begin{pmatrix}
x \\
\dot{x}
\end{pmatrix} +
\begin{pmatrix}
0 \\
-0.002747(\dot{x})^3 - 35.014x^3 + 0.02704(\dot{x})^2 x - 3.0187\dot{x}x^2
\end{pmatrix}
\]  
(7.27)
It can be noticed that these Equations (7.26-7.27) are in close agreement with Equations (7.24). The Figures (7.17-7.19) show the states identified for the case where 10% noise is present in the signal. The true state (solid) denotes the true state of the system and non-linear model (+) denotes the states obtained by integrating the identified non-linear model in Equation (7.27). As can be seen from the figures that the states are in very close agreement and hence the mathematical model identified is a very good estimate of the non-linear system considered here.
Figure 7.17: Positions: True (solid) and Mathematical model (+)

Figure 7.18: Velocity: True (solid) and Mathematical model (+)
Figure 7.19: Acceleration: True with 10% noise (solid) and Mathematical model (+)

7.6 Conclusion

As can be seen from the results, both the models are able to identify the system to a good degree of accuracy. But the model identified using the true linear model in the MME makes more physical sense. Hence it is recommended to include all linear data in the assumed model in the MME in order to get a nonlinear model which makes more physical sense.
Chapter 8

Conclusions and Future work

8.1 Summary and Conclusions

The nonlinear identification technique has been presented in this thesis. Given time-domain discrete measurements and an assumed model, containing what is known of the system, the MME algorithm computes the model dynamic error. A correlation technique is used to find the functional form(s) that best fit the dynamic error and least squares finds the corresponding coefficients. If the newly found model still fails to match the measurements accurately, the procedure is iterated until an accurate model is found.

The method was tested with digital simulated data and with measurements with experiments. It has been shown that the identification technique requires little or no a priori knowledge of the form of system nonlinearity. No knowledge of state initial conditions is needed and no unusual restrictions are placed on measurement sampling frequency, data length or input functions. Sampling frequency should be high enough for the measurements to include information of the signals highest frequency content. The number of
measurements should be large enough to approximately show one trace of the largest period. The identification technique is robust for noisy measurements and regardless of the type of nonlinearity. The method is computationally efficient and easy to implement.

By employing the correlation technique a concise number of functions may be found to model the nonlinear dynamic error term. In small dimensional systems for which the nonlinear terms are derived analytically, the selected nonlinear functions are easily identified. When the acceleration was the only state that was available as the measurement the MME was able to converge to the exact states after a few iterations independent of the initial conditions and noise level. This goes to show the robustness of the method in nonlinear system identification.

8.2 Recommendations for Future work

The MME technique has been modified to enable it to identify the nonlinear system even when only acceleration data is available. Additional work could improve the nonlinear identification technique. First, a nonlinear type of multiple shooting would aid in the iterative search of multiple terms in the model error. In multiple shooting the solution is computed at each interval by means of the fundamental solution matrix, the value of which can be found only for the linear part of the problem. This limits the accuracy of the identification technique. For dynamic errors which are composed of more than one nonlinear functional form, the identified functions cannot be incorporated in the assumed model and a new model error calculated, unless MME is employed without the multiple shooting algorithm. In the examples shown in the present work, when more than one nonlinear error
term is found, each one is subtracted from the dynamic error term until the error is reduced to zero. Although this works well, integration is more accurate than subtraction.

An efficient way of identification of the function library is to be addressed. At present the more the functions the better because that would increase the probability of the true nonlinear function to be present in the library. If the library can be condensed to a few nonlinear functions then the computation time could be drastically reduced.

Also the weighting functions are arrived at by pure guess. Once the weighting function is guessed the weighting function is altered such that the covariance constraint is achieved. This is a very tedious process. An efficient algorithm to identify the weighting function could reduce the computation time drastically by reducing the number of iterations.
Bibliography


Appendix A

MME evaluation of SDOF System, Accel. Data Only

clear all
pack
%*****************************************************************************

% NUMERICAL INTEGRATION TO GET THE INPUT DATA FOR MME
%*****************************************************************************
deltat=0.2;
t=0:deltat:30;
u=[zeros(1,length(t));zeros(1,length(t))];
x0=[2;0];
x=vtb9_3('oned',u,t,x0); % VIBRATION TOOLBOX FOR RK
% NUMERICAL INTEGRATION
plot(t,x(1,:));
hold on
plot(t,x(2,:),'g');
n=max(size(x));
ykk=x(2,:);
%*****************************************************************************

% CALCULATION OF ACCELERATION FROM POSITION AND VELOCITY
%*****************************************************************************
for j=1:n;
    yk(j)=-x(1,j)-((x(1,j))^3)*0.5;
end
end

% RANDOM NUMBER GENERATION TO SIMULATE NOISE

R1=rand(size(yk));
R2=rand(size(yk));
rr1=mean(R1);
rrr1=R1-rr1;
sr=std(rrr1);
RAN1=rrr1*0.1/sr;

% load RAN1.mat % FOR 1% NOISE
% load RAN5.mat % FOR 5% NOISE
% load RAN10.mat % FOR 10% NOISE

yk=yk+RAN1;
A=[0 1;-1 0];
xdotrhs=[x(2,:);yk];
xrhs=x;

for i=1:n,
    dtrhs(:,i)=xdotrhs(:,i)-A*xrhs(:,i);
end

% MINIMUM MODEL ERROR AND STATES IDENTIFICATION ALGORITHM

AL=[0 1;-1 0];
RINV=0.0025;
nd=2;

wl=100000;
w2=0.0001;
D=[1 0 0 0;0 1 0 0;0 0 (1+(1/(RINV*wl))) 0;
   (2/RINV) 0 0 (1+(1/(RINV*w2)))];
C=[AL -0.5*[1/w1 0;0 1/w2];zeros(2,2) -AL'];

BO=[zeros(2,2) eye(2,2);zeros(2,2) zeros(2,2)];
BF=[zeros(2,2) zeros(2,2);zeros(2,2) eye(2,2)];
BETA=zeros(4,1);

% LEFT HAND SIDE OF THE EQUATION

LHS=eye(4*n,4*n);
vto=zeros(2*nd,2*nd);
for i=1:n,
    YJ(:,i)=[zeros(2,1);0;(2/RINV)*yk(i)];
end

% DO V2 FIRST

VOTMINUSONEPLUS=eye(4,4);
F1=deltat*(C*VOTMINUSONEPLUS);
F2=deltat*(C*(VOTMINUSONEPLUS+0.5*F1));
F3=deltat*(C*(VOTMINUSONEPLUS+0.5*F2));
F4=deltat*(C*(VOTMINUSONEPLUS+F3));
VOTOMINUS=VOTMINUSONEPLUS+(1/6)*(F1+2*F2+2*F3+F4);

BB1=BO*VOTOMINUS;
for i=1:4,
    for j=1:4,
        LHS(i,j)=BB1(i,j);
    end
end

BB2=-D*VOTOMINUS;
for i=1:4,
    for j=1:4,
        LHS(i+4,j)=BB2(i,j);
    end
end
V1TOPLUS=eye(4,4);

FF1=deltat*(C*V1TOPLUS);
FF2=deltat*(C*(V1TOPLUS+0.5*FF1));
FF3=deltat*(C*(V1TOPLUS+0.5*FF2));
FF4=deltat*(C*(V1TOPLUS+FF3));
V1T1MINUS=V1TOPLUS+(1/6)*(FF1+2*FF2+2*FF3+FF4);
DV1T1MINUS=-D*V1T1MINUS;

for hh=1:n-1;
    for i=1:4,
        for j=1:4;
            LHS(hh*4+i,hh*4+j)=V1TOPLUS(i,j);
        end
        if hh<n-1,
            LHS((hh+1)*4+i,hh*4+j)=DV1T1MINUS(i,j);
        end
        if hh==n-1,
            BFEND=BF*(-DV1T1MINUS);
            LHS(i,hh*4+j)=BFEND(i,j);
        end
    end
end

%RIGHT HAND SIDE OF THE EQUATION

RHS=zeros(4*n,1);
votminusoneplus=zeros(4,1);

f1=deltat*(C*votminusoneplus);
f2=deltat*(C*(votminusoneplus+0.5*f1));
f3=deltat*(C*(votminusoneplus+0.5*f2));
f4=deltat*(C*(votminusoneplus+f3));
votominus=votminusoneplus+(1/6)*(f1+2*f2+2*f3+f4);
vltoplus=zeros(4,1);

ff1=deltat*(C*vltoplus);
ff2=deltat*(C*(vltoplus+0.5*ff1));
ff3=deltat*(C*(vltoplus+0.5*ff2));
ff4=deltat*(C*(vltoplus+ff3));
v1t1minus=vltoplus+(1/6)*(ff1+2*ff2+2*ff3+ff4);

for hh=1:n;
    for i=1:4,
        if hh<n,
            RRRR=D*votominus-vltoplus+YJ(:,hh);
            RHS(hh*4+i,1)=RRRR(i,1);
        else
            RRR=BETA-BO*votominus-BF*D*votominus-BF*YJ(:,n);
            RHS(i,1)=RRR(i,1);
        end
    end
end
end

CC=LHS\RHS;

for j=0:n-1,
    DISP(1,j+1)=CC(2*nd*j+5,1);
    VEL(1,j+1)=CC(2*nd*j+6,1);
    LL(1,j+1)=-CC(2*nd*j+7,1)/(2*w1);
    LLL(1,j+1)=-CC(2*nd*j+8,1)/(2*w2);
    tt(j+1)=t(j+1);
end

for i=1:n,
    xdot1lhs1(i)=[1 0]*(AL*[DISP(1,i);VEL(1,i)]) + LL(1,i);
    xdot1lhs2(i)=[0 1]*(AL*[DISP(1,i);VEL(1,i)]) + LLL(1,i);
end

%----------------------------------------------------------
% COVARIANCE CONSTRAINT CHECK
%----------------------------------------------------------

ERROR=0;
ERROR1=0;
for i=100:150,
    ERROR=ERROR+(xdotlhs1(i)-VEL(1,i))^2;
    ERROR1=ERROR1+(xdotlhs2(i)-yk(i))^2;
end

COVARIANCE=ERROR/50
COVARIANCE1=ERROR1/50

plot(tt,DISP,'+')
hold on
plot(t,x(1,:),'w')
plot(tt,VEL,'+')
plot(t,x(2,:),'w')
plot(t,yk,'+')
plot(t,xdotlhs2,'*')

plot(t,LL,'+')
plot(t,dtrhs(1,:),'w')
plot(t,LLL,'+')
plot(t,dtrhs(2,:),'c')

clear LHS;
clear RHS;
clear YJ;
clear D;
clear C;

for i=120:150,
    yy(1,i-119)=DISP(i);
    yy(2,i-119)=VEL(i);
    dd1(i-119)=LLL(i);
    time(i-119)=t(i);
end

% LEAST SQUARES FIT ALGORITHM FOR NONLINEAR FUNCTION
% IDENTIFICATION AND COEFFICIENT EVALUATION.
n = max(size(yy));

% LIBRARY OF FUNCTIONS USED FOR IDENTIFICATION

for i = 1:n,
    y1(i) = (yy(1,i)^3);
    y2(i) = (yy(2,i)^2);
    y3(i) = yy(1,i) * yy(2,i);
    y4(i) = yy(1,i)^2;
end

ddbar = mean(dd1);
y1bar = mean(y1);
y2bar = mean(y2);
y3bar = mean(y3);
y4bar = mean(y4);

prod1 = 0;
prod2 = 0;
prod3 = 0;
prod4 = 0;
prod5 = 0;

for i = 1:n,
    prod1 = prod1 + ((dd1(i)-ddbar) * (y1(i)-y1bar));
    prod2 = prod2 + ((dd1(i)-ddbar) * (y2(i)-y2bar));
    prod3 = prod3 + ((dd1(i)-ddbar) * (y3(i)-y3bar));
    prod4 = prod4 + ((dd1(i)-ddbar) * (y4(i)-y4bar));
end

sigmadd = STD(dd1);
sigmay1 = STD(y1);
sigmay2 = STD(y2);
sigmay3 = STD(y3);
sigmay4 = STD(y4);

% CORRELATION COEFFICIENT EVALUATION

C1 = prod1 / (sigmadd * sigmay1 * i);
C2 = prod2 / (sigmadd * sigmay2 * i);
C3=prod3/(sigmadd*sigmay3*i);
C4=prod4/(sigmadd*sigmay4*i);

%FUNCTION COEFFICIENT EVALUATION

M=[y1'];
D=dd1';
P1=((inv(M*M))*(M')*D)
Appendix B

Subroutine for Numerical Integration of One Degree of Freedom System

% Program name : oned.m

function [zd]=rfunc(z,u,t)

zd=[z(2);-z(1)-((z(1))^3)*0.5];

end;
Appendix C

Numerical Integration of 2DOF System, 2 Nonlinearities

% Program name: twoda.m

function [zd]=rfunc(z,u,t)

zd=[z(3);z(4);-2*z(1)-(z(1)^3)/2+z(2)-0.1*z(1)*z(3)^2;z(1)-2*z(2)];

end;
Appendix D

MME Evaluation of Two Degree of Freedom System Accel Data Only

clear all
pack

%NUMERICAL INTEGRATION TO GET THE INPUT DATA FOR MME

deltat=0.2;
t=0:deltat:20;
u=[zeros(1,length(t));zeros(1,length(t))];
x0=[2;0;0;0];
x=vtb9_3('twod',u,t,x0);

%CALCULATION OF ACCELERATION FROM POSITION AND VELOCITY

for j=1:n;
yk(1,j)=-2*x(1,j)-0.2*(x(1,j))^3-0*x(3,j)*x(1,j)+x(2,j);
yk(2,j)=x(1,j)-2*x(2,j);
end
% RANDOM NUMBER GENERATION TO SIMULATE NOISE

R1=rand(size(yk(1,:)));
rr1=mean(R1);
rrr1=R1-rr1;
sr1=std(rrr1);
RAN1=rrr1*0.15/sr1;

R2=rand(size(yk(1,:)));
rr2=mean(R2);
rrr2=R2-rr2;
sr2=std(rrr2);
RAN2=rrr2*0.15/sr2;

% save RAN15 RAN1 RAN2
% load RAN5.m
% load RAN10.m
load RAN15.m

yk(1,:)=yk(1,:)+RAN1;
yk(2,:)=yk(2,:)+RAN2;

% MINIMUM MODEL ERROR AND STATES IDENTIFICATION ALGORITHM

A=[0 0 1 0; 0 0 0 1; -2 1 0 0; 1 -2 0 0];

xdotlhs=[x(3,:);x(4,:);yk(1,:);yk(2,:)];
xrhs=x;

for i=1:n,
    dtrhs(:,i)=xdotlhs(:,i)-A*xrhs(:,i);
end

w1=7000000;
w2=7000000;
w3=1;
w4=1;
nd=4;
\[C=[A -0.5*[1/w1 0 0 0; 1/w2 0 0 0; 0 0 1/w3 0; 0 0 0 1/w4];
   zeros(4,4) -A'];\]

\[RINV=0.0025;\]

\[MRINV1=[zeros(2,4); (4/RINV) (-2/RINV) 0 0;
   (-2/RINV) (4/RINV) 0 0];\]

\[MRINV2=[1 0 0 0; 0 1 0 0; 0 0 (1+(1/(RINV*w3))) 0;
   0 0 0 (1+(1/(RINV*w4)))];\]

\[D=[eye(nd,nd) zeros(nd,nd); MRINV1 MRINV2];\]

\[BO=[zeros(nd,nd) eye(nd,nd); zeros(nd,nd) zeros(nd,nd)];\]

\[BF=[zeros(nd,nd) zeros(nd,nd); zeros(nd,nd) eye(nd,nd)];\]

\[BETA=zeros(2*nd,1);\]

% LEFT HAND SIDE OF THE EQUATION

LHS=eye(2*nd*n,2*nd*n);

vto=zeros(2*nd,2*nd);

for \(i=1:n,\)
    \(YJ(:,i)=[zeros(4,1); 0; 0; (2/RINV)*yk(1,i); (2/RINV)*yk(2,i)];\)
end

VOTMINUSONEPLUS=eye(2*nd,2*nd);

F1=deltat*(C*VOTMINUSONEPLUS);

F2=deltat*(C*(VOTMINUSONEPLUS+0.5*F1));

F3=deltat*(C*(VOTMINUSONEPLUS+0.5*F2));

F4=deltat*(C*(VOTMINUSONEPLUS+F3));

VOTOMINUS=VOTMINUSONEPLUS+\(1/6\)*(F1+2*F2+2*F3+F4);

BB1=BO*VOTOMINUS;

for \(i=1:2*nd,\)
    for \(j=1:2*nd,\)
        LHS(i,j)=BB1(i,j);
    end
end

BB2=-D*VOTOMINUS;
for i=1:2*nd,
    for j=1:2*nd,
        LHS(i+2*nd,j)=BB2(i,j);
    end
end

V1TOPLUS=eye(2*nd,2*nd);
FF1=deltat*(C*V1TOPLUS);
FF2=deltat*(C*(V1TOPLUS+0.5*FF1));
FF3=deltat*(C*(V1TOPLUS+0.5*FF2));
FF4=deltat*(C*(V1TOPLUS+FF3));
V1T1MINUS=V1TOPLUS+(1/6)*(FF1+2*FF2+2*FF3+FF4);
DV1T1MINUS=-D*V1T1MINUS;

for hh=1:n-1;
    for i=1:2*nd,
        for j=1:2*nd;
            LHS((hh*2*nd)+i,(hh*2*nd)+j)=V1TOPLUS(i,j);
        end
    end
    if hh<n-1,
        LHS(((hh+1)*2*nd)+i,(hh*2*nd)+j)=DV1T1MINUS(i,j);
    end
    if hh==n-1,
        BFEND=BF*(-DV1T1MINUS);
        LHS(i,(hh*2*nd)+j)=BFEND(i,j);
    end
end

%#####################################
% RIGHT HAND SIDE OF THE EQUATION
%#####################################

RHS=zeros(2*nd*n,1);
votminusoneplus=zeros(2*nd,1);
f1 = \text{deltat} \times (C \times \text{votminusoneplus})
f2 = \text{deltat} \times (C \times (\text{votminusoneplus} + 0.5 \times f1))
f3 = \text{deltat} \times (C \times (\text{votminusoneplus} + 0.5 \times f2))
f4 = \text{deltat} \times (C \times (\text{votminusoneplus} + f3))

\text{votminus} = \text{votminusoneplus} + (1/6) \times (f1 + 2 \times f2 + 2 \times f3 + f4)

\text{v1toplus} = \text{zeros}(2 \times \text{nd}, 1)
ff1 = \text{deltat} \times (C \times \text{v1toplus})
ff2 = \text{deltat} \times (C \times (\text{v1toplus} + 0.5 \times ff1))
ff3 = \text{deltat} \times (C \times (\text{v1toplus} + 0.5 \times ff2))
ff4 = \text{deltat} \times (C \times (\text{v1toplus} + ff3))

\text{v1t1minus} = \text{v1toplus} + (1/6) \times (ff1 + 2 \times ff2 + 2 \times ff3 + ff4)

for hh = 1:n;
  for i = 1:2*nd,
    if hh < n,
      \text{RRRR} = D \times \text{v1t1minus} - \text{v1toplus} + YJ(:, hh);
      \text{RHS}((hh \times 2 \times \text{nd}) + i, 1) = \text{RRRR}(i, 1);
    else
      \text{RRR} = \text{BETA} - \text{BO} \times \text{votminus} - \text{BF} \times D \times \text{votminus} - \text{BF} \times YJ(:, n);
      \text{RHS}(i, 1) = \text{RRR}(i, 1);
    end
  end
end

\text{CC} = \text{LHS} \backslash \text{RHS};

for j = 0:n-2,
  \text{DISP1}(1, j+1) = \text{CC}(2 \times \text{nd} \times j + 9, 1);
  \text{DISP2}(1, j+1) = \text{CC}(2 \times \text{nd} \times j + 10, 1);
  \text{VEL1}(1, j+1) = \text{CC}(2 \times \text{nd} \times j + 11, 1);
  \text{VEL2}(1, j+1) = \text{CC}(2 \times \text{nd} \times j + 12, 1);
  \text{LL1}(1, j+1) = -\text{CC}(2 \times \text{nd} \times j + 13, 1) / (2 \times w1);
  \text{LL2}(1, j+1) = -\text{CC}(2 \times \text{nd} \times j + 14, 1) / (2 \times w2);
  \text{LLL1}(1, j+1) = -\text{CC}(2 \times \text{nd} \times j + 15, 1) / (2 \times w3);
  \text{LLL2}(1, j+1) = -\text{CC}(2 \times \text{nd} \times j + 16, 1) / (2 \times w4);
  \text{tt}(j) = t(j);
end

for i = 1:n,
  \text{xdotlhs}( :, i) = A \times [\text{DISP1}(1, i); \text{DISP2}(1, i); \text{VEL1}(1, i);
                  \text{VEL2}(1, i)] + [\text{LL1}(1, i); \text{LL2}(1, i); \text{LLL1}(1, i)];
end
LLL2(1,i)];
end

plot(tt,DISP1,’+’)
hold on
plot(t,x(1,:),’y’)

plot(tt,DISP2,’+’)
plot(t,x(2,:),’y’)

plot(tt,VEL1,’+’)
plot(t,x(3,:),’g’)

plot(tt,VEL2,’+’)
plot(t,x(4,:),’g’)

plot(t,LLL1,’+’)
plot(t,dtrhs(3,:),’m’)

plot(t,LLL2,’+’)
hold on
plot(t,dtrhs(4,:),’m’)

%################################################### ######### ######### ######## ######
% COVARIANCE CONSTRAINT CHECK
%################################################### ######### ######### ######## ######

ERROR=0;
ERROR1=0;
ERROR2=0;
ERROR3=0;

mm1=n;
mm2=1;
mm=mm1-mm2;

for i=mm2:mm1,
    ERROR=ERROR+(xdotlhs(1,i)-VEL1(1,i));
    ERROR1=ERROR1+(xdotlhs(2,i)-VEL2(1,i));
    ERROR2=ERROR2+(xdotlhs(3,i)-yk(1,i));
    ERROR3=ERROR3+(xdotlhs(4,i)-yk(2,i));
end
\texttt{A1=ERROR/mm;}
\texttt{A2=ERROR1/mm;}
\texttt{A3=ERROR2/mm;}
\texttt{A4=ERROR3/mm;}

\texttt{[A1 A2 A3 A4];}

\texttt{clear yy;}
\texttt{clear yyy;}

\texttt{hh=55;}

\texttt{for \ i=hh:n-5,}
\texttt{\indent yy(1,i-(hh-1))=DISP1(1,i);}
\texttt{\indent yy(2,i-(hh-1))=DISP2(1,i);}
\texttt{\indent yy(3,i-(hh-1))=VEL1(1,i);}
\texttt{\indent yy(4,i-(hh-1))=VEL2(1,i);}
\texttt{\indent dd1(1,i-(hh-1))=LLL1(i);}
\texttt{\indent dd2(1,i-(hh-1))=LLL2(i);}
\texttt{\indent time(i-(hh-1))=t(i);}
\texttt{end}

\texttt{n=max(size(yy));}

\texttt{\% LIBRARY OF FUNCTIONS USED FOR IDENTIFICATION}

\texttt{for \ i=1:n,}
\texttt{\indent y1(i)=(yy(1,i))^3;}
\texttt{\indent y2(i)=(yy(3,i))*yy(1,i);}
\texttt{\indent y3(i)=(yy(3,i)^3);}
\texttt{\indent y4(i)=(yy(3,i)^2);}
\texttt{\indent y5(i)=(yy(1,i))*(yy(3,i)^2);}
\texttt{\indent y6(i)=(yy(2,i)^3);}
\texttt{\indent y7(i)=(yy(4,i))*yy(2,i);}
\texttt{\indent y8(i)=(yy(4,i)^3);}
\texttt{\indent y9(i)=(yy(4,i)^2);}
\[
y_{10}(i) = (y_{y}(2,i)) \times (y_{y}(4,i)^2);
\]
end

ddbar1=mean(dd1);
ddbar2=mean(dd2);

y1bar=mean(y1);
y2bar=mean(y2);
y3bar=mean(y3);
y4bar=mean(y4);
y5bar=mean(y5);
y6bar=mean(y6);
y7bar=mean(y7);
y8bar=mean(y8);
y9bar=mean(y9);
y10bar=mean(y10);

prod1=0;
prod2=0;
prod3=0;
prod4=0;
prod5=0;
prod6=0;
prod7=0;
prod8=0;
prod9=0;
prod10=0;

for i=1:n,
    prod1=prod1+((dd1(i)-ddbar1) \times (y1(i)-y1bar)));
    prod2=prod2+((dd1(i)-ddbar1) \times (y2(i)-y2bar)));
    prod3=prod3+((dd1(i)-ddbar1) \times (y3(i)-y3bar)));
    prod4=prod4+((dd1(i)-ddbar1) \times (y4(i)-y4bar)));
    prod5=prod5+((dd1(i)-ddbar1) \times (y5(i)-y5bar)));
    prod6=prod6+((dd2(i)-ddbar2) \times (y6(i)-y6bar)));
    prod7=prod7+((dd2(i)-ddbar2) \times (y7(i)-y7bar)));
    prod8=prod8+((dd2(i)-ddbar2) \times (y8(i)-y8bar)));
    prod9=prod9+((dd2(i)-ddbar2) \times (y9(i)-y9bar)));
    prod10=prod10+((dd2(i)-ddbar2) \times (y10(i)-y10bar)));
end
sigmadd1=STD(dd1);
sigmadd2=STD(dd2);
sigmay1=STD(y1);
sigmay2=STD(y2);
sigmay3=STD(y3);
sigmay4=STD(y4);
sigmay5=STD(y5);
sigmay6=STD(y6);
sigmay7=STD(y7);
sigmay8=STD(y8);
sigmay9=STD(y9);
sigmay10=STD(y10);

%CORRELATION COEFFICIENT EVALUATION

C1=prod1/(sigmadd1*sigmay1*i);
C2=prod2/(sigmadd1*sigmay2*i);
C3=prod3/(sigmadd1*sigmay3*i);
C4=prod4/(sigmadd1*sigmay4*i);
C5=prod5/(sigmadd1*sigmay5*i);
C6=prod6/(sigmadd2*sigmay6*i);
C7=prod7/(sigmadd2*sigmay7*i);
C8=prod8/(sigmadd2*sigmay8*i);
C9=prod9/(sigmadd2*sigmay9*i);
C10=prod10/(sigmadd2*sigmay10*i);

FUNCTION COEFFICIENT EVALUATION

M1=[y1’ y2’ ];
D1=dd1’;
P1=((inv(M1’*M1)))*(M1’)*D1

M2=[y6’ y7’];
D2=dd2’;
P2=((inv(M2’*M2)))*(M2’)*D2
Appendix E

Numerical Integration of Nonlinear Two DOF System

% Program name : twod.m

function [zd]=rfunc(z,u,t)

zd=[z(3);z(4);-2*z(1)-0.2*(z(1))^3+z(2);z(1)-2*z(2)];

end
Appendix F

2DOF ID Code Using Position and Velocity

clear all
pack
%NUMERICAL INTEGRATION TO GET THE INPUT DATA FOR MME

deltat=0.1;
t=0:deltat:5;
u=[zeros(1,length(t));zeros(1,length(t))];
x0=[2;0;0;0];
x=vtb9_3(’twoda’,u,t,x0);
n=max(size(x));
%RANDOM NUMBER GENERATION TO SIMULATE NOISE

ykk=[x(1,:);x(2,:);x(3,:);x(4,:)];

R1=rand(size(x(1,:)));
rr1=mean(R1);
rrr1=R1-rr1;
sr1=std(rrr1);
%RAN1=rrr1*0.1/sr1;

R2=rand(size(x(1,:)));

173
\[
rr2=\text{mean}(R2);
rrr2=R2-rr2;
sr2=\text{std}(rrr2);
\]
\[
\% \ RAN2=rrr2*0.1/sr2;
R3=\text{rand}(\text{size}(x(1,:))) ;
rr3=\text{mean}(R3);
rrr3=R3-rr3;
sr3=\text{std}(rrr3);
\]
\[
\% \ RAN3=rrr3*0.1/sr3;
R4=\text{rand}(\text{size}(x(1,:))) ;
rr4=\text{mean}(R4);
rrr4=R4-rr4;
sr4=\text{std}(rrr4);
\]
\[
\% \ RAN4=rrr4*0.1/sr4;
\]
\[
\%\text{save RAN10B1 RAN1 RAN2 RAN3 RAN4}
\]
\[
\%\text{load RAN1b.mat}
\%\text{load RAN5b.mat}
\text{load RAN10b.mat}
\]
\[
ykk(1,:)=ykk(1,:)+\text{RAN1};
ykk(2,:)=ykk(2,:)+\text{RAN2};
ykk(3,:)=ykk(3,:)+\text{RAN3};
ykk(4,:)=ykk(4,:)+\text{RAN4};
\]
\[
\%\text{CALCULATION OF ACCELERATION FROM POSITION AND VELOCITY}
\]
\[
\%\text{for} \ j=1:n;
yk(1,j)=-2*ykk(1,j)-((ykk(1,j))^3)/2+ykk(2,j)-0.1*ykk(1,j)*ykk(3,j)^2;
yk(2,j)=ykk(1,j)-2*ykk(2,j);
\text{end}
\]
\[
\%\text{MINIMUM MODEL ERROR AND STATES IDENTIFICATION ALGORITHM}
\]
\[
A=[0 \ 0 \ 1 \ 0;0 \ 0 \ 0 \ 1;-2 \ 1 \ 0 \ 0;1 \ -2 \ 0 \ 0];
\]
\[
\text{xdotlhs}=[ykk(3,:);ykk(4,:);yk(1,:);yk(2,:)];
\]
xrhs=[ykk(1,:);ykk(2,:);ykk(3,:);ykk(4,:)];

for i=1:n,
dtrhs(:,i)=xdotlhs(:,i)-A*xrhs(:,i);
end

w1=0.01;
w2=0.001;
nd=4;
C=[A -0.5*[1/w1 0 0 0;0 1/w1 0 0;0 0 1/w2 0;0 0 0 1/w2];zeros(4,4) A'];

RINV=0.0025;
MRINV1=((-2/RINV)*[1 0 0 0;0 1 0 0;0 0 1 0;0 0 0 1]);
MRINV2=eye(4,4);
D=[eye(nd,nd) zeros(nd,nd);[MRINV1] [MRINV2]];
BO=[zeros(nd,nd) eye(nd,nd);zeros(nd,nd) zeros(nd,nd)];
BF=[zeros(nd,nd) zeros(nd,nd);zeros(nd,nd) eye(nd,nd)];
BETA=zeros(2*nd,1);

%############################### ######### ######### ######## ######
% LEFT HAND SIDE OF THE EQUATION
%############################### ######### ######### ######## ######

LHS=eye(2*nd*n,2*nd*n);
vto=zeros(2*nd,2*nd);

for i=1:n,
    YJ(:,i)=[zeros(4,1);(2/RINV)*ykk(1,i);(2/RINV)*ykk(2,i);
             (2/RINV)*ykk(3,i);(2/RINV)*ykk(4,i)];

end

% DO V2 FIRST

VOTMINUSONEPLUS=eye(2*nd,2*nd);
F1=deltat*(C*VOTMINUSONEPLUS);
F2=deltat*(C*(VOTMINUSONEPLUS+0.5*F1));
F3=deltat*(C*(VOTMINUSONEPLUS+0.5*F2));
F4=deltat*(C*(VOTMINUSONEPLUS+F3));
VOTOMINUS=VOTMINUSONEPLUS+(1/6)*(F1+2*F2+2*F3+F4);
BB1=BO*VOTOMINUS;
for i=1:2*nd,
    for j=1:2*nd,
        LHS(i,j)=BB1(i,j);
    end
end
BB2=-D*VOTOMINUS;

for i=1:2*nd,
    for j=1:2*nd,
        LHS(i+2*nd,j)=BB2(i,j);
    end
end
V1TOPLUS=eye(2*nd,2*nd);
FF1=deltat*(C*V1TOPLUS);
FF2=deltat*(C*(V1TOPLUS+0.5*FF1));
FF3=deltat*(C*(V1TOPLUS+0.5*FF2));
FF4=deltat*(C*(V1TOPLUS+FF3));
V1T1MINUS=V1TOPLUS+(1/6)*(FF1+2*FF2+2*FF3+FF4);
DV1T1MINUS=-D*V1T1MINUS;

for hh=1:n-1;
    for i=1:2*nd,
        for j=1:2*nd;
            LHS((hh*2*nd)+i,(hh*2*nd)+j)=V1TOPLUS(i,j);
        end
        if hh<n-1,
            LHS(((hh+1)*2*nd)+i,(hh*2*nd)+j)=DV1T1MINUS(i,j);
        end
        if hh==n-1,
            BFEND=BF*(-DV1T1MINUS);
            LHS(i,(hh*2*nd)+j)=BFEND(i,j);
        end
    end
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% RIGHT HAND SIDE OF THE EQUATION

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
RHS=zeros(2*nd*n,1);

votminusoneplus=zeros(2*nd,1);
f1=deltat*(C*votminusoneplus); 
f2=deltat*(C*(votminusoneplus+0.5*f1));
f3=deltat*(C*(votminusoneplus+0.5*f2));
f4=deltat*(C*(votminusoneplus+f3));
votominus=votminusoneplus+(1/6)*(f1+2*f2+2*f3+f4);
v1toplus=zeros(2*nd,1);

ff1=deltat*(C*v1toplus);
ff2=deltat*(C*(v1toplus+0.5*ff1));
ff3=deltat*(C*(v1toplus+0.5*ff2));
ff4=deltat*(C*(v1toplus+ff3));
v1t1minus=v1toplus+(1/6)*(ff1+2*ff2+2*ff3+ff4);

for hh=1:n;
    for i=1:2*nd,
        if hh<n,
            RRRR=D*v1t1minus-v1toplus+YJ(:,hh);
            RHS((hh*2*nd)+i,1)=RRRR(i,1);
        else
            RRR=BETA-BO*votominus-BF*D*votominus-BF*YJ(:,n);
            RHS(i,1)=RRR(i,1);
        end
    end
end

CC=LHS\RHS;

for j=0:n-2,
    DISP1(1,j+1)=CC(2*nd*j+9,1);
    DISP2(1,j+1)=CC(2*nd*j+10,1);
    VEL1(1,j+1)=CC(2*nd*j+11,1);
    VEL2(1,j+1)=CC(2*nd*j+12,1);
    LL1(1,j+1)=-CC(2*nd*j+13,1)/(2*w1);
    LL2(1,j+1)=-CC(2*nd*j+14,1)/(2*w1);
    LLL1(1,j+1)=-CC(2*nd*j+15,1)/(2*w2);
    LLL2(1,j+1)=-CC(2*nd*j+16,1)/(2*w2);
    tt(j+1)=t(j+1);
end

plot(tt,DISP1,’+’)
hold on
plot(t,x(1,:),'r')
plot(tt,DISP2,'+')
plot(t,x(2,:),'r')
plot(tt,VEL1,'+')
plot(t,x(3,:),'*')
plot(tt,VEL2,'+')
plot(t,x(4,:),'*')
plot(tt,LLL1,'+')
plot(t,dtrhs(3,:),'m')
plot(t,LLL2,'+')
plot(t,dtrhs(4,:),'m')

% COVARIANCE CONSTRAINT CHECK

ER1=0;ER2=0;ER3=0;ER4=0;
for i=15:n-15,
    ER1=ER1+(DISP1(i)-x(1,i));
    ER2=ER2+(DISP2(i)-x(2,i));
    ER3=ER3+(VEL1(i)-x(3,i));
    ER4=ER4+(VEL2(i)-x(4,i));
end
A1=ER1/i;
A2=ER2/i;
A3=ER3/i;
A4=ER4/i;

[A1 A2 A3 A4]
clear yy;

% COVARIANCE CONSTRAINT CHECK

for i=10:n-5,
    yy(1,i-9)=DISP1(1,i);
    yy(2,i-9)=DISP2(1,i);
yy(3,i-9)=VEL1(1,i);
yy(4,i-9)=VEL2(1,i);
dd1(1,i-9)=LLL1(i);
dd2(1,i-9)=LLL2(i);
time(i-9)=t(i);
end

% LEAST SQUARES FIT ALGORITHM FOR NONLINEAR FUNCTION IDENTIFICATION AND COEFFICIENT EVALUATION.

n=max(size(yy));
% LIBRARY OF FUNCTIONS USED FOR IDENTIFICATION
for i=1:n,
y1(i)=(yy(1,i)^3);
y2(i)=(yy(3,i)^2)*yy(1,i);
y3(i)=(yy(2,i)^3);
y4(i)=(yy(4,i)^2);
end

ddbar1=mean(dd1);
ddbar2=mean(dd2);
y1bar=mean(y1);
y2bar=mean(y2);
y3bar=mean(y3);
y4bar=mean(y4);

prod1=0;
prod2=0;
prod3=0;
prod4=0;

for i=1:n,
prod1=prod1+((dd1(i)-ddbar1).*(y1(i)-y1bar));
prod2=prod2+((dd1(i)-ddbar1).*(y2(i)-y2bar));
prod3=prod3+((dd2(i)-ddbar2).*(y3(i)-y3bar));
prod4=prod4+((dd2(i)-ddbar2).*(y4(i)-y4bar));
end
sigmadd1=STD(dd1);
sigmadd2=STD(dd2);
sigmay1=STD(y1);
sigmay2=STD(y2);
sigmay3=STD(y3);
sigmay4=STD(y4);

% CORRELATION COEFFICIENT EVALUATION

C1=prod1/(sigmadd1*sigmay1*i);
C2=prod2/(sigmadd1*sigmay2*i);
C3=prod3/(sigmadd2*sigmay3*i);
C4=prod4/(sigmadd2*sigmay4*i);

% FUNCTION COEFFICIENT EVALUATION

M1=[y1' y2'];
D1=dd1';
P1=((inv(M1'*M1))*(M1')*D1)

M2=[y3' y4'];
D2=dd2';
P2=((inv(M2'*M2))*(M2')*D2)
Appendix G

Program for Reducing the 64 Node Data to a SDOF Data

```matlab
% **************************** ********* ********* ******** *
% PROGRAM FOR REDUCING THE 64 NODE PLATE FLUTTER DATA
% TO AN EQUIVALENT SINGLE DEGREE OF FREEDOM SYSTEM
% **************************** ********* ********* ******** *

clear all
pack
% **************************** ********* ********* ******** *
% LOAD THE MAT FILE FROM THE FLUTTER SIMULATION
% **************************** ********* ********* ******** *
load m1v4a.mat

t=0.001:0.001:1;

for j=1:62,
    x(j,:)=Wff(j,:);
    vel(j,:)=Wff(j+62,:);
end

n=max(size(x));
% **************************** ********* ********* ******** *
% REDUCING TO GENERALIZED SINGLE DEGREE OF FREEDOM SYSTEM
```
for i=1:n,
    xx(i)=x(:,800)'*x(:,i);
    vvel(i)=x(:,800)'*vel(:,i);
    tt2(i)=t(i);
end

h=0.001;

for i=3:n-3
    xd(i)=(1/(2*h))*(xx(i+1)-xx(i-1))-(1/(12*h))*(xx(i+2)-2*xx(i+1)+2*xx(i-1)-xx(i-2));
    xdd(i)=(1/(2*h))*(vvel(i+1)-vvel(i-1))-(1/(12*h))*(vvel(i+2)-2*vvel(i+1)+2*vvel(i-1)-vvel(i-2));
    error(i)=xdd(i)+xx(i);
    ttt(i)=tt2(i);
end

for j=1:100;
    yy(1,j)=xx(700+j);
    yy(2,j)=vvel(700+j);
    dd1(j)=error(700+j);
    tt1(j)=tt2(700+j);
    acc(j)=xdd(700+j);
end

% SAVE THE FILE AS DATA.MAT

save data yy dd1 acc tt1
Appendix H

Program for Minimum Model Error Identification

```
% *********** *********** *********** 
% PROGRAM FOR MODEL ERROR IDENTIFICATION
% *********** *********** *********** 

clear all
pack
% *********** *********** *********** 
% LOAD THE SDOF DATA FROM DATA.MAT
% *********** *********** *********** 
load data.mat 
% *********** *********** *********** 
% ADD 10% NOISE
% *********** *********** *********** 
R=rand(size(yy(1,:)));
R1=rand(size(yy(2,:)));
rr=mean(R);
rr1=mean(R1);
rrr=R-rr;
rrr1=R1-rr1;
sr=std(rrr);
```
sr1=std(rrr1);
RAN1=rrr*0.1/sr;
RAN2=rrr*0.1/sr1;

% save RAN10 RAN1 RAN2
load RAN10.mat
yy(1,:)=yy(1,:)+RAN1;
yy(2,:)=yy(2,:)+RAN2;

% ****************************** ********* ********* ******** *****
AL=[0 1;-1 0];
RINV=0.0025;
nd=2;
deltat=0.001;
n=max(size(yy));
w1=0.0000001;
w2=0.00000002;

MM1=[-2*(1/RINV) 0;0 -2*(1/RINV)];
MM2=eye(2,2);

D=[eye(nd,nd) zeros(nd,nd);MM1 MM2];
C=[AL -0.5*[1/w1 0;0 1/w2];zeros(2,2) -AL'];

BO=[zeros(2,2) eye(2,2);zeros(2,2) zeros(2,2)];
BF=[zeros(2,2) zeros(2,2);zeros(2,2) eye(2,2)];
BETA=zeros(4,1);

% LHS
LHS=eye(4*n,4*n);
vto=zeros(2*nd,2*nd);

for i=1:n,
   YJ(:,i)=[zeros(2,1);(2/RINV)*yy(1,i);(2/RINV)*yy(2,i)];
end

% DO V2 FIRST
VOTMINUSONEPLUS=eye(4,4);
VOTOMINUS=deltat*(C*VOTMINUSONEPLUS)+VOTMINUSONEPLUS;
BB1=BO*VOTOMINUS;
for i=1:4,
    for j=1:4,
        LHS(i,j)=BB1(i,j);
    end
end
BB2=-D*VOTOMINUS;
for i=1:4,
    for j=1:4,
        LHS(i+4,j)=BB2(i,j);
    end
end
V1TOPLUS=eye(4,4);
V1T1MINUS=deltat*(C*V1TOPLUS)+V1TOPLUS;
DV1T1MINUS=-D*V1T1MINUS;
for hh=1:n-1;
    for i=1:4,
        for j=1:4;
            LHS(hh*4+i,hh*4+j)=V1TOPLUS(i,j);
        end
    if hh<n-1,
        LHS((hh+1)*4+i,hh*4+j)=DV1T1MINUS(i,j);
    end
    if hh==n-1,
        BFEND=BF*(-DV1T1MINUS);
        LHS(i,hh*4+j)=BFEND(i,j);
    end
end
end
% RHS
RHS=zeros(4*n,1);
votminusoneplus=zeros(4,1);
votominus = deltat * (C * votminusoneplus) + votminusoneplus;

vltoplus = zeros(4,1);
vltlminus = deltat * (C * vltoplus) + vltoplus;

for hh=1:n;
    for i=1:4,
        if hh<n,
            RRRR = D * votominus - vltoplus + YJ(:,hh);
            RHS(hh*4+i,1) = RRRR(i,1);
        else
            RRR = BETA - BO * votominus - BF * D * votominus - BF * YJ(:,n);
            RHS(i,1) = RRR(i,1);
        end
    end
end

CC = LHS \ RHS;

for j=0:n-1,
    DISP(1,j+1) = CC(2*nd*j+1,1);
    VEL(1,j+1) = CC(2*nd*j+2,1);
    LL(1,j+1) = -CC(2*nd*j+3,1)/(2*w1);
    LLL(1,j+1) = -CC(2*nd*j+4,1)/(2*w2);
end

for i=1:n-1;
    DISP1(i) = DISP(i+1);
    VEL1(i) = VEL(i+1);
    xxx(i) = yy(1,i);
    vvel1(i) = yy(2,i);
    LL1(i) = LL(i);
    LLL1(i) = LLL(i+1);
    dd11(i) = dd1(i);
    tt(i) = ttl1(i);
end

plot(tt,DISP1,’w’)
hold on
plot(tt,VEL1,’r’)
plot(tt,xxx,’y’)
plot(tt,vvvel,'g')
plot(tt,dd11,'r')
hold on
plot(tt,LLL1,'G')

%******************************************************************************
% CHECK THE COVARIANCE CONSTRAINT
%******************************************************************************
ERROR=0;
ERROR1=0;
ERROR2=0;
for i=1:n-1,
    ERROR=ERROR+abs(DISP1(1,i)-yy(1,i));
    ERROR1=ERROR1+abs(VEL1(1,i)-yy(2,i));
    ERROR2=ERROR2+abs(LLL1(i)-dd11(i));
end

ACCURACY=ERROR/i
ACCURACY1=ERROR1/i
ACCURACY2=ERROR2/i

y=yy;

%******************************************************************************
% SAVE THE MODEL ERROR AS DATA2.MAT
%******************************************************************************
save data2 DISP1 VEL1 tt LLL1 dd11 y
Appendix I

Program for Nonlinear Function and Coefficient Identification

```matlab
% ******************************************************************************************
% PROGRAM FOR NONLINEAR FUNCTION AND COEFFICIENT IDENTIFICATION
% ******************************************************************************************

clear all
pack
load data2.mat

yy(1,:)=y(1,:);
yy(2,:)=y(2,:);
n=max(size(yy));

for i=1:n-1,
    dd1(i)=LLL1(i)+1250.61652204 944*yy(1, i)- ...
        (0.03539363550581+0.01463446832972)*( yy(2,i)ˆ2 )*yy(1,i )+ ...
        (19.74495010398953)*(yy(1,i) )ˆ3;
    ttt(i)=tt(i);
end

% ******************************************************************************************
% FUNCTIONAL FORM IDENTIFICATION
% ******************************************************************************************

n=max(size(dd1));
for i=1:n,
    % ...
end
```

188
\begin{verbatim}
Y(1,i) = (yy(1,i))^3;
Y(2,i) = ((yy(1,i))^2) * yy(2,i);
Y(3,i) = (yy(2,i) * yy(1,i))^2);
Y(4,i) = (yy(2,i))^3;
Y(5,i) = yy(2,i);
Y(6,i) = yy(1,i);
end

for nn = 1:6,

ddbar = mean(dd1);
Ybar = mean(Y(nn,:));

prodl = 0;

for i = 1:n,
    prodl = prodl + ((dd1(i) - ddbar) .* (Y(nn,i) - Ybar));
end

sigmadd = STD(dd1);
sigmaY = STD(Y(nn,:));

C1(nn) = prodl / (sigmadd * sigmaY * i);

M = [Y(nn,:)'];
D = dd1';

P1(nn) = ((inv(M' * M)) * (M') * D);
end
\end{verbatim}
Appendix J

Code for Numerical Integration of the Identified Model

```matlab
% PROGRAM FOR NUMERICAL INTEGRATION OF THE IDENTIFIED MODEL

pack

deltat=0.002;
u=[zeros(1,length(t));zeros(1,length(t))];
x0=[1.0348239678272;53.4566154073770];
x=vtb9_3('checkrk',u,t,x0);
plot(t,x(1,:),'y');
hold on
plot(t,x(2,:),'r');

% SUBROUTINE FOR NUMERICAL INTEGRATION OF THE IDENTIFIED MODEL

function [zd]=rfunc(z,u,t)

zd=[z(2);-z(1)-1250.61652204944*z(1)+...
(0.03539363550581+0.01463446832972)*(z(2)^2)*z(1)-...
(19.74495010398953)*(z(1))^3];
end;
```
Appendix K

MME Code for Panel Flutter, Acceleration Data Only

```matlab
% % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
% PROGRAM FOR MME OF THE FLUTTER PROBLEM WHEN
% ONLY ACCELERATION DATA IS AVAILABLE
% % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % % %
clear all
pack
load data.mat

yk1=yyy;
err=dda;
t=ta;
deltat=t(2)-t(1);
n=max(size(yyy));

R1=rand(size(yk1));
rr1=mean(R1);
rrr1=R1-rr1;
sr=std(rrr1);
RAN1=rrr1*2460*0.1/sr;
%save RAN10a RAN1
load RAN10a.mat
yk=yk1+RAN1;
AL=[0 1;-1203.03 10.964];

RINV=0.0025;
```
\[ D = \begin{bmatrix} 1 & 0 & 0 & 0; 0 & 1 & 0 & 0; 0 & 0 & \left(1+\frac{1}{RINV \cdot w1}\right) & 0; \\
(2*1203.03/RINV) & -2*10.964/RINV & 0 & \left(1+\frac{1}{RINV \cdot w2}\right) \end{bmatrix}; \]

\[ C = \begin{bmatrix} AL & -0.5 \cdot \begin{bmatrix} 1/w1 & 0; 0 & 1/w2 \end{bmatrix}; zeros(2,2) & -AL' \end{bmatrix}; \]

\[ BO = \begin{bmatrix} zeros(2,2) & eye(2,2); zeros(2,2) & zeros(2,2) \end{bmatrix}; \]

\[ BF = \begin{bmatrix} zeros(2,2) & zeros(2,2); zeros(2,2) & eye(2,2) \end{bmatrix}; \]

\[ \text{LHS} \]

\[ \text{LHS} = \text{eye}(4 \cdot n, 4 \cdot n); \]

\[ \text{vto} = \text{zeros}(2 \cdot nd, 2 \cdot nd); \]

\[
\text{for } i=1:n, \\
YJ(:,i) = \begin{bmatrix} \text{zeros}(2,1); 0; \left(2/RINV\right) \cdot y_k(i) \end{bmatrix}; \\
\text{end}
\]

\% DO V2 FIRST

\[ \text{VOTMINUSONEPLUS} = \text{eye}(4,4); \]

\% \[ \text{VOTOMINUS} = \text{deltat} \cdot \left( C \cdot \text{VOTMINUSONEPLUS}\right) + \text{VOTMINUSONEPLUS}; \]

\[ \text{F1} = \text{deltat} \cdot \left( C \cdot \text{VOTMINUSONEPLUS}\right); \]

\[ \text{F2} = \text{deltat} \cdot \left( C \cdot \left( \text{VOTMINUSONEPLUS} + 0.5 \cdot \text{F1}\right)\right); \]

\[ \text{F3} = \text{deltat} \cdot \left( C \cdot \left( \text{VOTMINUSONEPLUS} + 0.5 \cdot \text{F2}\right)\right); \]

\[ \text{F4} = \text{deltat} \cdot \left( C \cdot \left( \text{VOTMINUSONEPLUS} + \text{F3}\right)\right); \]

\[ \text{VOTOMINUS} = \text{VOTMINUSONEPLUS} + \left(1/6\right) \cdot \left( \text{F1} + 2 \cdot \text{F2} + 2 \cdot \text{F3} + \text{F4}\right); \]

\[ \text{BB1} = \text{BO} \cdot \text{VOTOMINUS}; \]

\[
\text{for } i=1:4, \\
\text{for } j=1:4, \\
\text{LHS}(i,j) = \text{BB1}(i,j); \\
\text{end} \\
\text{end}
\]
BB2=-D*VOTOMINUS;

for i=1:4,
for j=1:4,
LHS(i+4,j)=BB2(i,j);
end
end

FF1=deltat*(C*V1TOPLUS);
FF2=deltat*(C*(V1TOPLUS+0.5*FF1));
FF3=deltat*(C*(V1TOPLUS+0.5*FF2));
FF4=deltat*(C*(V1TOPLUS+FF3));
V1T1MINUS=V1TOPLUS+(1/6)*(FF1+2*FF2+2*FF3+FF4);

DV1T1MINUS=-D*V1T1MINUS;

for hh=1:n-1;
for i=1:4,
for j=1:4;
LHS(hh*4+i,hh*4+j)=V1TOPLUS(i,j);
if hh<n-1,
LHS((hh+1)*4+i,hh*4+j)=DV1T1MINUS(i,j);
end
if hh==n-1,
BFEND=BF*(-DV1T1MINUS);
LHS(i,hh*4+j)=BFEND(i,j);
end
end
end

% RHS
RHS=zeros(4*n,1);

votminusoneplus=zeros(4,1);

votominus=deltat*(C*votminusoneplus)+votminusoneplus;

f1=deltat*(C*votminusoneplus);
f2=deltat*(C*(votminusoneplus+0.5*f1));
f3=deltat*(C*(votminusoneplus+0.5*f2));
f4=deltat*(C*(votminusoneplus+f3));
votominus=votminusoneplus+(1/6)*(f1+2*f2+2*f3+f4);

v1toplus=zeros(4,1);

v1t1minus=deltat*(C*v1toplus)+v1toplus;

ff1=deltat*(C*v1toplus);
ff2=deltat*(C*(v1toplus+0.5*ff1));
ff3=deltat*(C*(v1toplus+0.5*ff2));
ff4=deltat*(C*(v1toplus+ff3));
v1t1minus=v1toplus+(1/6)*(ff1+2*ff2+2*ff3+ff4);

for hh=1:n;
  for i=1:4,
    if hh<n,
      RRRR=D*votominus-v1toplus+YJ(:,hh);
      RHS(hh*4+i,1)=RRRR(i,1);
    else
      EE=zeros(2*nd,1);
      RRR=BETA-BO*votominus-BF*D*votominus-BF*YJ(:,n);
      RHS(i,1)=RRR(i,1);
    end
  end
end

end

end

CC=LHS\RHS;

for j=0:n-1,
  DISP(1,j+1)=CC(2*nd*j+1,1);
  VEL(1,j+1)=CC(2*nd*j+2,1);
  LL(1,j+1)=-CC(2*nd*j+3,1)/(2*w1);
  LLL(1,j+1)=-CC(2*nd*j+4,1)/(2*w2);
end

for i=1:n,
    xdot1hs1(i)=[1 0]*(AL*[DISP(1,i);VEL(1,i)]) + LL(1,i);
    xdot1hs2(i)=[0 1]*(AL*[DISP(1,i);VEL(1,i)]) + LLL(1,i);
end

ERROR=0;
ERROR1=0;
for i=98:149,
    ERROR=ERROR+(xdot1hs1(i)-VEL(1,i));
    ERROR1=ERROR1+(xdot1hs2(i+1)-yk(i));
end

ACCURACY=ERROR/50
ACCURACY1=ERROR1/50

for i=1:n-1;
    DISP1(i)=DISP(i+1);
    VEL1(i)=VEL(i+1);
    LLLA(i)=yk(i)+1203.03*DISP1(i)-10.964*VEL1(i);
    LLLAA(i)=LLL(i+1);
    xdot1hs2a(i)=xdot1hs2(i+1);
    tt(i)=t(i);
    ykk(i)=yk(i);
end

plot(t,LLL,'m')
hold on
plot(ta,dda)
plot(tt,LLLAA,'c')
save mmedata LLLAA DISP1 VEL1 ykk tt