NASA Contractor Report 170412

NASA-CR-170412
19840012444

Real-Time Flutter Analysis

Robert Walker and Naren Gupta

Contract NAS4-2955
March 1984

National Aeronautics and
Space Administration

NASA Contractor Report 170412

Real-Time Flutter Analysis

Robert Walker and Naren Gupta
Integrated Systems, Inc., 151 University Avenue, Suite 400, Palo Alto, California 94301

Prepared for
Ames Research Center
Dryden Flight Research Facility
Edwards, California
under Contract NAS4-2955

1984

National Aeronautics and
Space Administration
Ames Research Center
Dryden Flight Research Facility
Edwards, California 93523

FOREWARD

The report documents techniques and a FORTRAN 77 computer program for the identification of flutter-mode frequencies and damping ratios in near-real-time. Recent advances in recursive identification algorithms and analysis are applied to the flutter test problem. The accuracy and convergence of the algorithms are predicted analytically and substantiated with results from simulated and flight data. The results show promise for monitoring aircraft flutter characteristics in real-time with a high degree of reliability.

The work has been performed under NASA Contract NAS4-2955. Technical direction and discussions with NASA technical monitor Mr. Glenn Gilyard and with Mr. Richard Maine are gratefully acknowledged.

(805) 258-3724
work in AI Centers Shop

TABLE OF CONTENTS

SECTION 1

INTRODUCTION

Real-time monitoring of flutter parameters can significantly improve safety of flight tests for new and modified aircraft, reduce flight test time, and aid envelope expansion. The monitoring system must be reliable and accurate and should require minimum inputs from the operator. To aid flutter clearance of many aircraft, it is desirable to monitor flutter characteristics with randomly forced (turbulence excitation) as well as known forcing inputs. The most success is likely to be achieved with the use of advanced system identification techniques. For widespread use, the computation time should be suitable for a real-time implementation and a standard computer language like the ANSI FORTRAN-77 is desirable.

The goal of this effort is to develop a computer program--based on an identification algorithm--with excellent accuracy and convergence performance in estimating flutter mode characteristics of test aircraft from real data.

1.1 SUMMARY OF APPROACH

A summary of the approach used to develop and document the near-real-time flutter analysis program is shown in Figure 1-1.

The performance of two different algorithms was evaluated with a high-level command-driven identification and control program called MATRIX_X [1]. This capability enabled a thorough yet efficient evaluation of the algorithms before implementation and testing of the Near-Real-Time Flutter Analysis (NRTFA) program in FORTRAN-77 source code.

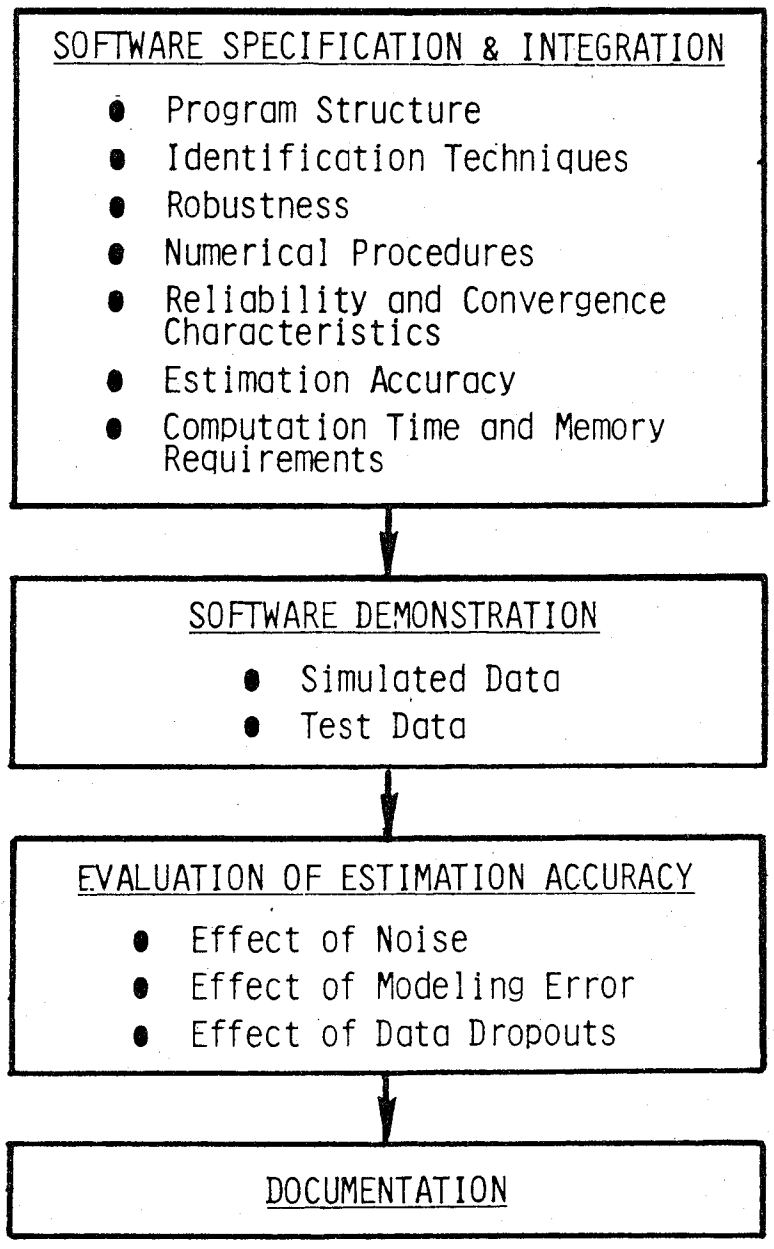


Figure 1-1 Real-Time Flutter Analysis Approach

1.2 RESULTS SUMMARY

The selected techniques are based on recursive prediction error methods (RPEM), which can emulate a maximum likelihood, instrumental variables and least squares.

The Cramer-Rao estimate error accuracy bounds for a two-active mode model predict that the critical parameter, damping, can be estimated quite satisfactorily after approximately 500 data points. Actual simulations indicate that the convergence transient from no *a priori* information lasts slightly longer, but that for long data records of 2500 data points, damping ratios can be estimated very accurately (<.05% error). Modal parameters can be estimated from unknown inputs (turbulence excitation) with sufficient accuracy when the signal-to-noise ratio is approximately ten or greater. For lower turbulence levels almost any known input, even at very low levels, significantly improves the estimation accuracy.

The delivered algorithm tracks the flutter mode of the third DAST flight quite successfully, indicating where the damping trend breaks toward zero at the onset of a pulse response decay, prior to the flutter condition. Quantitative comparisons of the convergence, robustness and accuracy of the recursive identification algorithm analysis and simulation results show the near-real-time flutter analysis (NRTFA) program to be highly reliable.

1.3 REPORT ORGANIZATION

Candidate algorithms and the associated model forms most suitable for use with them in flutter testing are described in Section 2. The reliability, robustness and accuracy of these flutter mode estimates are evaluated in Section 3, supported by results on simulated and flight data in Sections 4 and 5. The program architecture, configured with an interactive driver which calls the appropriate identification subroutines, with an instal-

lation guide is documented in Section 6. The memory and CPU requirements for the flutter analysis program as it is now configured are also given. Appendix A serves as a brief User's Guide giving an example interactive setup session, with guidelines for parameter setup options, and a description of the modal parameter estimate output. Appendix B gives the FORTRAN listings of the principal routines of the NRTFA program.

SECTION 2

REQUIREMENTS FOR REAL-TIME FLUTTER ANALYSIS

The development of a system identification software package to monitor aircraft flutter characteristics is influenced by the following considerations:

1. The basic requirements in real-time flutter analysis involves estimation of damping ratios and changes in natural frequencies of aeroelastic modes. Mode shapes are not required but may be determined if they improve damping ratio and natural frequency estimates. Physical parameters are usually not estimated in real-time.

2. Several flutter modes must be tracked simultaneously. The modes can be closely spaced. For successful real-time implementation, computational efficiency is very important.

3. The real-time software must be robust and reliable because its failure could cause delay or termination of a flight test.

4. Since the flight test engineer is generally busy with several tasks during the flight test, the software should require minimum commands during the test. Set-up parameters, if any, should be entered prior to the flight test, and the algorithm should be able to restart without entering any startup values.

5. Because of the damping ratios of flutter modes are required with high precision and the data is often noisy, advanced identification techniques are highly desirable whose convergence properties and accuracy performance can be analyzed.

6. The estimated parameters must be updated at regular intervals to continuously track any changes in flutter characteristics. Thus, a recursive technique is desirable.

The following characteristics of a real-time flutter-monitoring software package determine the extent to which the above requirements are met.

1. model representations,
2. identification procedures and robustness characteristics, and
3. numerical techniques, program architecture and input/output structure.

These issues are interrelated since identification procedures and numerical techniques depend on the model form. The model form affects the shape of the criterion function we seek to minimize with respect to the parameters, as well as the type of equations which must be propagated to find the gradients. This section explains why certain model forms and algorithms are preferred for real-time flutter analysis. The identification algorithms are broadly grouped as recursive and block recursive algorithms. Robustness characteristics, numerical techniques and a suitable program architecture are discussed in subsequent sections.

2.1 Model Forms

Three model forms can be used:

1. transfer function or continuous autoregressive moving average (ARMA),
2. sample-data transfer function or discrete ARMA,
3. state variables in continuous or discrete forms.

2.1.1 Transfer Function

The q x 1 input u to p x 1 output y transfer function is

$$\frac{y(s)}{u(s)} = T(s) , \qquad (2.1)$$

which may be written as

$$T(s) = \frac{N(s)}{D(s)} \tag{2.2}$$

$$= \sum_{i=1}^{n} \frac{R_i}{(s + \lambda_i)} \tag{2.3}$$

$N(s)$ is a matrix polynomial and $D(s)$ is a scalar polynomial in s. R_i are $p \times q$ matrices and λ_i are the pole functions.

For scalar input and output, the transfer function is also written as

$$T(s) = K \sum_{i=1}^{m} (s + z_i) \sum_{i=1} (s + \lambda_i) , \tag{2.4}$$

where z_i are the zeros and K is the gain. A continuous autoregressive moving-average form is

$$y^{(n)} + \sum_{i=1}^{n} \alpha_i \, y^{(n-1)} = \sum_{i=0}^{m} B_i \, u^{(m-1)} , \tag{2.5}$$

where $y^{(j)}$ is the jth derivative of y.

2.1.2 Sampled Data Transfer Function

This transfer function and ARMA representation is similar to the continuous transfer function except a z-transform is used in place of the s-transform. Thus

$$y(z^{-1}) = T(z^{-1}) \, u(z^{-1}) \tag{2.6}$$

where

$$z^{-1} y_k = y_{k-1} \tag{2.7}$$

This model can be written in any of the forms corresponding to Equations (2.3) to (2.5).

The representation maps the imaginary axis into the unit circle. Lightly damped oscillatory continuous modes will be close to the unit circle in the ARMA formulation, depending on how far the continuous roots are into the left half plane and the sampling time. The continuous damped frequency and sample time determine the angular orientation of the poles in the z-plane,

$$\theta = \mathrm{Tan}^{-1}\omega_n(1 - \xi^2)^{\frac{1}{2}}\,\tau \quad .$$

The different discrete polynomial models for both known and unknown inputs and their state-space realizations are discussed more fully in the next subsection on recursive algorithms, where they are particularly attractive.

2.1.3 State Variable Models

Defining the state vector as x, a state variable model is written as

$$\dot{x} = Fx + Gu + \Gamma w \quad , \tag{2.8}$$

$$y = Hx + v \quad .$$

w is the process noise vector and v is the measurement noise vector. F, G and H matrices can take various forms depending on the particular selection of the state vector. In many offline applications, the state vector is selected to represent a set of physical quantities (e.g., body rates, deflection at a certain point on the wing). Such representations give rise to physical parameters, such as stability and control coefficients.

In real-time flutter analysis, the natural frequencies and damping ratios of aeroelastic modes are of primary interest. Since a state variable model [F, G, H] has the same input-output behavior as a [TFT^{-1}, TG, HT^{-1}] model if

T is a nonsingular matrix, the following representation appears useful (for single-input systems):

$$F = \begin{bmatrix} B_1 & & & \\ & B_2 & & \\ & & \ddots & \\ & & & B_n \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{2.10}$$

$$B_i = \begin{bmatrix} 0 & 1 \\ -\omega_{n_i}^2 & -2\xi_i\mu_{n_i} \end{bmatrix}, \quad \text{or} \quad \begin{bmatrix} \sigma & \omega \\ -\omega & \sigma \end{bmatrix}, \tag{2.11}$$

and H is a general matrix with potentially all nonzero elements. With multi-input systems, the remaining part of matrix G is general and unknown. Γ is a general matrix.

The turbulence excited structural vibrations are of special interest. In this case $G = 0$. For the purpose of identification, the Kalman filter representation is desirable

$$\begin{aligned} \dot{\hat{x}} &= F\hat{x} + Gu + K(y - H\hat{x}) \\ y &= H\hat{x} + v \end{aligned} \tag{2.12}$$

$[F,G,H,\Gamma]$ have been mapped into $[F,G,H,K]$. It is usually easier to identify K rather than Γ, and in general requires fewer parameters.

In the state variable form, the natural frequencies and damping ratios of all modes are estimated directly. The measurements are, however, nonlinear functions of parameters leading to a more difficult convergence problem.

The propagation of the linear equation, as well as the computation of the gradients of the innovations with respect to parameters are much more easily accomplished with a discrete space model of the form

$$x_{k+1} = Fx_k + Gu_k + \Gamma w_k$$

$$y_k = Hx_k + Du_k + v_k$$

where w_k and v_k are independent, identically distributed sequences with zero mean. If the time variation of the estimator during the transient phase is important or there is a priori structural knowledge about the noise covariance matrices that considerably reduce the number of parameters below pn, the dimension the Kalman gain matrix K (where p is the number of outputs and n the number of states), then the general equations above should be considered. In real time flutter identification, the Γ matrix generally will not be known, nor the structure of the noise covariances, making the innovation form a much more desirable set of eqautions. They are given by

$$x_{k+1} = Fx_k + Gu_k + Kv_k$$

$$y_k = Hx_k + Du_k + v_k$$

where $\{v_k\}$ is a sequence of independent variables with zero mean. Moreover, the model generating $\hat{y}$ from y, i.e., $\hat{y} = H(zI-F+K\ H)^{-1}y$, is stable (eigenvalues of F-KH are in the unit circle). In the next subsection, we will see that ARMA type-models have direct realizations as a state-space innovations form.

The discrete controller and real Jordan block form, analogous to the continuous block forms described previously, both keep a simple parameterization of an oscillatory mode; however,

the real Jordan block form is a preferable canonical form for identification of physical systems, generally giving a better conditioned optimization problem as described below.

Consider a single mode (a general system is a linear combination of such modes) of a continuous system

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_n^2 & -2\xi\omega_n \end{bmatrix} \begin{bmatrix} x \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u,$$

$$y = [h_1 \quad h_2] \begin{bmatrix} x \\ v \end{bmatrix} .$$

The input is an acceleration forcing the velocity equation, with position the integral of velocity. The discrete form of this equation (for a pure oscillator using the simplest case) is

$$\begin{bmatrix} x_{k+1} \\ v_{k+1} \end{bmatrix} = \begin{bmatrix} 1 & \frac{\sin(\omega_n \Delta)}{\omega_n} \\ -\frac{\sin(\omega_n \Delta)}{\omega_n} & \frac{1}{\omega_n^2} \end{bmatrix} \begin{bmatrix} x_k \\ v_k \end{bmatrix} + \begin{bmatrix} -\frac{\omega_n^2 - 1}{\omega_n^4} \\ \frac{\sin(\omega_n \Delta)}{\omega_n} \end{bmatrix} u_k$$

$$y_k = [h_1 \quad h_2] \begin{bmatrix} x_k \\ y_k \end{bmatrix}$$

If time is normalized such that for the primary mode of interest, $\omega_n = 1$, then the above discrete equations are in real Jordan form. Consider, in addition, the identification of the mode in both the block real Jordan form and the block controller forms of Equation (2.10). The two equations are

$$x_{k+1} = Jx_k + G_m u_k \quad , \qquad \tilde{x}_{k+1} = F_c \tilde{x}_k + G_c u_k$$

$$y_k = H_m x_k \quad , \qquad y_k = \tilde{H} \tilde{x}_k \quad ,$$

where in both cases the H matrices have been changed to fix G at $[0\ 1]^T$, and conserve the residues. For our one mode example the parameter vectors are

$$\theta \begin{bmatrix} h_{m_1} \\ h_{m_2} \\ \sigma \\ \omega \end{bmatrix} \quad \text{and} \quad \tilde{\theta} = \begin{bmatrix} h_1 \\ h_2 \\ -b \\ -a \end{bmatrix} \quad \text{for} \quad \begin{bmatrix} 0 & 1 \\ -b & -a \end{bmatrix} .$$

The gradients of the estimate with respect to these two different parameterizations expressed in the real Jordan form state variables are

$$\frac{\partial \hat{y}}{\partial \theta} = T_m [x_1, x_2, x_1(-), x_2(-)]^T \ ,$$

and

$$\frac{\partial \hat{y}}{\partial \tilde{\theta}} = \tilde{T} [x_1, x_2, x_1(-), x_2(-)]^T \ ,$$

where

$$T_m = \begin{bmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} & \\ & \begin{pmatrix} h_{m_1} & h_{m_2} \\ h_{m_2} & -h_{m_1} \end{pmatrix} \end{bmatrix} , \text{and} \quad \tilde{T} = \begin{bmatrix} \begin{pmatrix} 1 & 0 \\ \sigma & \omega \end{pmatrix} \sqrt{2} & \\ & \frac{-h_{m_2}}{\omega} \begin{pmatrix} 1 & 0 \\ \sigma & \omega \end{pmatrix} \end{bmatrix} .$$

The Hessian, or information matrix is approximated by the outer product of these two gradients. If

$$\phi = \begin{bmatrix} x_1(k) \\ x_2(k) \\ x_1(k-1) \\ x_2(k-1) \end{bmatrix} \qquad M = \sum_{k=1}^{N} \phi \, \phi^T \quad ,$$

then

$$M_m = T_m M T_m^T \quad , \quad \text{and} \quad \tilde{M} = \tilde{T} \, M \, \tilde{T}^T \quad .$$

In other words the transformations, T_m and $\tilde{T}$ are directly effecting the condition of the estimation problem. Since we are referencing both to the real Jordan form, $\kappa_2^{\dagger}(T_m) = 1$, or it is perfectly conditioned with respect to its own coordinate system whereas $\kappa_2(\tilde{T})$ can be quite large. It was pointed out above that continuous modes driven directly by accelerating forces, give discrete models that are closer to being in real Jordan form, hence transformation to the block controller form can only make the numerical conditioning of the information matrix worse. Of course in high order physical systems the phase relationships between modes can be such that this preference for the real Jordan form over the block controller form would no longer hold. However, the primary modes in flutter testing are expected to sufficiently follow the phase behavior of a simple oscillatory mode, that the real Jordan form will be used.

Having described different model forms and reasons for preferred ones, the next two subsections describe the algorithms used to identify the parameters of these model sets.

---

†The condition of a matrix A with respect to any norm x is $\kappa_x(A) = \| A \|_x \| A^{-1} \|_x$.

2.2 Recursive Identification Algorithms

Recursive identification refers to identification algorithms that update parameters at every sampling instant. Such algorithms are characterized by finite non-increasing storage requirements. Typically they are well-suited for on-line, real-time identification on computers of modest size.

For single-input single-output systems in linear finite dimensional form, the AutoRegressive Moving Average with eXogenous inputs (ARMAX) models are popular. They are represented with the difference equations:

$$(1+a_1z^{-1}+a_2z^{-2}\ldots+a_{NA}z^{-NA})y_{k+1} = (b_0+b_1z^{-1}\ldots+b_{(NB-1)}z^{-(NB-1)})u_k + (1+c_1z^{-1}+c_2z^{-2}\ \ldots+c_{NC}z^{-NC})e_{k+1}$$

or

$$A(z^{-1})y_{k+1} = B(z^{-1})u_k + C(q^{-1})e_{k+1}$$

where

$y_k, y_{k+1}, \ldots$ are the outputs

$u_k, u_{k-1}, \ldots$ are the exogenous inputs

$e_k, e_{k-1}, \ldots$ are the innovations or white noise.

z^{-1} represents the unit delay operator, i.e., $z^{-1}y_k = y_{k-1}$

Inputs and outputs are observed but the innovations sequence is not observed. In the transfer function language,

$$y(z^{-1}) = z^{-1}\frac{B(z^{-1})}{A(z^{-1})}u(z^{-1}) + \frac{C(z^{-1})}{A(z^{-1})}v(z^{-1})$$

$C(z^{-1})$ always represents a stable polynomial, i.e., all its roots are inside the unit circle. In general $B(z^{-1})$ and $A(z^{-1})$ are not stable and not comprime. A general black-box

model

$$F(z^{-1})Y(z^{-1}) = \frac{B(z^{-1})}{A(z^{-1})} u(z^{-1}) + \frac{C(z^{-1})}{D(z^{-1})} e(z^{-1})$$

can also be used but they have not been very popular, partly because the ARMAX formulation can subsume the general form without substantial penalty. The ARMAX form can be seen as an implementation of the so-called observer cannonical state-space form,

$$x_{k+1} = Fx_k + Gu_k + Ke_k$$

$$y_k = Hx_k + e_k$$

$$F = \begin{bmatrix} -a_1 & 1 & 0 & \dots & 0 \\ -a_2 & 0 & 1 & \dots & \\ -a_3 & 0 & & \ddots & \\ \vdots & & & & 1 \\ -a_n & 0 & & & 0 \end{bmatrix} \qquad G = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \qquad K = \begin{bmatrix} c_1 - a_1 \\ c_2 - a_2 \\ \vdots \\ c_n - a_n \end{bmatrix}$$

$$H = [1, 0 \dots 0]$$

The vector ARMAX models consist of the vector difference equations

$$A(z^{-1})y_{k+1} = B(z^{-1})u_k + C(z^{-1})e_{k+1}$$

where

$y_k, y_{k-1} \dots$ are vector outputs,

$u_k, u_{k-1} \dots$ are vector inputs

$e_k, e_{k-1} \dots$ are vector innovations

Some special model forms also have names,

$$A(z^{-1})y_{k+1} = B(z^{-1})u_k + e_{k+1}$$

is known as equation error form. It is obtained as a special case of the ARMAX form by setting $C(\cdot) = 1$.

Setting $P(\cdot) = 0$ and $C(\cdot) = 1$ gives the autoregressive (AR) form:

$$A(z^{-1})y_{k+1} = e_{k+1} \quad .$$

Setting $A(z^{-1}) = 1$ and $P(z^{-1}) = 0$ gives the moving average (MA) form:

$$y_{k+1} = C(z^{-1})v_{k+1}$$

The ARMAX forms most useful in identifying flutter modes are the recursive least squares (RLS) model ($F(z^{-1}) = C(z^{-1}) = D(z^{-1}) = 1$) , useful with persistent known inputs of reasonable magnitude with a very large signal-to-noise ratio, and the recursive maximum likelihood (RML) method [2] ($F(z^{-1}) = C(z^{-1}) = 1$) , useful for modeling with unknown input (the B polynomial can be zero when there is no known input) or with considerable noise disturbance. Identifying the $C(z^{-1})$ polynomial is equivalent to identifying the Kalman gain directly in a state-space innovations form model (described previously).

2.2.1 Recursive Least Squares (RLS)

RLS is a very popular recursive identification algorithm. It is very simple to implement and provides reasonable estimates with excellent robustness.

RLS gives exactly the same estimates as the batch least-squares on the cumulative data. In addition to giving the least squares parameter estimate at each time instant, RLS also permits forgetting of past data to accommodate quasi-

stationary models. It also permits incorporation of prior uncertainty about the starting values of the parameter estimates. The model underlying RLS is the same as the one for batch least-squares,

$$y(k) = \phi(k)\theta(k-1) + e(k)$$

In the context of linear dynamical systems the auto-regressive form and the equation error forms can be transformed to RLS form with

$$(1+a_1z^{-1}+a_2z^{-2}\ldots\ a_{nA}z^{-nA})y(k) = e(k)$$

$$\phi_k = \begin{bmatrix} -y_{(k-1)} \\ -y_{(k-2)} \\ \vdots \\ -y_{(k-nA)} \end{bmatrix} \qquad \theta = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{NA} \end{bmatrix}$$

and

$$(1+a_1z^{-1}\ldots+a_{NA}z^{-NA})y_{k+1} = (b_o+b_1z^{-1}\ldots+b_{NB-1}z^{-(NB-1)})u_k+e_{k+1}$$

$$\phi_k = \begin{bmatrix} -y_{(k-1)} \\ -y_{(k-2)} \\ \vdots \\ -y_{(k-n)} \\ u_{(k-1)} \\ u_{(k-2)} \\ \vdots \\ u_{(k-N_B)} \end{bmatrix} \qquad \theta_k = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_{nA} \\ b_o \\ b_1 \\ \vdots \\ b_{NB-1} \end{bmatrix}$$

2.2.2 Recursive Maximum Likelihood (RML2 [2])

Estimation of the A, B, and C polynomials of the ARMAX model:

$$A(z^{-1})y_k = B(z^{-1})u_k + C(z^{-1})e_k \ ,$$

is called recursive maximum maximum likehood. For SISO systems, since A and B are both monic polynomials, this equation can be written as

$$y_k = 1 - \frac{a(z^{-1})}{c(z^{-1})} y_k + \frac{b(z^{-1})}{c(z^{-1})} u_k + e_k$$

The one step predictor of y_k based on a certain set of values of a_i, b_i and c_i, denoted by θ, is as follows

$$\hat{y}_{k|\theta} = \left(1 - \frac{a(z^{-1})}{c(z^{-1})}\right) y_k + \frac{b(z^{-1})}{c(z^{-1})} u_k$$

The one-step-ahead prediction error is given by

$$\varepsilon_k(\theta) = y_k - \hat{y}_{k|\theta_{k-1}}$$

The prediction error methods are based on the minimization of a quadratic function of the form

$$J = \frac{1}{2} \sum_{k=1}^{N} \varepsilon_k^2(\theta)$$

The above minimization problem is nonlinear in θ. Hence an explicit solution is not available. Therefore, a numerical search procedure is used to find θ.

Differentiating J with respect to θ gives

$$J = \sum_{k=1}^{N} \varepsilon_k(\theta)\nabla\varepsilon_k(\theta)$$

∇J and $\nabla\varepsilon_k(\theta)$ are the gradients of J and $\varepsilon_k(\theta)$. The second gradient is given as follows

$$\nabla^2 J = \sum_{k=1}^{N} [\nabla\varepsilon_k(\theta)\nabla\ \varepsilon_k(\theta) + \nabla^2\varepsilon_k(\theta)\varepsilon_k(\theta)]$$

From the prediction error equation

$$\nabla\varepsilon_k(\theta) = -\nabla\hat{y}_{k|\theta}$$

The Hessian, $\nabla^2 J$, will now be approximated at the true minimum of J ; the second term of $\nabla^2 J$ can be shown to be zero at the true parameter value. Since the true Hession is more important close to the true minimum than elsewhere, we approximate the Hessian by the first term of $\nabla^2 J$ which, in view of $\nabla\varepsilon_K(\theta)$, becomes

$$\nabla^2 J \cong \sum_{k=0}^{N} \nabla\hat{y}_{k|\theta}\nabla^T\hat{y}_{k|\theta}$$

Further approximations are needed in order to obtain a recursive Gauss-Newton algorithm from the above equations and computation of $\varepsilon_k(\theta)$ requires all the data up to t . This computation is approximated by using the latest values of the data and the parameter estimates. Different approaches to affect this approximation lead to several algorithms. The RML2 algorithm uses the following approximation

$$\varepsilon(k) = y(k) - \bar{\phi}(k)\theta(k-1) ,$$

where

$$\bar{\phi}(k) = \begin{bmatrix} y(k-1) \\ \vdots \\ y(k-n_a) \\ u(k-1) \\ \vdots \\ u(k-n_b) \\ \bar{\varepsilon}(k-1) \\ \vdots \\ \bar{\varepsilon}(k-n_c) \end{bmatrix} ,$$

and $\bar{\varepsilon} = y_k - \bar{\phi}_k\theta_k$ are the residuals or the a posteriori prediction errors.

2.3 Block Recursive Identification Algorithms

In block recursive or semi-batch algorithms a block of data is stored in active memory and the identification routine may make several passes over the data. When the new data block is brought in, the old one is discarded. Maximum likelihood techniques are well suited to this situation, particularly in the known input case where an output error minimization can be used. Sensitivities of the outputs with respect to parameters are propagated as well as the estimate. The parameters of the Kalman gain in the innovations form can also be propagated for the unknown input case.

Two algorithms were developed in a high-level interpretive language [1] to study their performances. The first propagates the analytic sensitivities of a SISO system, propagating the minimum number of equations possible. The second algorithm is structured for a general MIMO system with potential nonlinearities. The sensitivities are found by perturbing the nominal model. The following sets of

discrete equations are propagated

$$X_{k+1} = \begin{bmatrix} f(x_k, u_k, \theta) \\ f(x_k, u_k, \theta+\Delta\theta_1) \\ \vdots \\ f(x_k, u_k, \theta+\Delta\theta_m) \end{bmatrix}, \qquad X_o = \begin{bmatrix} x_o \\ 0 \\ \vdots \\ 0 \end{bmatrix}_{(m+1)nx1}$$

where $\theta = \theta_{mx1}$.

The outputs and associated innovations at each time point k are:

$$\begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_m \end{bmatrix} = \begin{bmatrix} h(x_o, u, \theta) \\ h(x_1, u, \theta+\Delta\theta_1) \\ \\ h(x_m, u, \theta+\Delta\theta_m \end{bmatrix} \Rightarrow \begin{bmatrix} \nu \\ \nu_1 \\ \vdots \\ \nu_m \end{bmatrix} = \begin{bmatrix} z_o - z \\ (z_1 - z) - (z_o - z) \\ \vdots \\ z_m - z_o \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \nu \\ \vdots \\ \nu_m \end{bmatrix} = \Psi$$

Rather than analytic sensitivity time histories as the regressors, the perturbed sensitivities are regressed on the nominal innovations to identify a scaled version of the parameter step,

$$\nu_o = \frac{\partial J}{\partial \theta} \delta\theta \cong \Psi(\delta\theta/\Delta\theta)$$

The parameter update equation is

$\theta_{\ell+1} = \theta_\ell - (\delta\theta/\Delta\theta)_\ell \cdot \text{diag}(\Delta\theta_i)$, where ℓ is the interation count. The parameters associated with measurement equation above are updated with a U-D update performed once

for the entire data block. The measurement covariance is estimated by

$$\hat{R}_{\ell+1} = \text{diag}\{[\nu_o - \Psi(\widehat{\delta\theta/\Delta\theta})][\nu_o - \Psi(\widehat{\delta\theta/\Delta\theta})^T]\} \quad .$$

Results using this algorithm on a two mode example with very lightly damped modes are shown in Section 4.3.

SECTION 3

ALGORITHM RELIABILITY, ACCURACY AND ROBUSTNESS

The previous section described both a general class of recursive estimators called recursive predictive error methods and a semi-batch maximum likelihood algorithm for on-line flutter identification. Application of such algorithms to flutter identification can not be made in a useful way without a qualitative and quantitative assessment of the reliability, accuracy and robustness of the algorithms. By reliability we mean, can the algorithm fail to converge, converge to a wrong value or give grossly erroneous accuracy estimates? An accurate, but large parameter error estimate is reliable information, quantifying the appropriate confidence in poorly identifiable parameters. The accuracy of parameter estimates can be assessed in three ways; simulation, the Cramer-Rao bound or estimates of this bound from either the propagated covariance or limit evaluation. All three accuracy measures and their use with flight data are discussed in this section. Robustness of flutter identification algorithms addresses the ability of the algorithm to maintain reasonable accuracy under violations of assumptions on the model and experiment. Thus, all three of these algorithm attributes are inter-related as will be elaborated in subsequent discussion.

3.1 Algorithm Reliability and Convergence

The primary reliability issue for flutter identification algorithms is convergence.

- Do the estimates of the modal parameters converge?

- If the estimates converge, what is their limit? Is it unique?
- What is the convergence rate and the accuracy of the estimates?

The semi-batch maximum likelihood algorithm requires good a priori estimates for convergence, but the convergence is quadratic in the neighborhood of the true values. An example is shown in Section 4.3 where RPEM applied to only 100 points can give excellent ML start-up values from an initial RPEM estimate of zero for all parameters. Quite a bit can be said about the convergence of recursive predictive error methods. The conclusions are affected by the user choice of model set, choice of input signal, choice of criterion function, choice of gain weighting sequence, choice of numerical serach direction, and choice of initial conditions, as well as the selection of the "approximate" gradient of the predictor (see Ljung and Sodestrom [3] for a thorough discussion of all these issues). Here we summarize general conclusions on RPEM applicable to the chosen model sets (see Section 2.2) and experimental conditions experienced in flutter mode monitoring.

The recursive predictive error method <u>will converge with probability one to a local minimum of the expected cost as N, the number of observed data, tends to infinity</u>. Whether local minima exist apart from the global minimum, depends on the model strucure chosen. For the auto regressive form

$$A(z)y_k = e_k ,$$

and ARMA form

$$A(z)y_k = C(z^{-1})e_k ,$$

no false local minima exist [4]. For the equation error model set,

$$A(z^{-1})y_k = B(z^{-1})u_k + e_k ,$$

no false local minima exist since the cost is quandratic in the parameters.

For the ARMAX (Auto Regressive Moving Average with eXogenous inputs) model set,

$$A(z^{-1})\ y_{k+1} = B(z^{-1})\ u_k + C(z^{-1})\ e_{k+1}$$

The negative log likelihood function of the ARMAX form has a unique global minimum if the signal-to-noise ratio is sufficiently large. Local minima do exist if the signal-to-noise ratio is very small [5]. This will be the principal model used, with the Recursive Prediction Error algorithm (RPEM [3], also known as RML2 [2] for the ARMAX form), which implements the Gauss-Newton update for parameter estimates.

The RML1 algorithm [2] and the AML (approximate Maximum Likelihood) algorithm of Solo [7] use a further approximation in the Gauss-Newton update which in turn poses the so-called Strict Positive Real condition for convergence [8]. However, the projection of the estimates into a region of stability is not needed with AML. Even the problem of the estimates getting trapped into local minima disappears. If the identifiability conditions are not met, the parameter estimates do not converge, but the prediction-error sequence does converge to its minimum value.

Starting RPEM or RML2 as AML and gradually shifting to fully RPEM or RML2 is possible by using the so-called contraction factor on the estimated $\hat{C}(z^{-1})$ polynomial [3]. Doing so eliminates the problem of getting trapped in the local minima if they exist. The transient convergence rate is also markedly improved.

Another significant assertion about the above convergence results, is that convergence holds whether or not the actual system has the same model structure as the chosen one. The identification procedure will pick the "best"

approximation of the system, or in other words the approximation from the selected model structure which best minimizes the prediction errors. However the identified model may depend on the input, since the best approximation for a sinusoidal input, for example, may be different than for a white noise input. If the selected model structure is exactly the same as the true system, the "best approximation" is equal to the true system, independent of the input.

In summary, these black-box models are highly desirable for identification of flutter modes where a detailed model of the aircraft may not be available and could change considerably during a flutter test.

Thus far we have described whether the proposed algorithms converge and if so, whether they converge to a unique global minimum. In the next section a description of asymptotic distribution of the estimate $\hat{\theta}(t)$ will be given, which provides both the rate of convergence and the accuracy of the estimates asymptotically. It should be emphasized that there is no similar analysis for the transient convergence behavior of recursive algorithms, and that this behavior is very strongly a function of the "user choices" mentioned at the outset of this section. Transient convergence behavior can best be investigated by simluation. The simulation performance, of the model analyzed theoretically in Subsection 3.22, for short, intermediate, and long data records is given in Section 4.

3.2 Estimation Accuracy

The semi-batch maximum likelihood method and the recursive prediction error technique propagate the covariance matrix (inverse of the Cramer-Rao information matrix). This matrix is propagated in the U-D factored form. The Cramer-Rao bound is based on an assumed noise distribution, typically Gaussian noise sources, which may not hold in flight tests.

Also the bound holds for the asymptotic nature of the parameter estimates. The physical environment and system characteristics can change before a long enough data record can be processed, to achieve the asymptotic behavior.

The estimation errors predicted by the Cramer-Rao bound are usually too small. However, it provides an excellent measure of relative parameter estimation errors. Experience [9] has shown that in stability and control derivative estimation, the Cramer-Rao bound should be multiplied by a factor of three to five to obtain actual estimation errors. In model parameter estimation with high band width sensors, the factor is closer to two based on limited experience. The simulation cases of Section 4.4 show the actual errors and the theoretical bound (computed from the true model). For the long data simulation sequences, the bound gives a good accuracy estimate for the input noise sequences, which are Gaussian in the simulations.

For both the semi-batch maximum likelihood approach and the recursive predictive error method, the information matrix inverse is approximated as

$$P = M^{-1} = \left[\frac{\partial V}{\partial \theta} \hat{R}^{-1} \frac{\partial V}{\partial \theta}^{T} \right]^{-1}$$

In other words the second gradient of the criterion function (V) is approximated as a weighted outer product of the first gradient of V with respect to the parameters, where $\hat{R}$ is the estimated innovations covariance.

3.2.1 Maximum Likelihood Estimation Accuracy

As explained in Section 2.3 the parameter update in the semi-batch ML algorithm which was the inverse of the Hession, is accomplished with the Bierman's U-D update routines [10].

The variances of identified parameters can be obtained easily from the U-D factored form of the covariance. These parameters are, however, parameters of a discrete model used for efficient propagation, whereas we are interested in the continuous frequency and damping estimates and their standard deviations. We can use the Jacobian of the transformation from the discrete roots (the ML parameterization) to the continuous frequency and damping parameters. This is a local linearization of this non-linear mapping; however, for lightly damped modes, this transformation is well-conditioned indicating that it is not sensitive to small variations in discrete parameter values.

Since the parameter covariance is

$$P_\theta = E[\Delta\theta \; \Delta\theta^T] \; ,$$

the covariance of different parameters which are functions of the original ones

$$\underline{\theta}_1 = f(\theta) \; ,$$

$$P_{\theta_1} = JP_\theta J^T$$

where

$$J = \frac{\partial f}{\partial \theta} \Rightarrow \quad \Delta\theta_1 = J\Delta\theta \; .$$

It is easier to write the inverse function in this case

$$\theta = f^{-1}(\theta_1) \; ,$$

where θ are the discrete parameters and θ_1 the continuous frequencies and damping ratios. For each mode, the discrete parameters are given by

$$\sigma_D = e^{-\xi\omega_n\tau}\cos[\omega_n(1-\xi^2)^{\frac{1}{2}}\tau] \quad ,$$

$$\omega_D = e^{-\xi\omega_n}\sin[\omega_n(1-\xi^2)^{\frac{1}{2}}\tau]$$

and their variations are given by

$$\begin{bmatrix}\Delta\sigma_D \\ \Delta\omega_D\end{bmatrix} = \begin{bmatrix}\frac{\partial\sigma_D}{\partial\omega_n} & \frac{\partial\sigma_D}{\partial\xi} \\ \frac{\partial\omega_D}{\partial\omega_n} & \frac{\partial\omega_D}{\partial\xi}\end{bmatrix} \begin{bmatrix}\Delta\omega_n \\ \Delta\xi\end{bmatrix} \quad .$$

Hence

$$J_i^{-1} = -\tau e^{-\xi\omega_n\tau}\begin{bmatrix}[\xi\cos(\omega_{n_D}\tau)+\frac{\omega_{n_D}}{\omega_n}\sin(\omega_{n_D}\tau)], & \tau[\cos(\omega_{n_D}\tau)-\xi\frac{\omega_n^2}{\omega_{n_D}}\sin(\omega_{n_D}\tau)] \\ [\xi\sin(\omega_{n_D}\tau)-\frac{\omega_{n_D}}{\omega_n}\cos(\omega_{n_D}\tau)], & \tau[\sin(\omega_{n_D}\tau)+\xi\frac{\omega_n^2}{\omega_{n_D}}\cos(\omega_{n_D}\tau)]\end{bmatrix}$$

where τ is the sampling time and $\omega_{n_D} = \omega_n(1-\xi^2)^{\frac{1}{2}}$ the continuous damped frequency. With the U-D factored covariance it is necessary to "square up", the new covariance. However, for accuracy prediction, any loss of precision due to "squaring up", is insignificant in $P_{\theta_1} = JUDU^TJ^T$, where for k modes

$$J = \begin{bmatrix} I_{NH_{parms}} & & & \\ & [J_1] & & \\ & & \ddots & \\ & & & [J_k] \\ & & & & I_{NG_{parms}}\end{bmatrix}$$

This estimate of the Cramer-Rao bound from the inverse of the approximated Hessian is accurate only in the region of the convergence point. This is demonstrated quite clearly at the convergence point on the simulated example of

Section 4.3, where the actual estimation errors show nearly quadratic behavior as well.

3.2.2 Recursive Predictive Error Method (RPEM) Accuracy Estimates

The internal covariance estimate propagated in U-D factored form in the RPEM algorithm is not suitable to estimate the Cramer-Rao bound for short data records because it is regularized to prevent singularity and also effected by the transient effects in the exponential forgetting factor which weights recent data more significantly than past data. The Cramer-Rao bound can be computed for RPEM algorithms, however, exactly in the simulation case, where the true parameters are known, and estimated during a flight test propagated UD covariance after the initial convergence transient.

Consider the ARMAX model,

$$A(z^{-1})y_k = B(z^{-1})u_{k-1} + C(z^{-1})e_k \quad ,$$

with the negative log likelihood function

$$V = \frac{1}{2N}\sum_{k=1}^{N} \varepsilon_k^2(\theta), \quad \text{with} \quad \varepsilon_k(\theta) = y_k - \hat{y}_{k|\theta}$$

$$\hat{y}_{k|\theta} = \theta^T \phi_{k|\theta}$$

as described in Section 2.2.2. The individual component derivatives of $\frac{\partial \hat{y}_k}{\partial \theta}$ are

$$\frac{\partial \hat{y}_k}{\partial a_i} = \frac{1}{C(z^{-1})} y_{k-i} \quad ,$$

$$\frac{\partial y_k}{\partial b_i} = \frac{}{C(z^{-1})} u_{k-i} \quad ,$$

$$\frac{\partial \hat{y}_k}{\partial c_i} = \frac{1}{C(z^{-1})} e_{k-i} .$$

Thus gradient of $\frac{\partial \varepsilon_k}{\partial \theta} = \psi_k = \frac{1}{C(z^{-1})} \phi_k$. The regressors are "filtered" versions of the output, input, and innovation sequences.

We seek the asymptotic parameter covariance, where

$$(NP)^{-1} = \bar{E}[\psi R^{-1} \psi^T] = \frac{1}{N} \sum_{k=1}^{N} E[\psi R^{-1} \psi^T] ;$$

and N is the number of data points.

Just as the ARMAX model above can be realized as a state-space innovations model, the gradient ψ_k can also be realized as a linear model driven by the sequences u_k and e_k . This realization uses an augmental state and input

$$\zeta_k = \begin{bmatrix} \psi_k \\ \\ \phi_k \end{bmatrix} , \qquad w_{k-1} = \begin{bmatrix} u_{k-1} \\ \\ e_{k-2} \end{bmatrix} ,$$

in the equation,

$$\zeta_k = F\zeta_{k-1} + G\, w_{k-1}$$

For the scaler output case,

$$P = \frac{\Psi^{-1}}{NR} , \quad \text{with} \quad \Psi = HZH' ,$$

where Z is the solution of the discrete Lyapunov equation

$$Z = FZF^T + GQG^T .$$

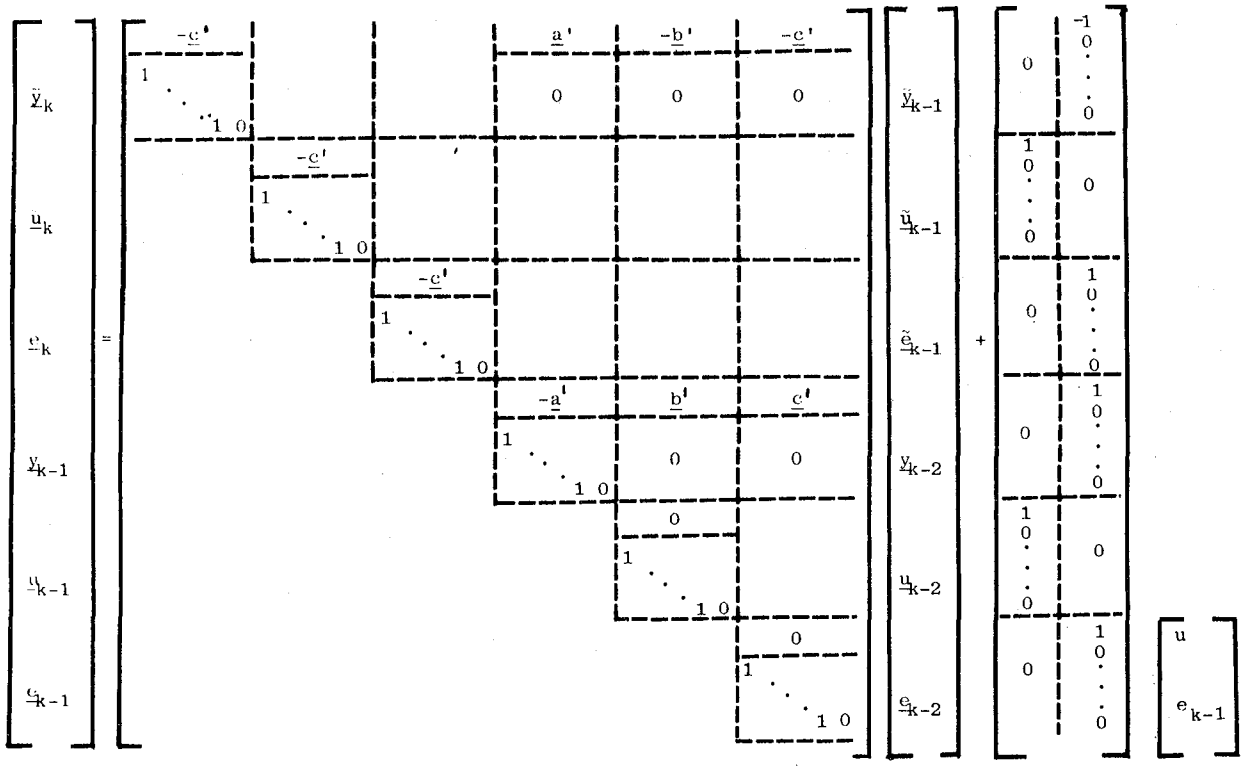
Q is the covariance of w_k, and H selects the upper corner of Z corresponding to the row dimension of ψ. The realization of the ζ_k equation above uses the relations

$$(1 + c(z^{-1}))\psi_k = \phi_{k-1} \cdot$$

and

$$y_k = \dot{\theta}_{k-1}\phi_k + e_k \quad ,$$

giving the following augmented equation.



Given quite general conditions on the experiment (see Ljung [3, Chapter 4], and conditioned on the event $\hat{\theta}_k \to \theta^*$ with probability 1 (w.p.1), then the constant scaled Lyapunov solution

$$NP = R\Psi^{-1}(\theta^*)$$

is the asymptotic variance of the statistic

$$\sqrt{N}\ (\hat{\theta} - \theta^*) \varepsilon\ N(0,NP)\ .$$

Noreover, for any $\delta > 0$

$$N^{\frac{1}{2}-\delta} |\hat{\theta}(N) - \theta^*| \to 0 \quad \text{w.p.1} \quad \text{as} \quad N \to \infty$$

(see Ljung [11] for the general, formal statement and proof of this theorem).

In simulation cases where θ^* is known the Cramer-Rao bound above can be computed and used to estimate the theoretically attainable accuracy of the parameter estimates as a function of time, whereas with experimental data only the identified model can be used to estimate this bound. The simulated results of Section 4.4 compare the accuracy of one single simulation and the theoretical Cramer-Rao bound for different data lengths.

In contrast to the maximum likelihood formulation, when the model is already in discrete modal form, an additional transformation must be applied to convert the discrete parameter covariance matrix to the continuous frequency and damping parameter covariance as follows,

$$P_{\theta_1} = J\ X\ P_\theta X^T J^T\ ,$$

with J defined as in the previous subsection, and

$$x = \begin{bmatrix} \mathrm{Re}\left[x^{(1)}_{(1)}\right] & \cdots & \mathrm{Re}\left[x^{(2k)}_{(1)}\right] \\ \mathrm{Im}\left[x^{(1)}_{(1)}\right] & & \mathrm{Im}\left[x^{(2k)}_{(1)}\right] \\ \vdots & & \vdots \\ \mathrm{Re}\left[x^{(1)}_{(k)}\right] & \cdots & \mathrm{Re}\left[x^{(2k)}_{(1)}\right] \\ \mathrm{Im}\left[x^{(1)}_{(k)}\right] & & \mathrm{Im}\left[x^{(2k)}_{(1)}\right] \end{bmatrix} \qquad x^{(i)}_{2kx1} = \mathrm{diag}\ \{M^{-1} \begin{bmatrix} 0 \\ \vdots \\ 1_{i,1} \\ \vdots \\ 0 \end{bmatrix} \quad 0 \quad M\}\ .$$

M is the modal matrix which transforms the polynomial parameters

$$M \begin{bmatrix} -a & \vdots & 1 & & \\ & \vdots & & \ddots & \\ & \vdots & & & 1 \\ & & & & 0 \end{bmatrix} M^{-1} = \begin{bmatrix} \sigma_1+j\omega_1 & & & \\ & \sigma_1-j\omega_1 & & \\ & & \ddots & \\ & & & \sigma_k+j\omega_k \\ & & & & \sigma_k-j\omega_k \end{bmatrix}$$

The above transformation was applied to the Cramer-Rao bound based on the true simulation parameters to produce the results of Tables 4-3 through 4-5. A high eigenvalue condition number, i.e., $s_i(\lambda_i) = \|x_i y_i^T\|$ (norm of λ_i's spectral projector; x_i, y_i are the left and right eigenvectors), is a warning that the respective polynomial roots are very sensitive to errors in the polynomial coefficients. For the simulation results described in the tables mentioned above, the eigensystem condition for both the true parameters and the estimated ones was a reasonable level, less than approximately 300 in the worst case.

In the known input case, which is the recursive least squares, also called the equation error form,

$$A(z^{-1})y_k = B(z^{-1})u_{k-1} + e_k ,$$

the gradient is

$$\frac{\partial e_k}{\partial \theta} = \phi(k) ,$$

where

$$\phi_k = [y_{k-1} \cdots y_{k-na}, u_{k-1} \cdots u_{k-nb}]^T$$

For arbitrary inputs u_k, the asymptotic covariance is

$$NP = \bar{E}[\phi R^{-1} \phi^T]^{-1}$$

This parameter covariance can then be found in terms of autocorrelations and cross-correlations of the input and output sequences.

For the special case where u_k and e_k are random sequences we can compute the parameter covariance directly by solving a discrete Lyapunov equation as in the ARMAX case, which employs the covariance of the two random sequences. The state space realization uses the state and input,

$$\zeta_k = \begin{bmatrix} \phi_k \\ \hat{x}_k \end{bmatrix}, \qquad w_k = \begin{bmatrix} u_k \\ e_k \end{bmatrix},$$

in the equation

$$\zeta_k = F\zeta_{k-1} + Gw_{k-1}$$

with F and G given by

$$F = \begin{bmatrix} 0 & & & & \\ 1 & \ddots & & & \\ & \ddots & 0 & & \\ & 1 & 0 & & \\ & & & 0 & \\ & & & 1 & \ddots \\ & & & \ddots & 0 \\ & & & 1 & 0 \\ & & & & \begin{matrix} -a & -b \\ 1 & \ddots \\ \ddots & 0 \\ 1 & 0 \end{matrix} \\ & & & & \begin{matrix} 1 & \ddots \\ \ddots & 0 \\ 1 & 0 \end{matrix} \end{bmatrix}, \quad G = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ \\ 1 \\ 0 \\ \vdots \\ 0 \\ \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Computation of the Cramer-Rao bound as described in this subsection allows the investigation of the effect of different gust and sensor noise levels without performing multiple

simulations. This is discussed in the next section under the general topic of robustness.

3.3 Robust Flutter Mode Identification

Violations in the assumptions of the statistical nature of the disturbances require estimation techniques robust with respect to distribution. These techniques are summarized and demonstrated by simulation. Violations in the assumptions of the number of active modes, the gust levels, or characteristics of the sensor noise effect the accuracy of estimation. The results of the previous section can be used to efficiently quantify these effects.

3.3.1 Robustness with Respect to Distribution

Robust estimation procedures are very important because a few bad data points can significantly degrade parameter estimation accuracy. In some cases convergence characteristics would also be affected. Ljung [3] summarizes robust estimation problem as follows:

1. When the measured data set contains some values that are abnormal, e.g., quite large due to sensor failures, straightforward use of a quadratic criterion function will give substantial jumps of the parameter estimates. Moreover, the estimates need a long period before converging back to the previous levels.
2. A way to cope with this problem, i.e., to robustify the algorithm, is to use a criterion function that grows more slowly with ε than the quadratic one. Then large prediction errors will get less influence on the parameter estimates and the algorithm becomes more robust.
3. Another approach is to test recursively if the data contains outliers. This can be done by comparing the prediction errors with a specified limit. Large prediction errors mean that an outlier or a measurement error is probable. The measurement can then be substituted with the predicted value. This approach is applicable when there is only a few outliers in the data.

Both identification techniques we have discussed in this section are based on minimizing a quadratic penalty functional, e.g.

$$V = \frac{1}{2} \sum_{k=1}^{N} e_k^2$$

The robust procedures modify this cost functional to include a weighting function g which is monotonically decreasing

$$V = \frac{1}{2} \sum_{k=1} g \frac{e_k^2}{B_k} e_k^2$$

where B_k is the covariance of e_k .

The use of such a robustness procedure in the real-time flutter identification procedure will now be discussed.

Recursive identification techniques are inherently well-suited for implementing robust estimation techniques because the reasonableness of each new data point can be evaluated before it is used to update the parameter estimates. An illustrative discussion of robustness measures and robust identification is given by Ljung [3]. A summary of the steps necessary to make the parameter estimates robust against data dropouts, wild points and outliers is as follows:

Step 1: After a data point is processed, the estimated parameter values are used to predict the next data and the standard deviation of the predicted value. Let the predicted value be y_p with standard deviation σ .

Step 2: The next data point is compared to its predicted value. The quantity

$$\xi = \frac{|y - y_p|}{\sigma}$$

is a measure of the probability that the data point is wild. Data points with $\xi > 3$ should be suspect. Thus, data corresponding to large values of ξ should be given less weighting. Depending on the expected distribution of data droupouts, any of the weighting functions of Figure 3-1 can be selected (see Huber [12]). If $f(\xi) = y_\xi$, the estimates have the robustness of a median.

Step 3: A data point completely outside the predicted distribution is not used

A restart is necessary if several contiguous points are rejected.

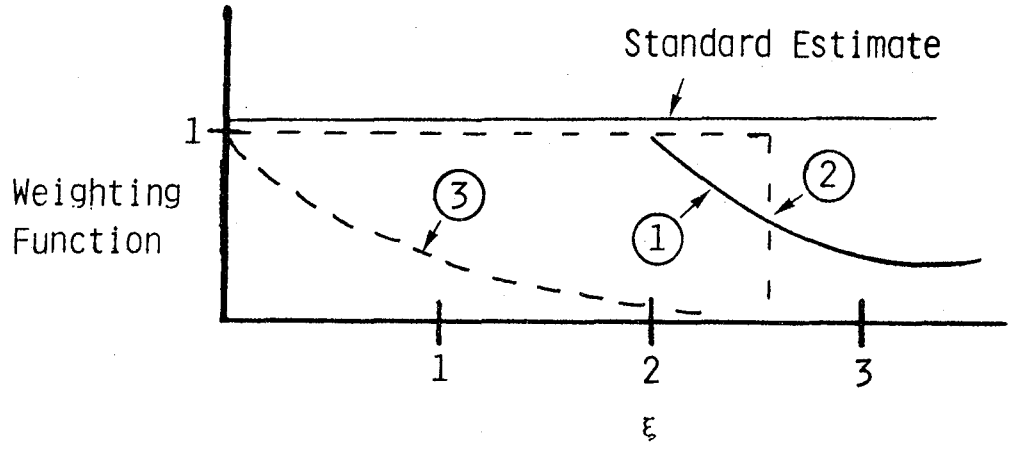


Figure 3-1 Weighting Functions for Enhanced Robustness

A weighting function of the second type was demonstrated, where the cost function is

$$\frac{e^2}{2\sigma^2} \qquad |e| < \alpha$$

$$\text{Sign}(e)\ \frac{e^2}{2\sigma^2} - \frac{(e-\alpha)^2}{2\sigma^2} \qquad |e| > \alpha \quad , \quad \text{with} \quad \alpha = 3\sigma$$

The measurements with predicted errors out of bounds are replaced with

$$y = y_p + \alpha \text{ sign } (e) \left[\frac{2e}{\alpha} - 1\right]^{\frac{1}{2}} .$$

Data dropouts have been generated with distribution shown below.

Amplitude	Normally distributed with the same variance as the data but centered at 5σ from the output estimate
Frequency and sign	Semirandom telegraph signal with λ = 1%

Results with and without these dropouts are compared in Table 3-1 for the known input case. (This table with a 6-model identified mode can also be compated to the same simulation in Tables 4-3 and 4-4, which have no data dropouts and use an 8-mode model.)

	ω_n	$\omega_n - \omega_n$	ξ(%)	$\hat{\xi}$(%)
125 points with robust estimation	1	.9998	4.70	8.50
	2	1.9592	4.80	6.36
125 point w/o robust estimation	1	.9998	4.70	8.48
	2	1.9597	4.80	6.39
500 points with robust estimation	1	.9997	4.70	4.78
	2	2.0009	4.80	4.71
500 points w/o robust estimation	1	.9980	4.70	4.87
	2	2.0037	4.80	4.51

Table 3-1. RPEM Frequency/Damping Estimates (With and Without Robustness Techniques - Known Input with Standard Gust Cases σ_u/σ gust = 25)

The robust estimation procedure can improve the accuracy of the flutter parameters (see Table 3-1); however, during the rapid portion of the algorithm convergence, the wild points actually increase the convergence rate. The extent to which the model is overparameterized also has a significant effect in making RPEM robust. 8-mode models identified from this data

showed little degradation until asympotic accuracies were achieved, and then the degradation was not severe.

3.3.2 Effects of Noise and Modeling Errors

Very high frequency effects can be modeled as white noise, unmodeled system dynamics with colored noise, and low frequency effects as a random walk. Errors in estimation are caused by an incorrect order linear approximation model, an incorrect intensity of Gaussian noise, or finally assumption that the noise distribution is, in fact, Gaussian.

The effects of these different noise sources on the Cramer-Rao bound for the modal parameters can be computed very efficiently for a large number of cases by merely solving matrix Ricatti and Lyapunov equations for each desired situation, rather than performing a simulation and doing the identification for a long data record. Consider simulations generated for the 2-mode model of Section 4.4. according to the following model,

$$x_{k+1} = Fx_k + Gw_k + Gu_k$$

$$y = Hx_k + v_k \quad ,$$

with

$$w_k = N(0,q) \quad , \qquad q = \sigma^2_{gust} \quad ,$$

$$u_k = N(0,\sigma^2_u)$$

$$v_k = N(0,r) \quad , \qquad r = \sigma^2_{sensor} \quad ,$$

but the sequence u_k is known. u_k can be considered wideband excitation which could quite possibly be a pseudo random binary sequence; we have made it Gaussian here for computational simplicity.

The "nominal" model, about which trends for how these different disturbances or inputs effect the achievable parameter estimation accuracy will now be described. The two modes have frequencies of 1 and 2 per rev (borrowing a term from rotor craft analysis for normalized frequencies), where 1 per rev is 11.8 hertz-emulating the open loop modes in the ARW-1 vehicle. The sampling rate is approximately 300 hertz. The gust noise covariance ($\sigma^2_{gust} = (.15)^2$ deg/rev^2) has been chosen to correspond to a continuous model with vertical gust root-mean-square values of approximately 1-2/3 feet/sec[†] (at the ARW-1 condition of Mach of .7, altitude of 15,000 feet, and a correlation distance of 1750 feet). The damping parameters were chosen to be similar to the ARW-1 flight condition analyzed in Section 5.1 (an 11.8 Hz mode with 4.70% of damping and a 23.6 Hz mode with 4.80% damping). The sensor noise observed on the data (see Figure 5-11) was taken to be approximately $R_o = .0050$. It should be pointed out that the significant difference between this simulation and the flight maneuver is the use of broad band rather than narrow band excitation.

The theoretically achievable parameter accuracy as a function of input and noise power is shown in Figures 3-2 and 3-3 for damping ratio and natural frequency estimation errors. The flutter parameter variances are a function of the gust, control input, and sensor noise covariances, variations of which are shown with families of curves in the upper two plots of the figures. The three-dimensional plots show the flutter parameter standard deviations as a function of power ratios (with a log scale on all axes). The parameter covariances have been computed via a discrete Lyapunov equation as described in subsection 3.2.2; they require the control and innovation covariances as input parameters. The innovation covariances were computed directly from the gust and noise covariances via the discrete algebraic Riccati equation [13].

[†]The gust level was very low during the ARW-1, Flight 3.

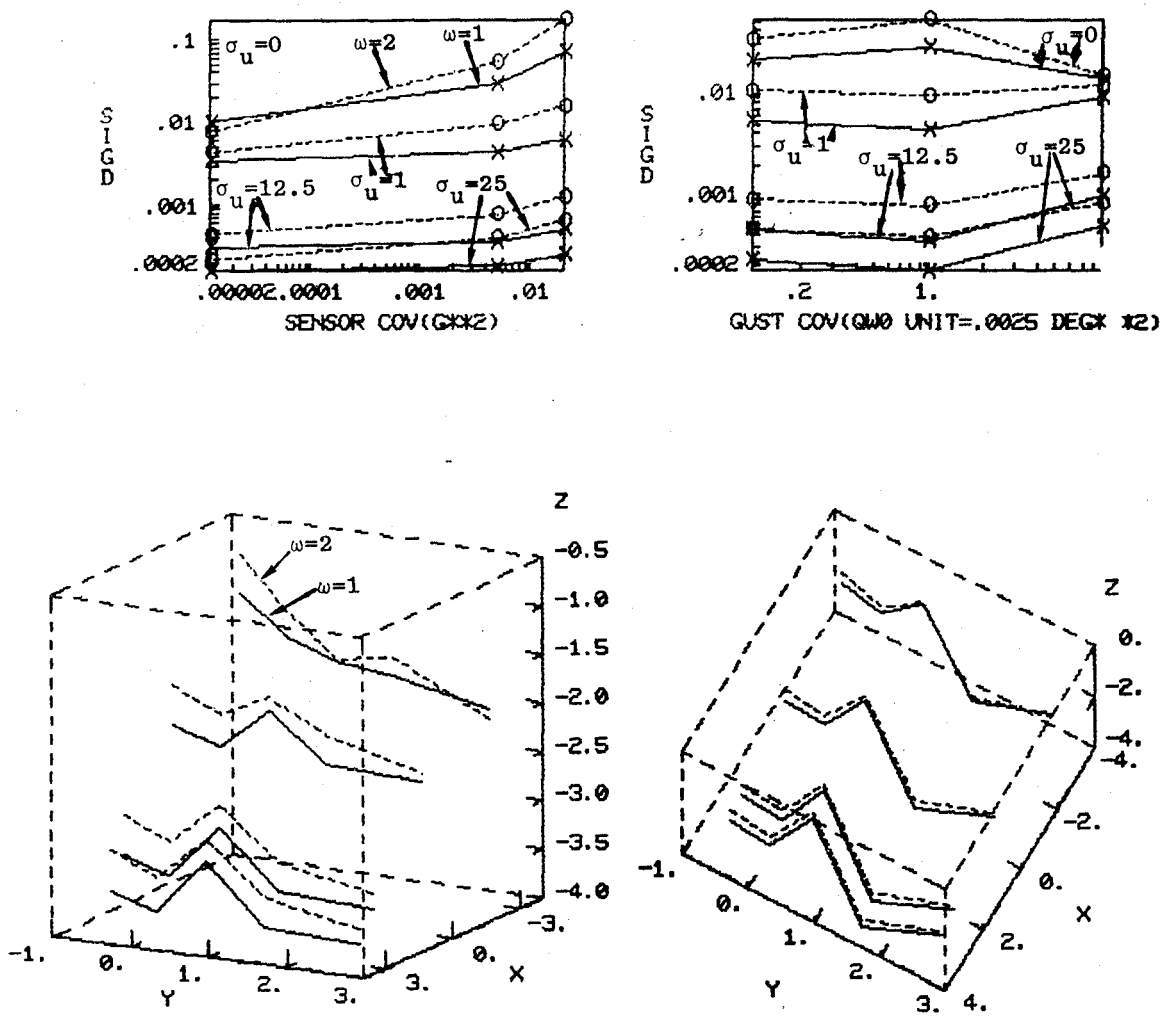


Figure 3-2. Effects of Gust Noise, Sensor Noise, and Broadband Input Power on Damping Estimation Errors - Two Mode Model.

(Nominal values are σ^2_{gust} = .0025 deg^2 of equivalent aileron)
σ^2_{sensor} = .005 g^2)

Note: 3-dimensional axes are Log_{10} of the data.
x-axis is $\sigma_u{}^2/\sigma_w{}^2$; y-axis is Q_w/R ; z-axis is $\sigma_{damping\ ratio}$.

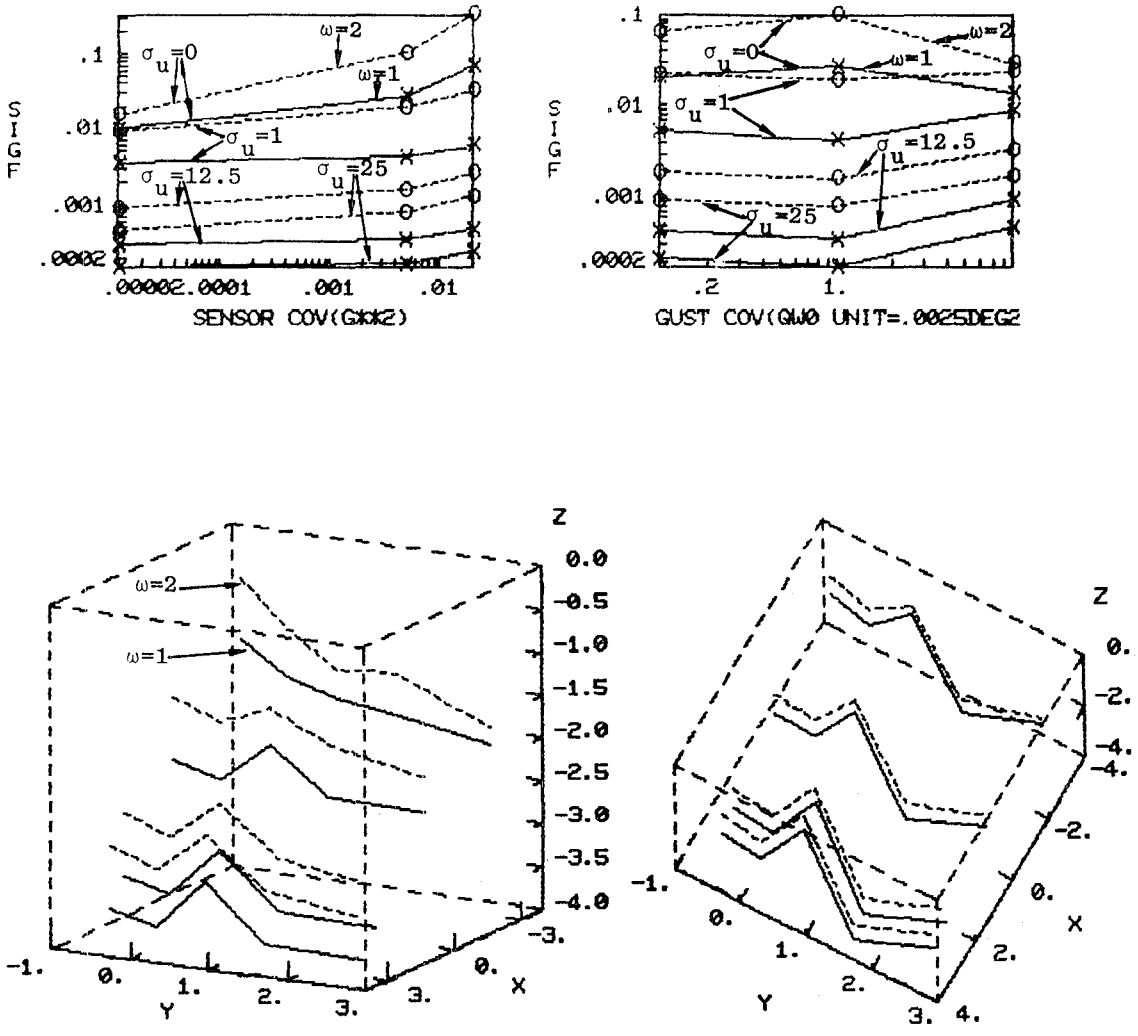


Figure 3-3. Effects of Gust Noise, Sensor Noise, and Broadband Input Power on Natural Frequency Estimation Errors - Two Mode Model

Note: 3-dimensional axes are Log_{10} of the data.
x-axis is σ_u^2/σ_w^2 ; y-axis is Q_w/R ; z-axis is $\sigma_{\text{damping ratio}}$.

The variations about nominal noise covariances were

$$q_{w_o} \begin{bmatrix} 9 \\ 1 \\ 1/9 \end{bmatrix} \quad , \quad \text{and} \quad R_o \begin{bmatrix} 4 \\ 1 \\ 1/400 \end{bmatrix} \quad .$$

The gust variations correspond to a range of gust velocity of $\frac{5}{9}$ft/sec - 5ft/sec , the nominal at $1\frac{2}{3}$ft/sec for the low gust values seen in the flight data, the low corresponding to negligible gust, and the 5ft/sec the Dryden gust model. The sensor nominal was varied to twice the noise standard deviation and twenty time less (corresponding to inertial grade instruments). The input power variations were

$$\sigma_u^2 = \begin{bmatrix} 0 \\ 1 \\ 156.25 \\ 625 \end{bmatrix} q_{w_o}$$

The highest level gave an amplitude response similar to the high-level level flight data inputs, and the next lower level is one-half the amplitude corresponding to the low-level flight data inputs.

These theoretical predictions are compared to simulation results in Section 4.4.

SECTION 4

SIMULATED DATA RESULTS

Three groups of flutter examples were simulated to test the performance of the RPEM and the maximum likelihood (ML) algorithms:

1. A two-mode example with very lightly damped modes at a frequency ratio of two.

2. Different combinations of two modes with light to moderate damping (1% - 15%).

3. Two-mode examples with different noise and modal disturbances discussed in the previous section on robustness, reliability and accuracy.

4.1 VERY LIGHTLY DAMPED MODES

The linear model used to test both the RPEM algorithm as the start up estimator, as well as the ML algorithm is

$$\dot{x} = Fx + Gu$$

$$y = Hx \quad ,$$

where

$$F = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & -.0136 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -4 & -.01 \end{bmatrix}, \quad G = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix},$$

$$H = [-1.9456 \quad 1.2239 \quad 2.1867 \quad -.07552] .$$

This model was configured to emulate the torsion response (wing

bending and torsion modes) of a wing with two modes close to the flutter condition with time measured in normalized units (revs). The frequency response of this model is shown below in Figure 4-1.

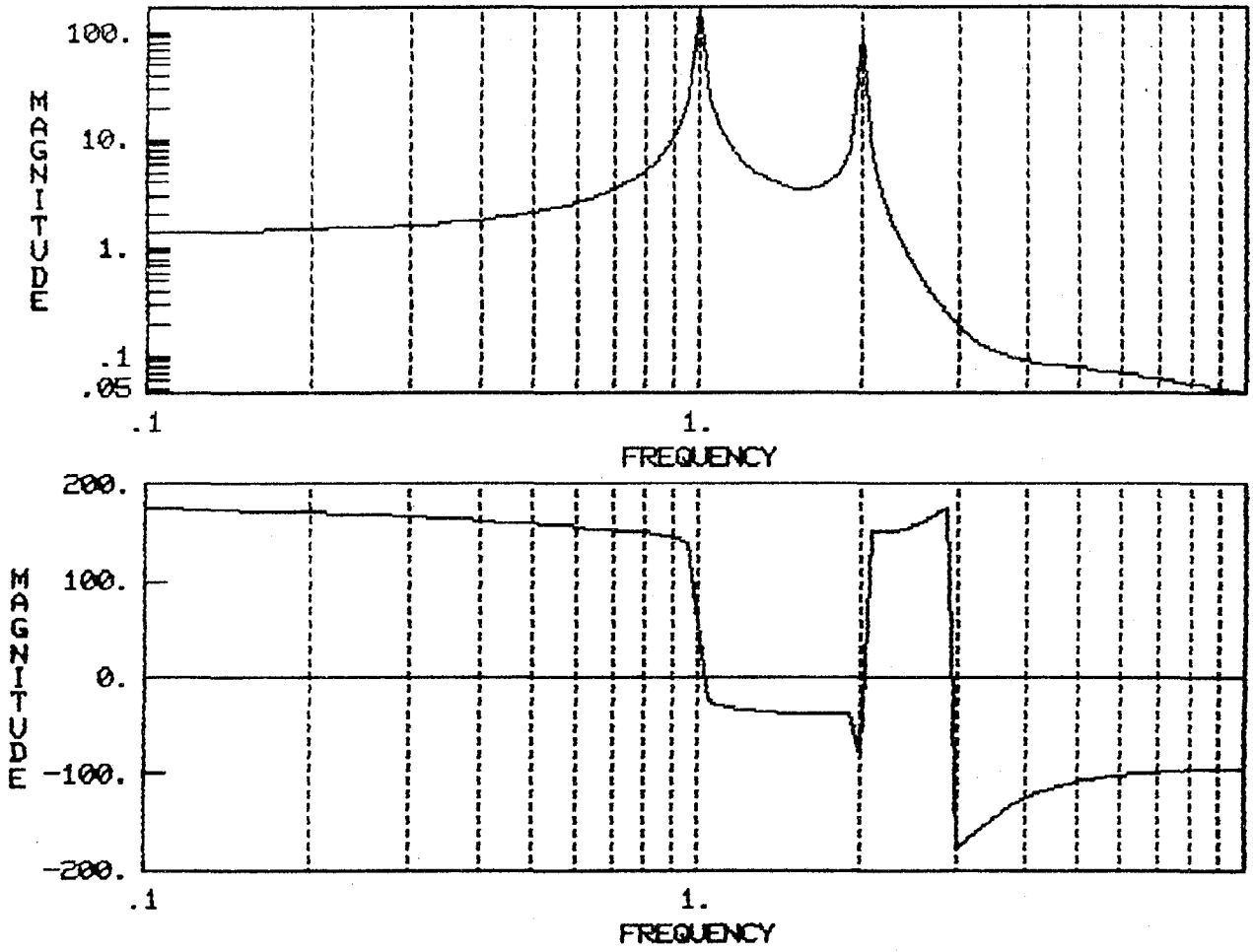


Figure 4-1. Two Mode Lightly-Damped Model Frequency Response. (Frequency in Per Rev)

As mentioned in Section 2, it is much more efficient to propagate linear systems in discrete form. The discrete form of this model (for a sampling time of .23 revolutions) is given by

$$x_{k+1} = F_D x_k + G_D u_k \quad ,$$

$$y_k = H_D x_k \quad ,$$

where

$$F_D = \begin{bmatrix} .9723 & .2268 & 0 & 0 \\ .2268 & .9723 & 0 & 0 \\ 0 & 0 & .8958 & .4420 \\ 0 & 0 & -.4420 & .8958 \end{bmatrix} , \qquad G_D = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} ,$$

$$H_D = [-.4733 \quad .2268 \quad .5027 \quad -.1105] .$$

The output of this linear system as forced by a known gaussian input gust sequence is shown in Figure 4-2.

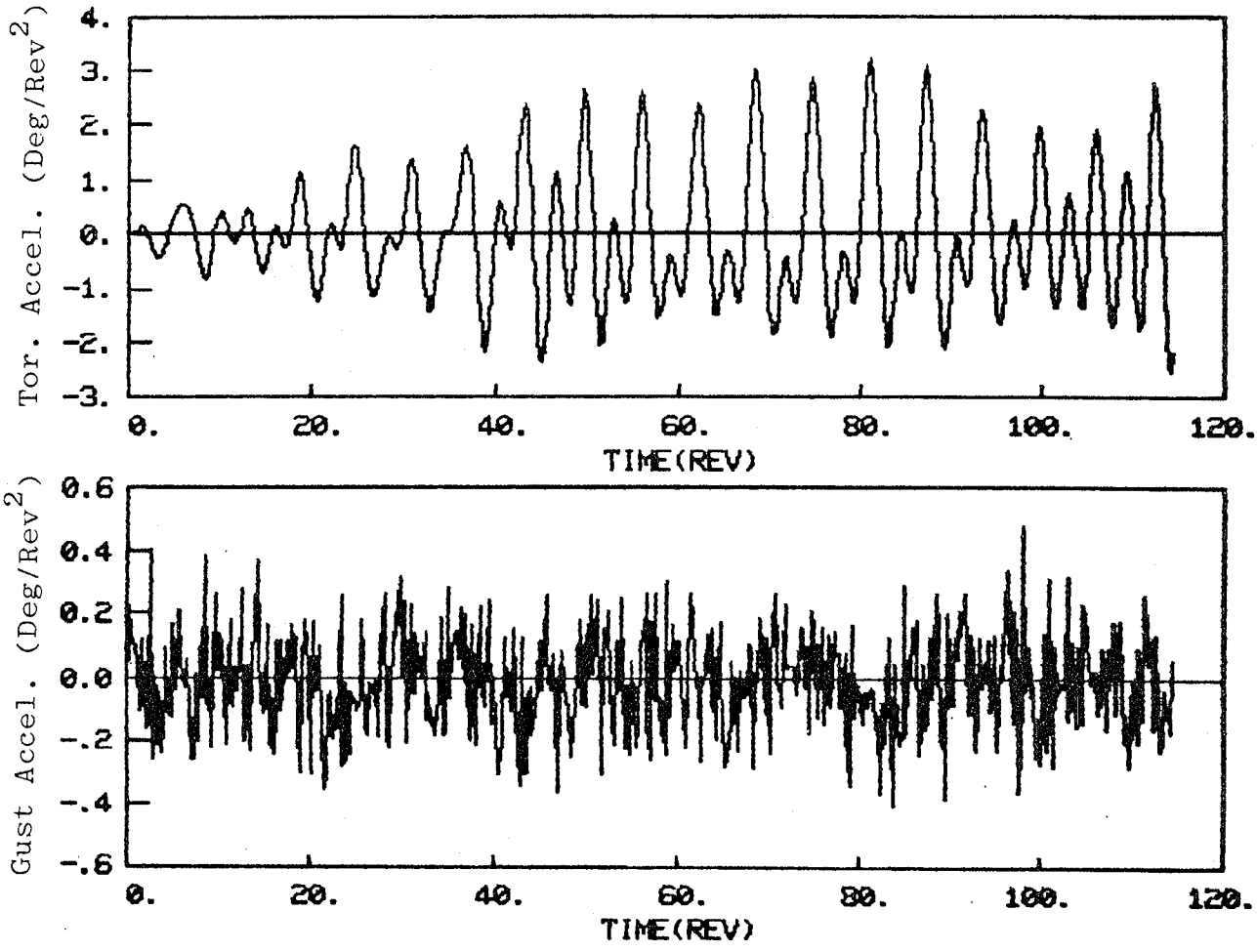


Figure 4-2. Lightly Damped Model Output and Input.
(σ_{gust} = .15 Deg/Rev^2)

4.2 RPEM AS A STARTUP ESTIMATOR

One of the desirable characteristics of the RPEM algorithm is the ability to start the estimation with no *a priori* information on the parameters. On short to intermediate data records with both process and measurement noise, however, overparameterization of the model aids in convergence of the flutter parameters considerably (see Section 4.4, where an 8-mode model was used to get excellent flutter identification with two active modes). In this section, where the simulation has no noise, the startup model is so good that overparameterization cannot measurably improve it. See Table 4-1 and Figure 4-3. Note that the residues of the artificial mode are negligibly small.

Table 4-1. RPEM Model Residue Comparison

True Residues	4-Mode Model Identified Residues	6-Mode Model Identified Residues
2.2683D-01	2.2683D-01	-6.5660D-09
-4.7332D-01	-4.7333D-01	-1.8139D-09
-1.1049D-01	-1.1049D-01	2.2683D-01
5.0268D-01	5.0270D-01	-4.7332D-01
		-1.1049D-01
		5.0268D-01

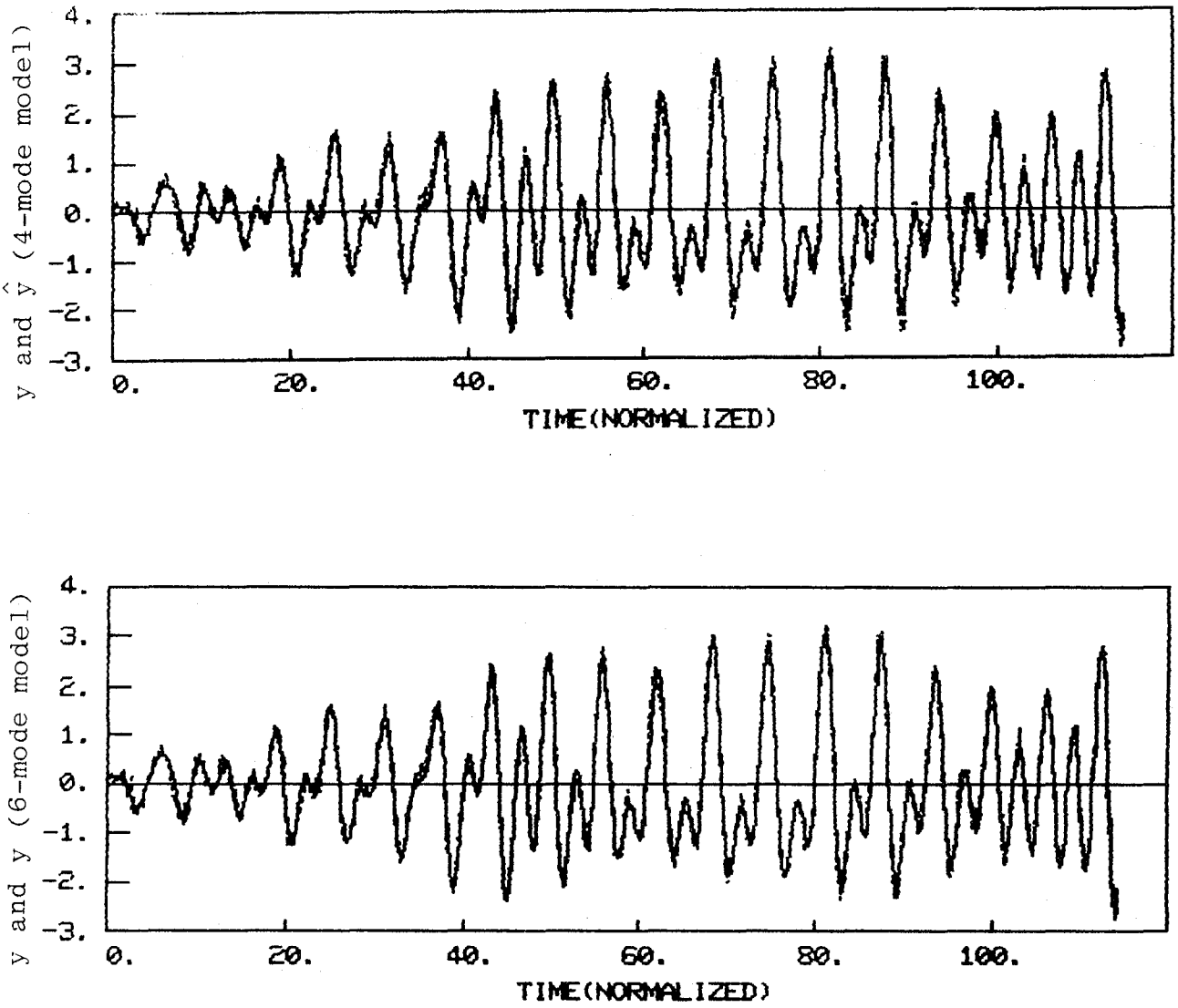


Figure 4-3. RPEM Performance Without and With Overparameterization.

These residues of the discrete transfer function correspond to the cosine and sine coefficients in a partial fraction expansion of a continuous transfer function. In other words, for

$$y(z^{-1}) = H(zI-F)^{-1} \quad G = \sum_{i=1}^{k} \frac{R_{c_i}(z+\sigma_i) + R_{s_i}\omega_i}{(z+\sigma_i)^2 + \omega_i^2} ,$$

R_{c_i} and R_{s_i} are the residues associated with the i-th mode where

$$H = [1\ 0\ \ldots\ 0], \quad F = \begin{bmatrix} & 1 & & \\ -\underline{a} & & \ddots & \\ & & & 1 \\ & & & 0 \end{bmatrix}, \quad G = \begin{bmatrix} \\ \underline{b} \\ \\ \end{bmatrix},$$

and

$$H(zI-F)^{-1} G = H_m(zI-J)^{-1} G_m ,$$

where H_m, J, and G_m are the modal form of the discrete equations. For the i-th mode the residues are

$$R_{c_i} = h_{1i} g_{i1} + h_{2i} g_{i2}$$

$$R_{s_i} = h_{1i} g_{i2} - h_{2i} g_{i1}$$

4.3 MAXIMUM LIKELIHOOD ESTIMATION WITH RPEM STARTUP PARAMETERS

To demonstrate the use of RPEM as a startup algorithm for maximum likelihood the first 100 data points were processed with a 3-mode (overparameterized) RPEM model. These parameters and the estimated output are equally as good as the outputs shown in Figure 4-4. These RPEM parameters are too accurate to reasonable test the maximum likelihood algorithm; therefore, an RPEM-identified set of parameters using the wrong input (shown in Table 4-1) were actually used.

As would be expected from the results of the previous section, the fit error of the final model is small, but the parameter estimates have not yet converged.

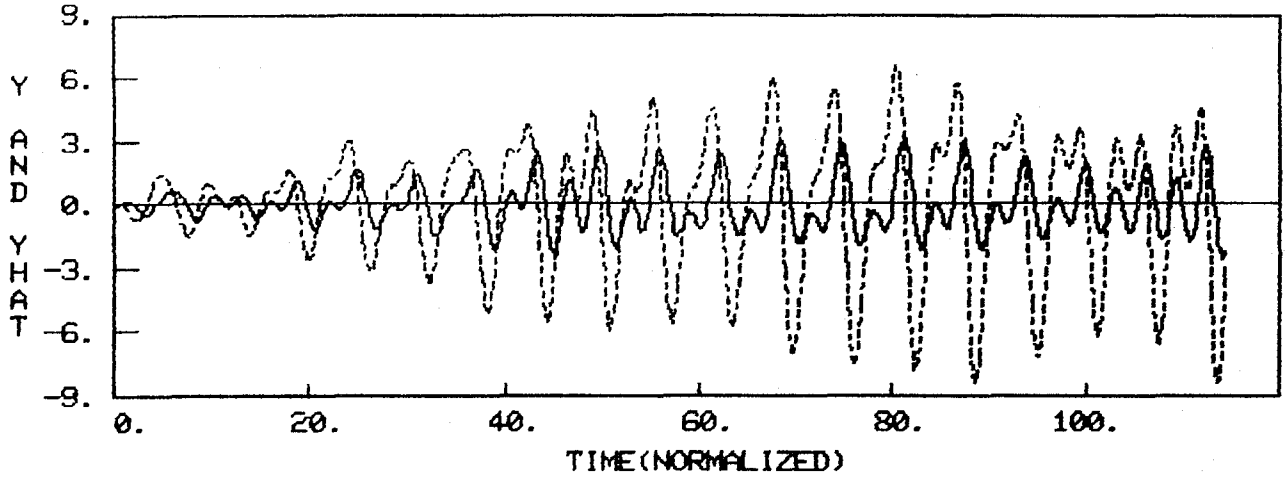


Figure 4-4. RPEM Start-Up Model. (Estimated from the first 100 points, but using an incorrect random input.)

θ_H	-0.4733	0.2268	0.5027	-0.1105
$\hat{\theta}_H$	-1.0679	-0.7230	0.3292	0.6849
	σ_1	ω_1	σ_2	ω_2
θ_F	0.9723	0.2268	0.8958	0.4420
$\hat{\theta}_F$	0.9737	0.2286	0.8985	0.4411

Table 4-2. Maximum Likelihood Start-Up Parameters.

The discrete θ_F parameters have reasonable startup values, however the mode shapes are essentially unknown.

The batch or semi-batch maximum likelihood algorithm described in Section 2.3 was applied to this 500 point simulation using the start-up values in Table 4-2. Figure 4-4 shows, first, the relative error in the natural frequencies on a linear scale and then the same error moduli are plotted on a logarithmic scale. The estimated variances of these parameters are also plotted with the actual errors. The algorithm has converged in four iterations. This particular simulation does not have any measurement noise added, which leads to the underestimate of variance after the modified Newton Raphson optimization has converged.

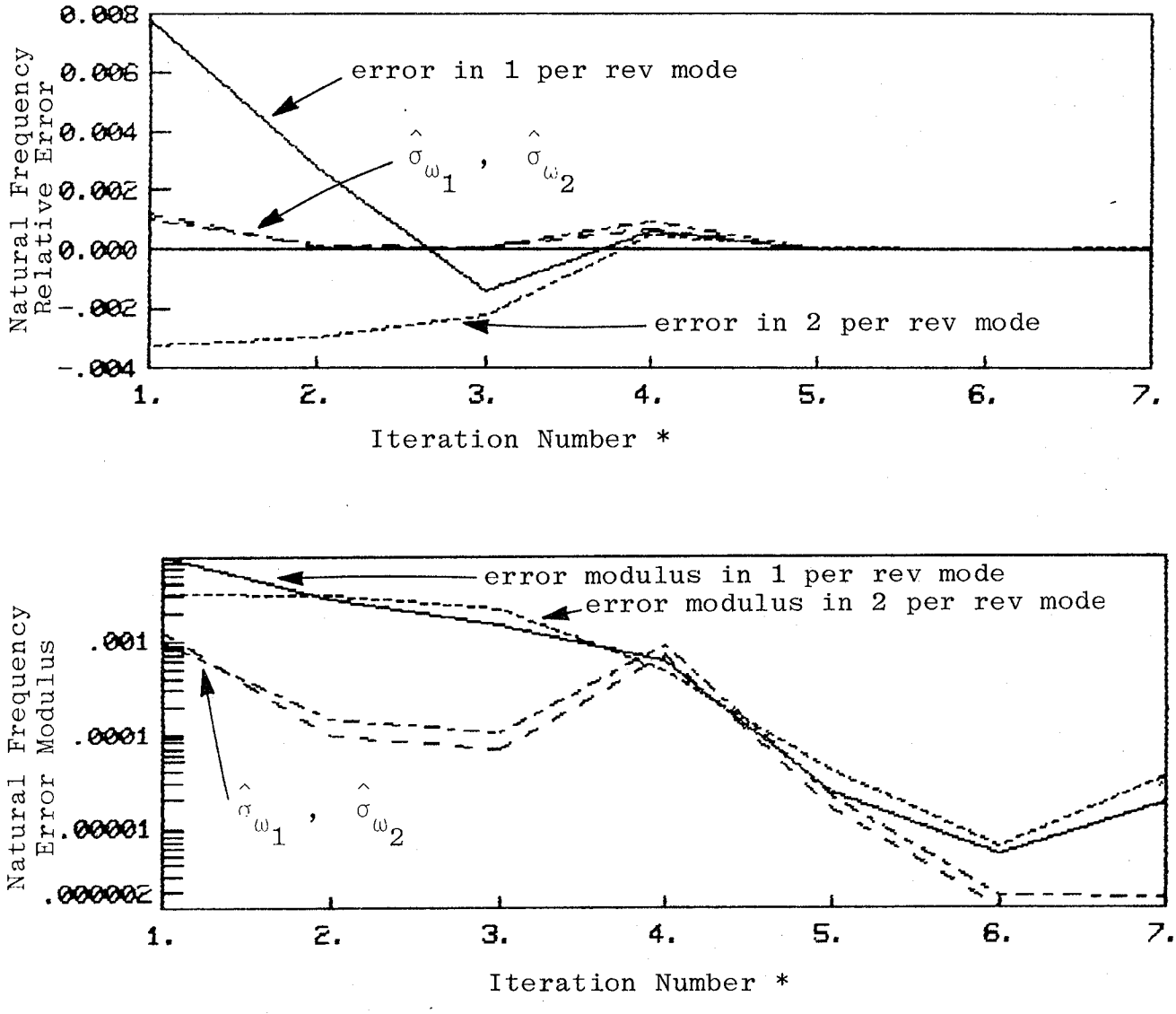


Figure 4-4. Maximum Likelihood Natural Frequency Errors (and Error Estimates)

---

* Note that the last iteration started from iteration five with the covariance reinitialized to large uncorrelated values.

The damping estimates show similar behavior to the frequencies, converging in four iterations, with reasonable error estimates at the convergence point. Further iterations decrease the parameter errors, but without measurement noise on the simulation, the error estimates underestimate the parameter error on iterations beyond the fourth iteration.

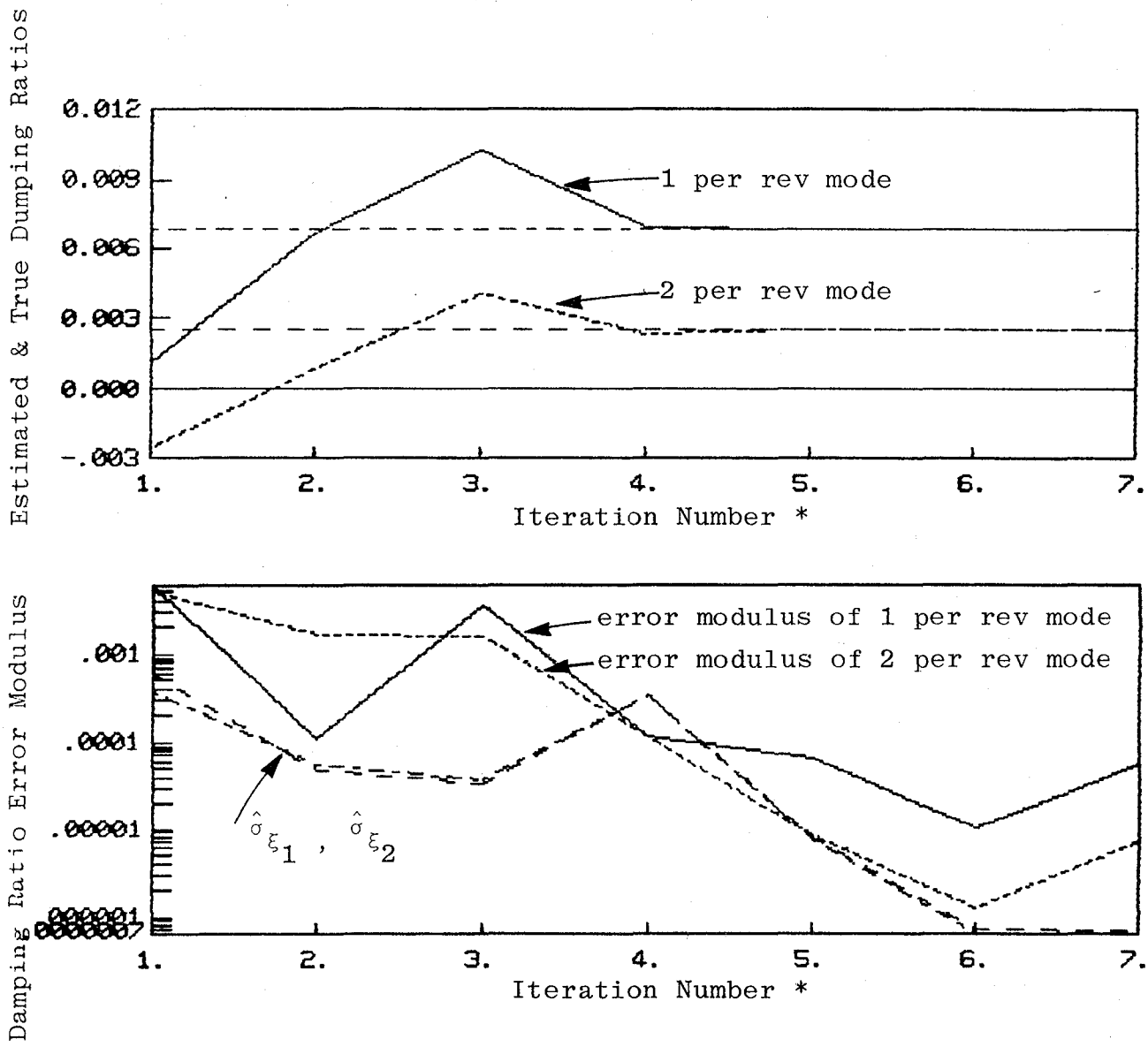


Figure 4-5. Maximum Likelihood Damping Ratio Errors (and Error Estimates)

---

* Note that the last iteration started from iteration five with the covariance reinitialized to large uncorrelated values.

Figure 4-6 shows the Torsion rate measurement and estimated output with the associated innovations or errors produced by the estimated model after six iterations with the covariance correction (shown as the seventh iteration on Figures 4-4 and 4-5).

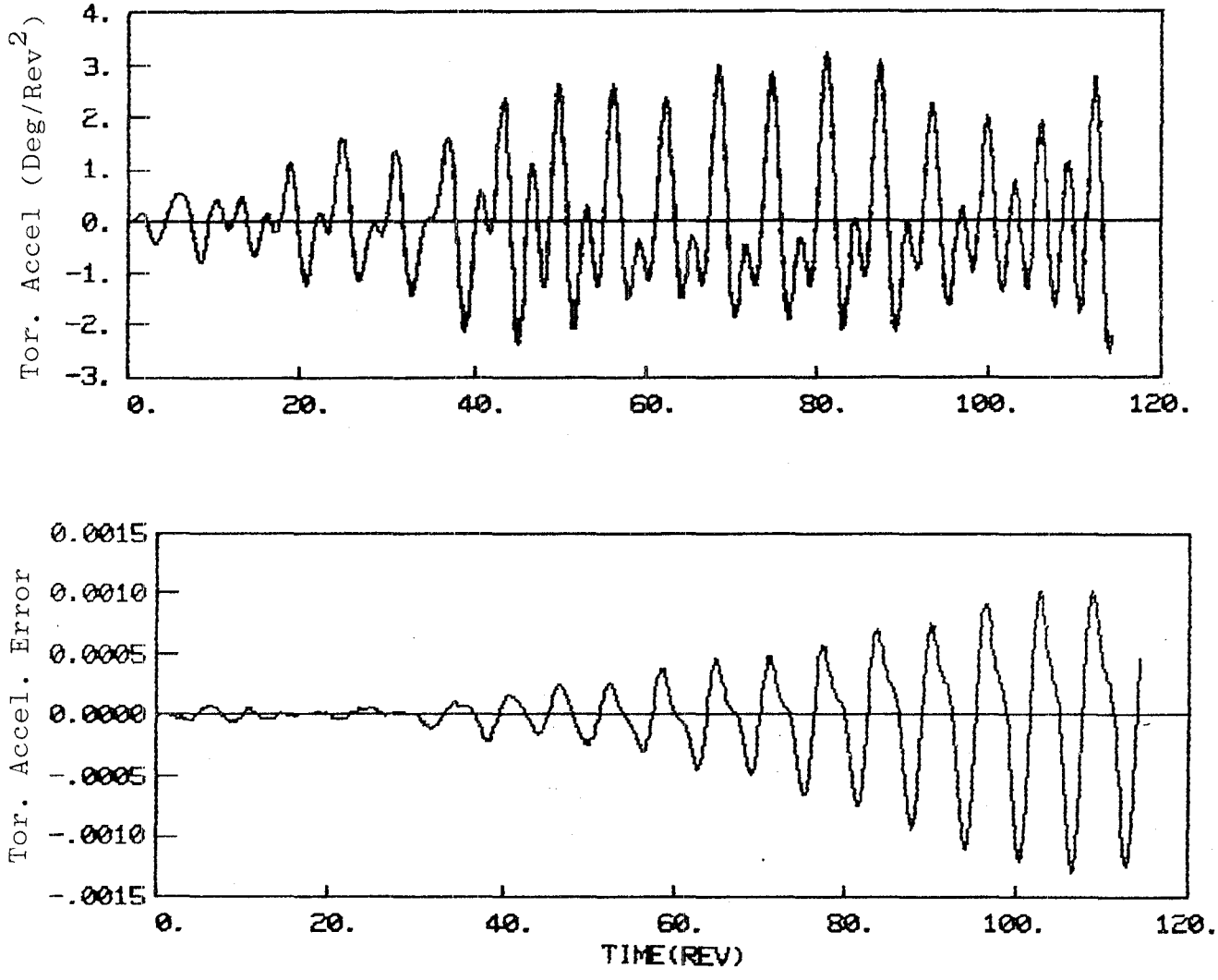


Figure 4-6. Torsion Acceleration Estimated Output and Output Error

4.4 RPEM IDENTIFICATION FROM SHORT, INTERMEDIATE AND LONG DATA RECORDS

To demonstrate the convergence, convergence rate and accuracy predictions of Section 3.3, simulations with two different models were performed. The input was treated as both known and unknown. Frequencies of 1 and 2 per rev (normalized frequency) were simulated in a model of the form

$$\dot{x} = \begin{bmatrix} 0 & 1 & & \\ -\omega_1^2 & -2\xi_1\omega_1^2 & & \\ & & 0 & 1 \\ & & -\omega_2^2 & -2\xi_2\omega_2^2 \end{bmatrix} x + \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} u \ , \quad \text{where}$$

$y = [0\ 1\ 0\ -.5]\ x$. The frequency response for the model ($\omega_1 = 1$, $\xi_1 = .0470$, $\omega_2 = 2$, $\xi_2 = .0480$) is shown in Figure 4-7. This is the same model used to study noise effects on achievable accuracy in Section 3.3. The same measurement white noise covariance (see Section 3.3) was used ($\sigma_v = .0707$), and gust covariance (σ gust = .05 deg) was used.

Modal parameters were identified with RPEM as a stand alone algorithm for short, intermediate, and long data records, rather than merely as a startup algorithm for semi-batch maximum likelihood. The overparameterized model had eight modes; the lowest frequencies are the simulated modes. Frequency and damping estimates and the Cramer-Rao bounds (computed from the true model) are compared to actual errors in Tables 4-3 through 4-5.

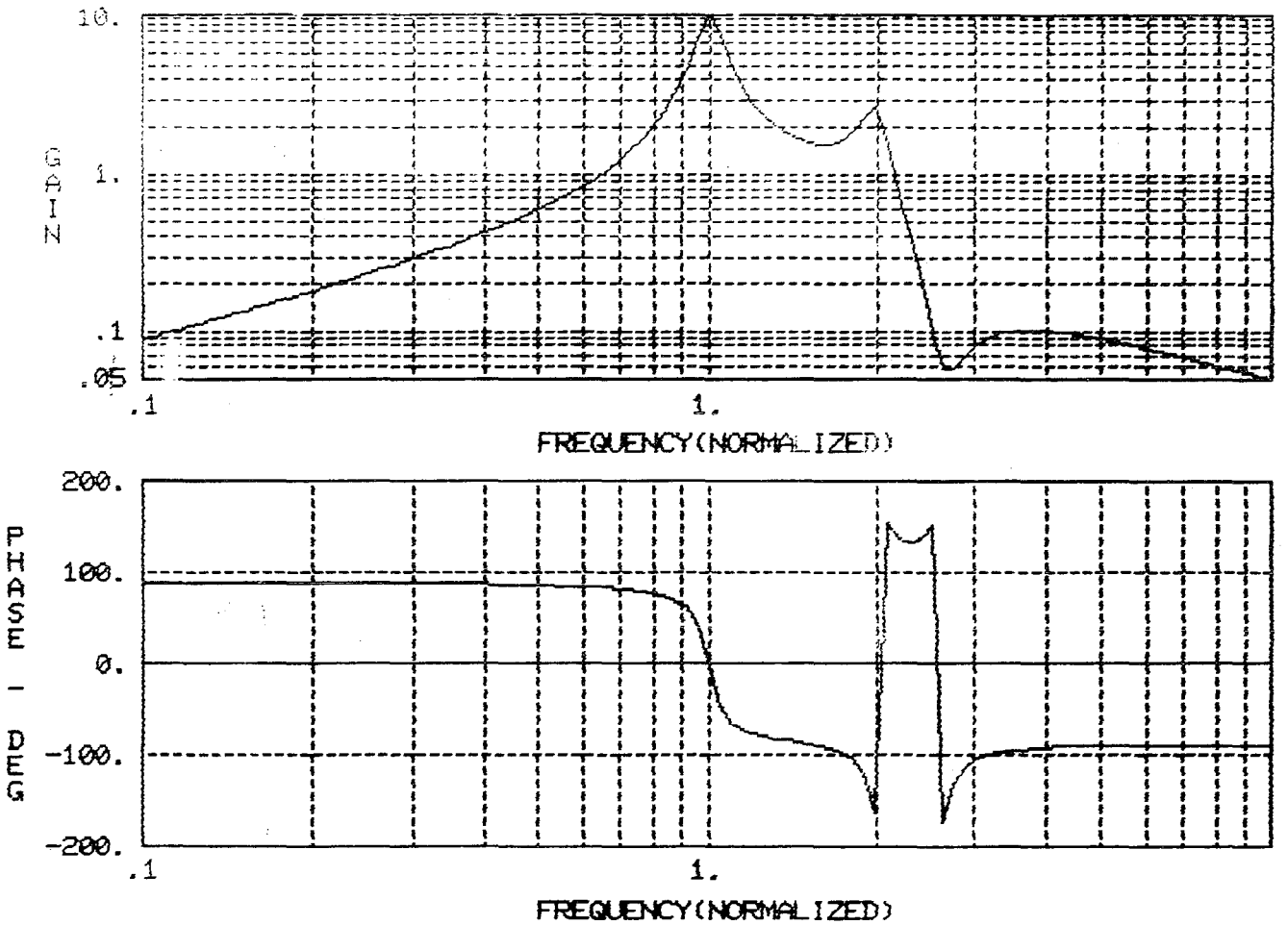


Figure 4-7. Two Mode Simulation - Damping ratios of 4.70% (for ω=1) and and 4.80% (for ω=2).

	ω_n	$\hat{\omega}_n$	$\sigma_{\omega_{n_{CR}}}$	$\xi(\%)$	$\hat{\xi}(\%)$	$\sigma_{\xi_{CR}}(\%)$
Known Input	1	.962	7.66_{10}^{-4}	4.70	10.4	.0784
	2	1.896	3.56_{10}^{-3}	4.80	.94	.174
Unknown Input	1	1.139	.124	4.70	45.2	12.8
	2	2.226	.464	4.80	93.7	23.6

Table 4-3. RPEM Frequency/Damping Estimates and Accuracy Estimates

(2-mode simulation of 125 data points. 8-mode model.)

	ω_n	$\hat{\omega}_n$	$\sigma_{\omega_{n_{CR}}}$	$\xi(\%)$	$\hat{\xi}(\%)$	$\sigma_{\xi_{CR}}\%$
Known Input	1	.9986	3.83_{10}^{-3}	4.70	4.34	.0392
	2	2.0101	1.78_{10}^{-4}	4.80	5.32	.0868
Unknown Input	1	1.1611	.0622	4.70	31.8	6.39
	2	3.525	.232	4.80	11.9	11.82

Table 4-4. RPEM Frequency/Damping Estimates and Accuracy Estimates

(2-mode simulation of 500 data points. 8-mode model.)

		ω_n	$\hat{\omega}_n - \omega_n$	$\sigma_{\omega_{n_{CR}}}$	$\xi(\%)$	$\hat{\xi}(\%)$	$\sigma_{\xi_{CR}}\%$
Known Input	1	.99963	1.71_{10}^{-4}	4.70	4.7083	.0175
	2	1.9990	7.95_{10}^{-4}	4.80	4.6663	.0388
Unknown Input	1	.9798	.0278	4.70	9.35	2.86
	2	3.366	.104	4.80	7.90	5.26

Table 4-5. RPEM Frequency/Damping Estimates and Accuracy Estimates

(2-mode simulation of 2500 data points. 8-mode model.)

The conclusions from these simulations both validate the performance of the algorithm RPEM for flutter mode identification with multiple modes and give insight into the validity of the results identified from the flight data in the next section. (The flight data also has two active modes with similar characteristics and also was overparameterized with an 8-mode model.)

The gust level is too low to identify the flutter modes with turbulence excitation anly. With a known input the algorithm finishes the convergence transient in about 750 data points or 1.5 seconds (see flight data results in Figure 5.15). For long data records (>4-5 seconds) RPEM gives extremely accurate estimates ($|\hat{\xi} - \xi| < .05\%$). For this particular simulation the damping estimate error (ω=1) is less than the Cramer-Rao bound, and for ω=2, the damping estimate error is less than $2\sigma_{CR}$.

This Page Intentionally Left Blank

SECTION 5

FLIGHT DATA RESULTS

Flutter-test flight data from the third flight of the ARW-1 vehicle of NASA's drones for aerodynamic and structural testing (DAST) program has been used to demonstrate the real-time flutter estimation algorithms. Three different maneuver types were processed:

1. A frequency sweep maneuver with significant excitation amplitude,
2. An intervening record between excitation inputs to emulate typical unknown input or turbulence excitation, and
3. The sequence of the last three pulses before the flutter incident resulting in failure of the right wing.

5.1 KNOWN INPUTS

Symmetric excitation with a logarithmic sweep from 10 - 40 hertz is shown in Figure 5-1.

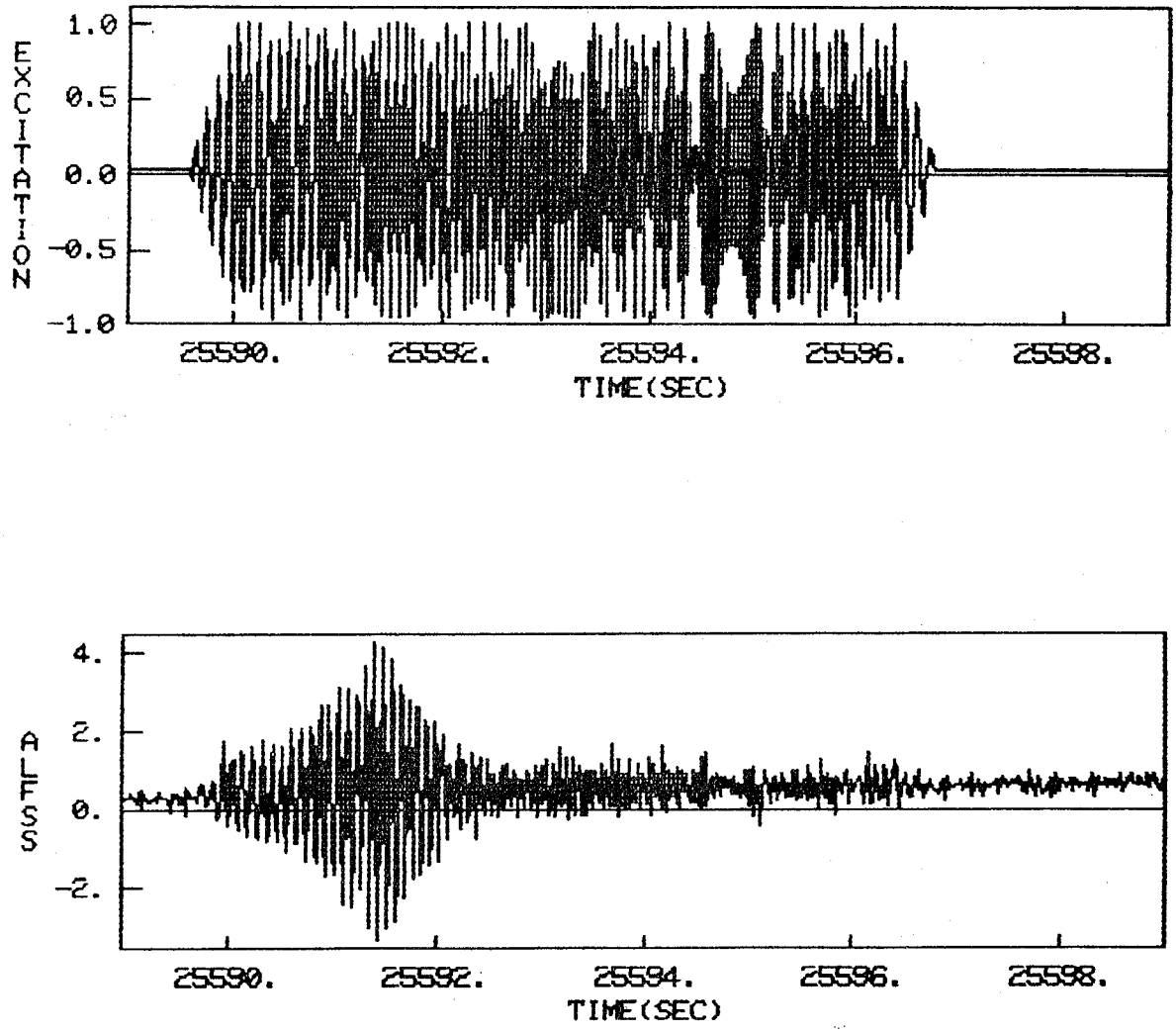


Figure 5-1. Symmetric Sweep Excitation Maneuver

The actual aileron positions include some asymmetric input due to the Flutter Suppression System (FSS), as can be seen in Figure 5-2.

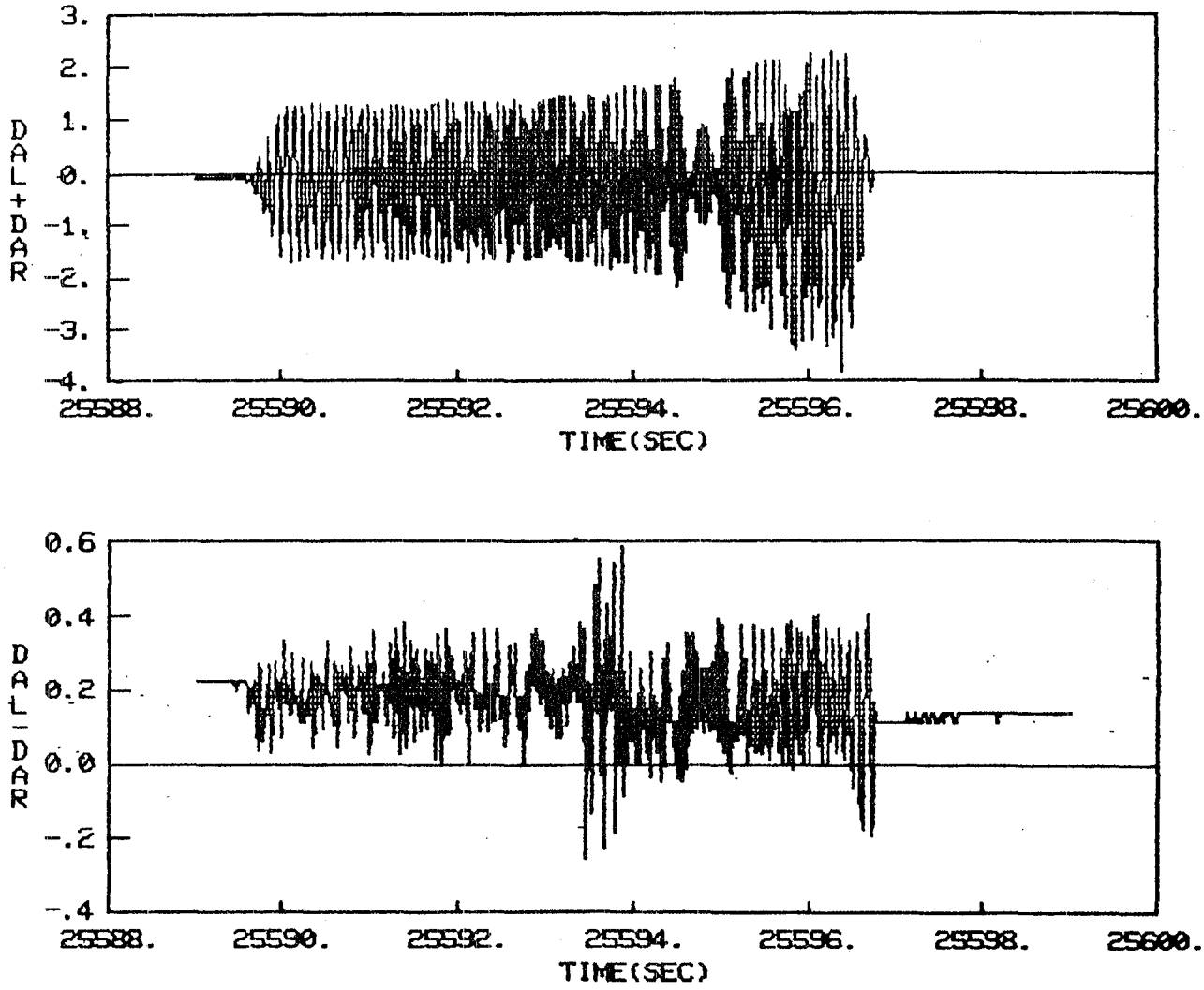


Figure 5-2. Symmetric and Asymmetric Inputs

The frequency responses of this input and the ALFSO accelerometer output are shown in Figure 5-3. The ratio of these two responses in Figure 5-3 is a coarse estimate of the transfer function, shown in Figure 5-4.

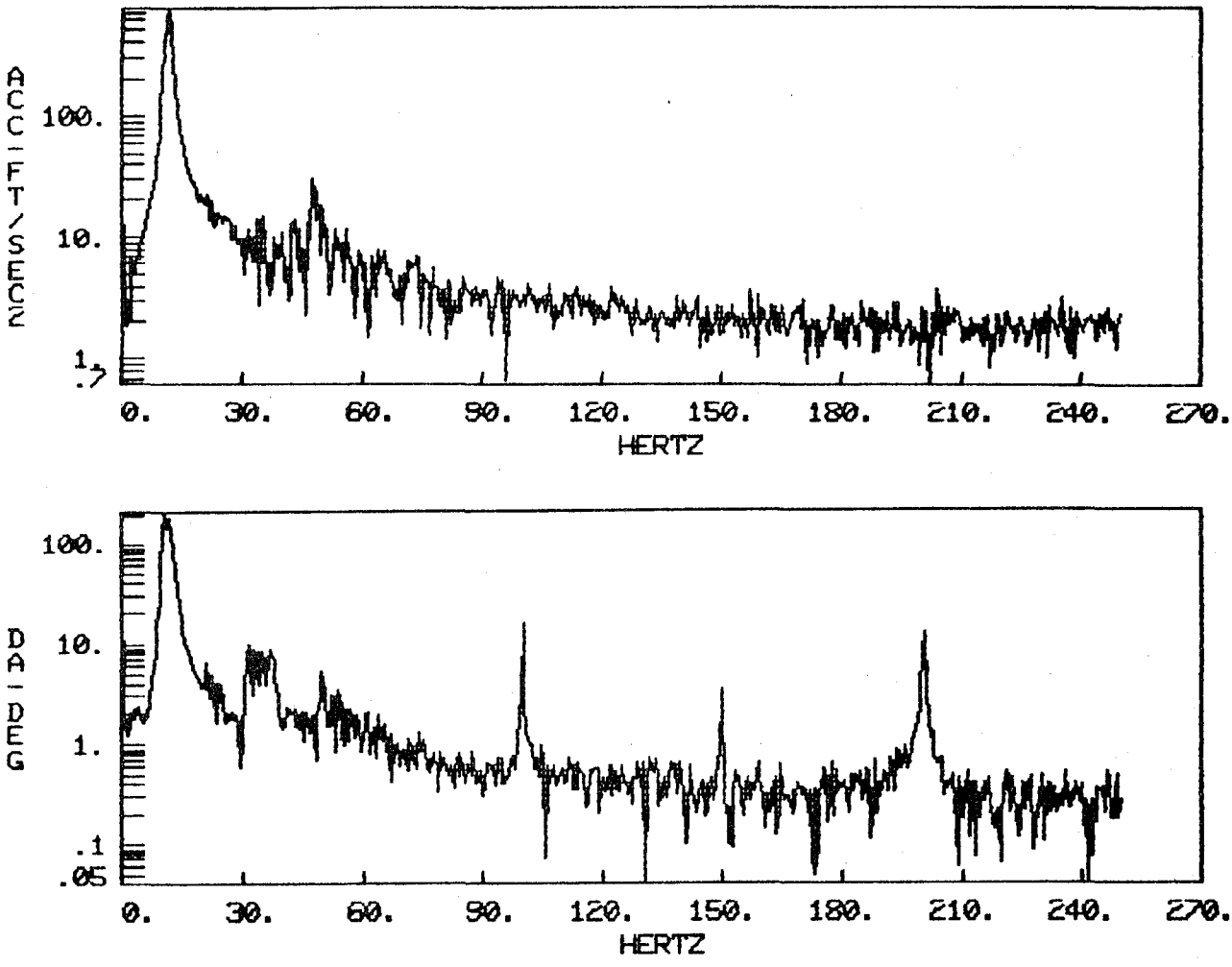


Figure 5-3. Output and Input Frequency Response

As can be seen from Figure 5-4, the noise is so great without averaging, that there is little recognizable information in any frequency range.

A single-input-single-output identification study was performed using the symmetric input and the ALFSO accelerometer with RPEM. Model orders of three, five and eight modes, with and without identifying the $C(z^{-1})$ polynomial (or equivalently the Kalman gain) are compared in Tables 5-1 and 5-2.

Table 5-1. Known Input Modal Estimates
(Estimation with $C(z^{-1})$ Polynomial)

Mode	3 Mode Model		5 Mode Model		8 Mode Model	
	ω_n(Hz)	ξ(%)	ω_n(Hz)	ξ(%)	ω_n(Hz)	ξ(%)
1	11.70	5.77	11.81	4.59	11.82	4.62
2	66.72	22.7	57.19	11.8	48.63	4.00
3	228.9	6.00	116.8	17.8	81.94	49.0
4			173.6	15.5	88.53	11.6
5			207.1	5.42	143.4	9.59
6					163.8	5.15
7					206.2	6.28
8					238.9	3.00

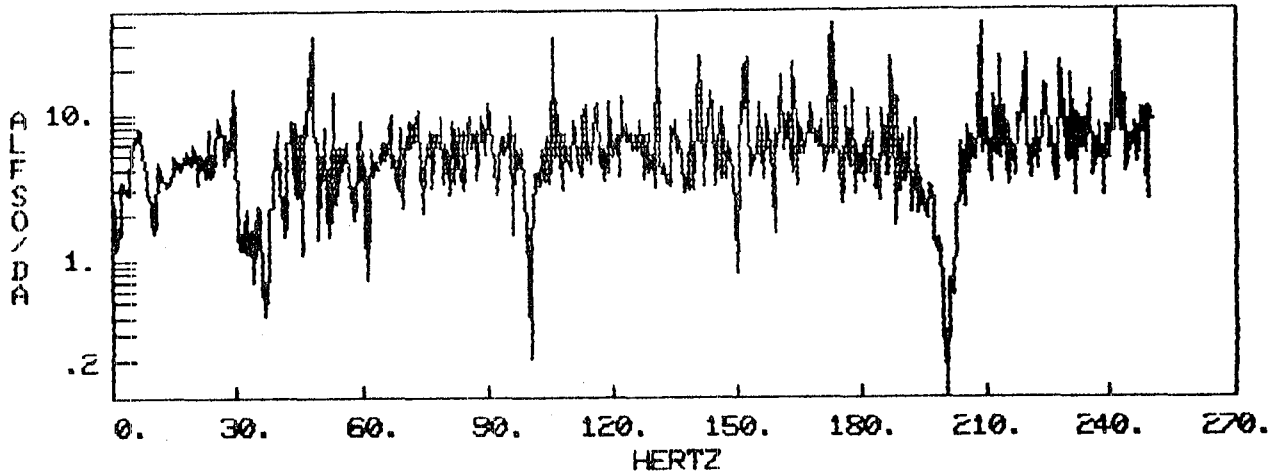


Figure 5-4. Coarse Estimate of $A_z(j\omega)/\delta_a(j\omega)$

Mode	3 Mode Model		5 Mode Model		8 Mode Model	
	ω_n(Hz)	ξ(%)	ω_n(Hz)	ξ(%)	ω_n(Hz)	ξ(%)
1	12.13	11.1	11.89	4.95	11.77	4.70
2	73.96	38.2	52.40	5.87	47.59	5.71
3	205.4	13.1	113.5	16.0	65.44	18.0
4			172.4	9.24	103.9	18.1
5			227.2	8.31	130.9	15.1
6					163.7	6.94
7					198.3	4.92
8					234.3	3.95

Table 5-2. Known Input Modal Estimates
(Estimation without $C(z^{-1})$ Polynomial)

Only the lowest mode is within the range predicted by NASTRAN analysis and ground vibration test. The frequency and damping estimate of this mode is comparable and reasonable for all six model structures identified except for model with the smallest number of parameters, the three mode model with no $C(z^{-1})$ polynomial.

The mode at approximately 11.8 hertz and damping of 4.6% is close to the predicted value of the symmetric first body bending mode at this flight condition (M = .7, h = 15 k feet). Gilyard and Edwards [14, Fig. 3] show predicted values of approximately ω_n = 11.8Hz with a damping ratio of 4.7%.

Note that here we are using the symmetric input $(\delta_{a_L} + \delta_{a_R})/2$ as the single input and hence identifying the open loop plant even though there is some asymmetric input. The

asymmetric input power is approximately 28 db down from the symmetric input (see Figure 5-2). Models with the $C(z^{-1})$ polynomials provide a more desirable model structure, able to model this unknown input.

There are several conclusions on the use of RPEM for real-time flutter analysis with known inputs:

1. Overparameterization is important to accurately identify the active flutter mode.
2. Overparameterization is even more important if the $C(z^{-1})$ polynomial is not identified with a known input.

5.2 UNKNOWN INPUTS

One desirable characteristic of a real-time flutter identification system is to identify the dominant system dynamics without a known input for continuous test monitoring. This can be accomplished with the RPEM algorithm by identifying the $A(z^{-1})$ and $C(z^{-1})$ polynomials and letting $B(z^{-1})=0$, or equivalently identifying the system dynamics matrix and the Kalman gain directly. Figure 5-6 shows a section of data early in the flight, between a sweep and an excitation pulse.

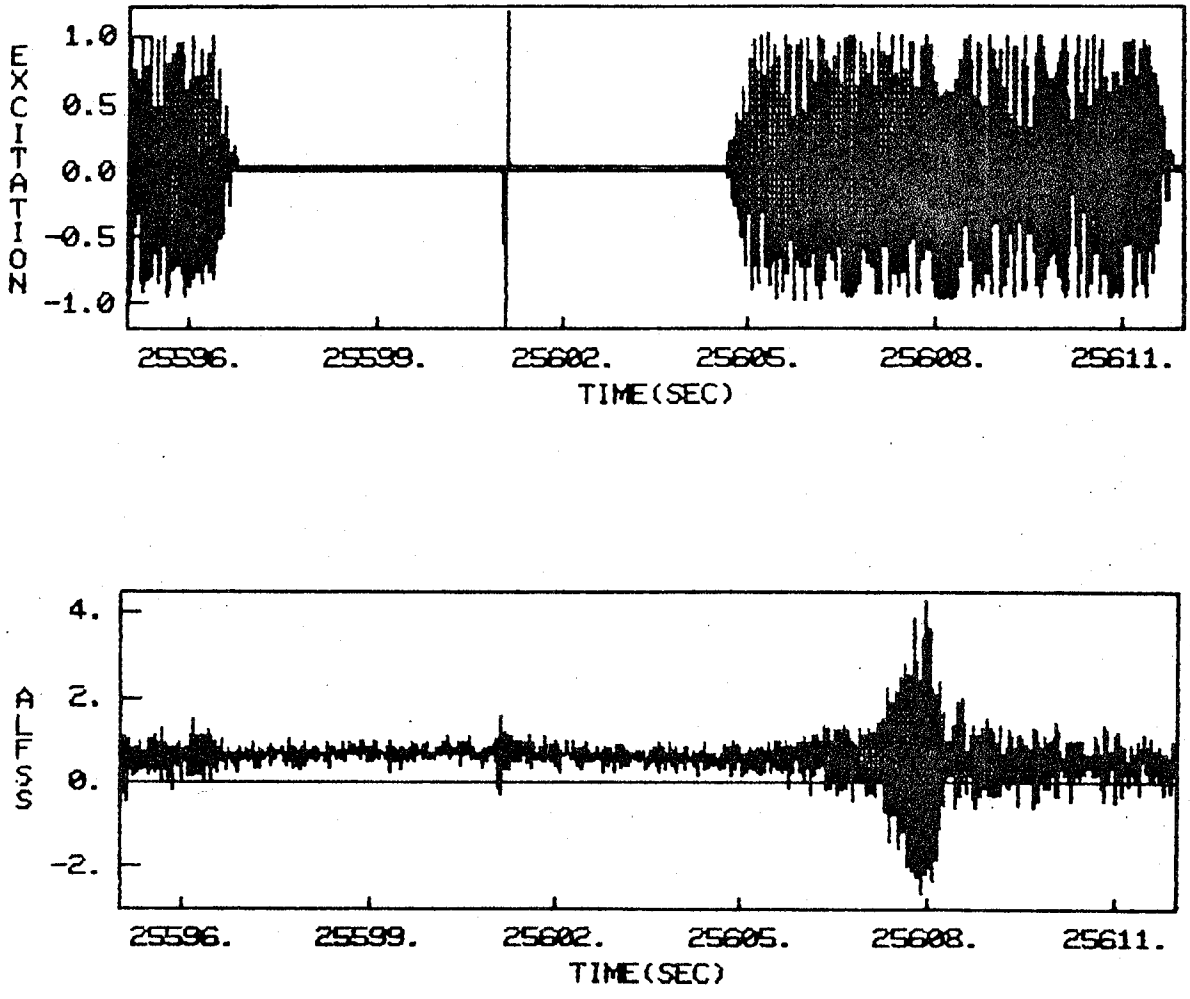


Figure 5-6. Inter-Excitation Data Section

The ALFSO sensor output for this interpulse section is shown in Figure 5-7.

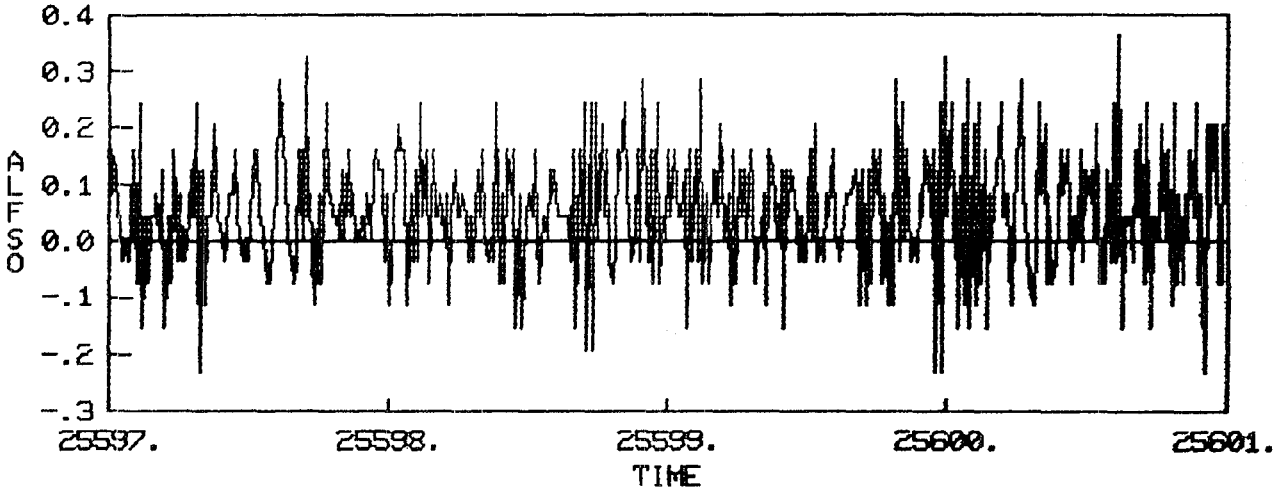


Figure 5-7. Turbulence Excited Accelerometer Output

Two thousand data points were used to estimate the 48 parameters and modal estimates of Table 5-3.

Mode No.	ω_n(Hertz)	ξ(%)
1	26.08	70.1
2	50.07	4.38
3	68.57	20.3
4	140.3	4.31
5	171.1	2.71
6	207.8	8.13
7	235.8	1.31

*

Table 5-3. Unknown Input Model Estimates (ALFSO Accelerometer Used as Recorded)

Besides the peaks at 26 and 50 hertz identified above, the ALFSO frequency response (see Figure 5-8) shows peaks at 12, 15, 20, and 42 hertz as well.

* Eighth discrete mode not realizable as a real continuous system.

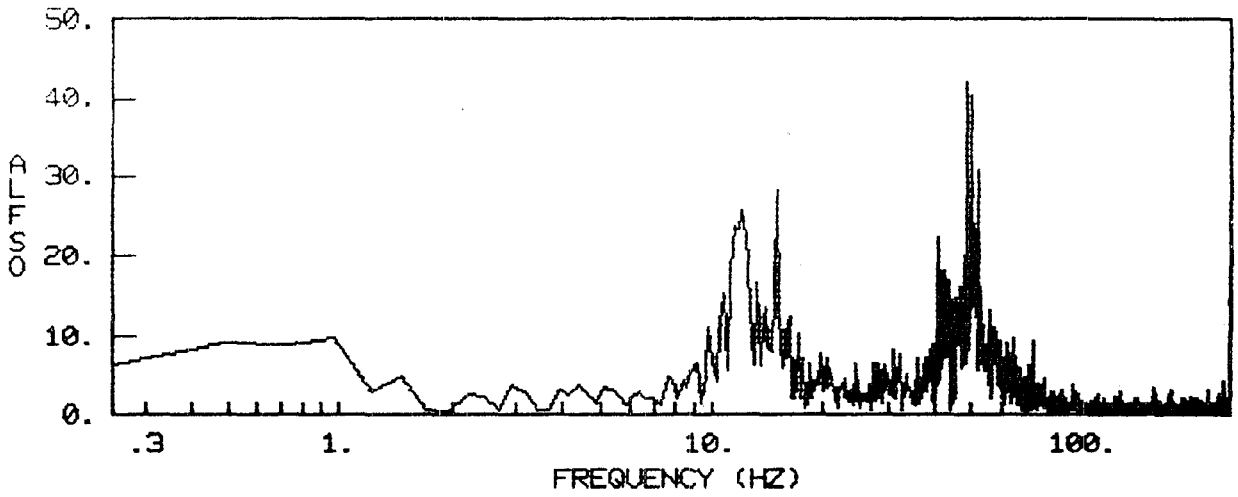


Figure 5-8. ALFSO Accelerometer Spectrum Forced by Turbulence

This estimation was affected by the bias in the accelerometer, which when taken out gives the modal estimates of Table 5-4.

Mode No.	ω_n(Hertz)	ξ(%)
1	14.51	24.0
2	46.39	13.6
3	51.43	7.92
4	87.79	8.99
5	105.5	4.13
6	149.1	6.94
7	191.3	3.94
8	233.4	.46

Table 5-4. Unknown Input Model Estimates (ALFSO Bias Removed Without Estimation of $B(z^{-1})$ Polynomial)

The lowest frequency is reasonable, but its damping is too high. The estimate errors and error spectrum are given in Figure 5-9.

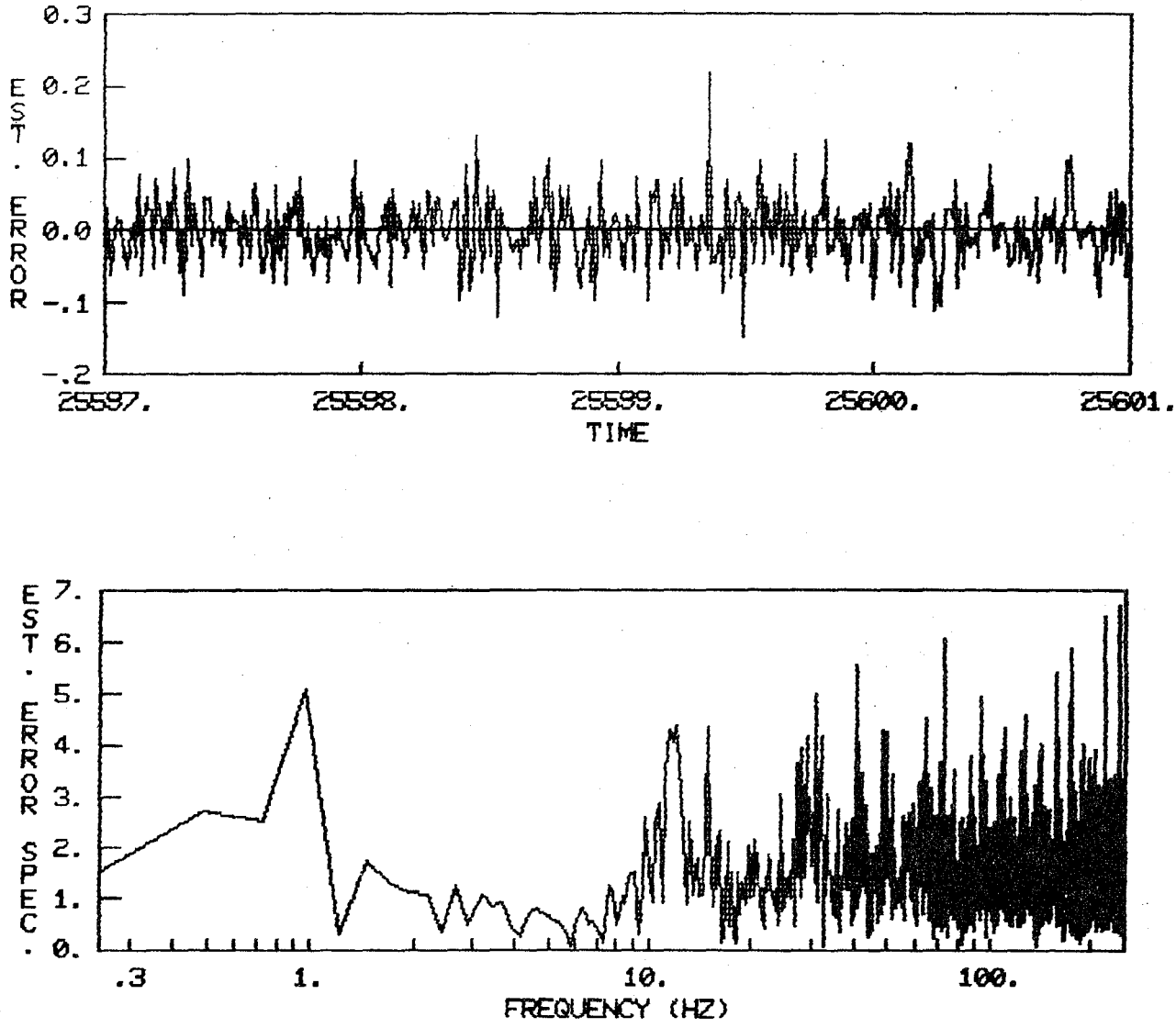


Figure 5-9. Estimate Errors and Error Spectrum ($A(z^{-1})$ and $C(z^{-1})$ Polynomials Estimated)

It is evident that the effect of the low frequency modes has not been completely removed. The approximate signal-to-noise ratio is

$$SNR = (\sigma_y^2 - \sigma_e^2) \; / \; \sigma_e^2 = 5.77 \; ,$$

so that the turbulence level is quite low.

It is interesting to examine the excitation channel more closely (see Figure 5-10), which shows what could have been noise in digitization, or noise actually transmitted and passed through the control system.

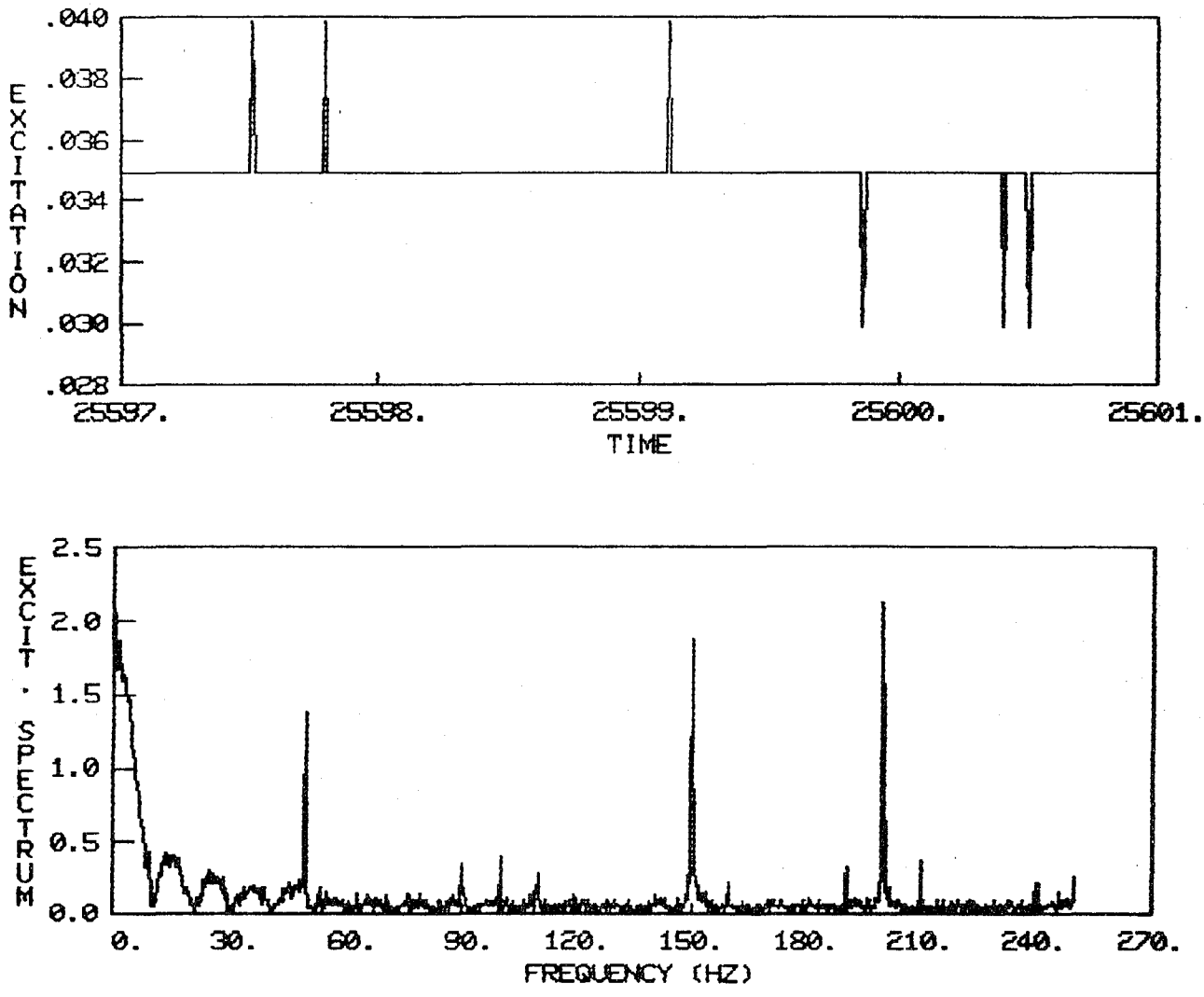


Figure 5-10. Interpulse Excitation and Excitation Spectrum

The A, B, and C polynomials were identified using this input (a 48 parameter model) and the resulting modal parameters are given in Table 5-5.

Mode No.	ω_n(Hertz)	ξ(%)
1	13.02	9.39
2	48.81	5.90
3	65.33	16.0
4	103.1	15.1
5	137.6	10.2
6	180.0	7.09
7	221.0	6.81

*

Table 5-5. Unknown Input Modal Estimates (ALFSO Bias Removed, with Estimation of $B(z^{-1})$ Polynomial)

The lowest natural frequency estimate is still reasonable for this flight condition ($M \simeq .7$, $h = 15$ k feet) with the damping estimate much more reasonable, for the first bending mode.

The spectrum of the estimated model errors (Figure 5-11) does look like nearly white noise, removing the peak in the region of the first symmetric mode.

The estimated signal-to-noise ratio is slightly larger at $SNR \simeq 5.79$.

---

* Eighth mode not realizable as a real system.

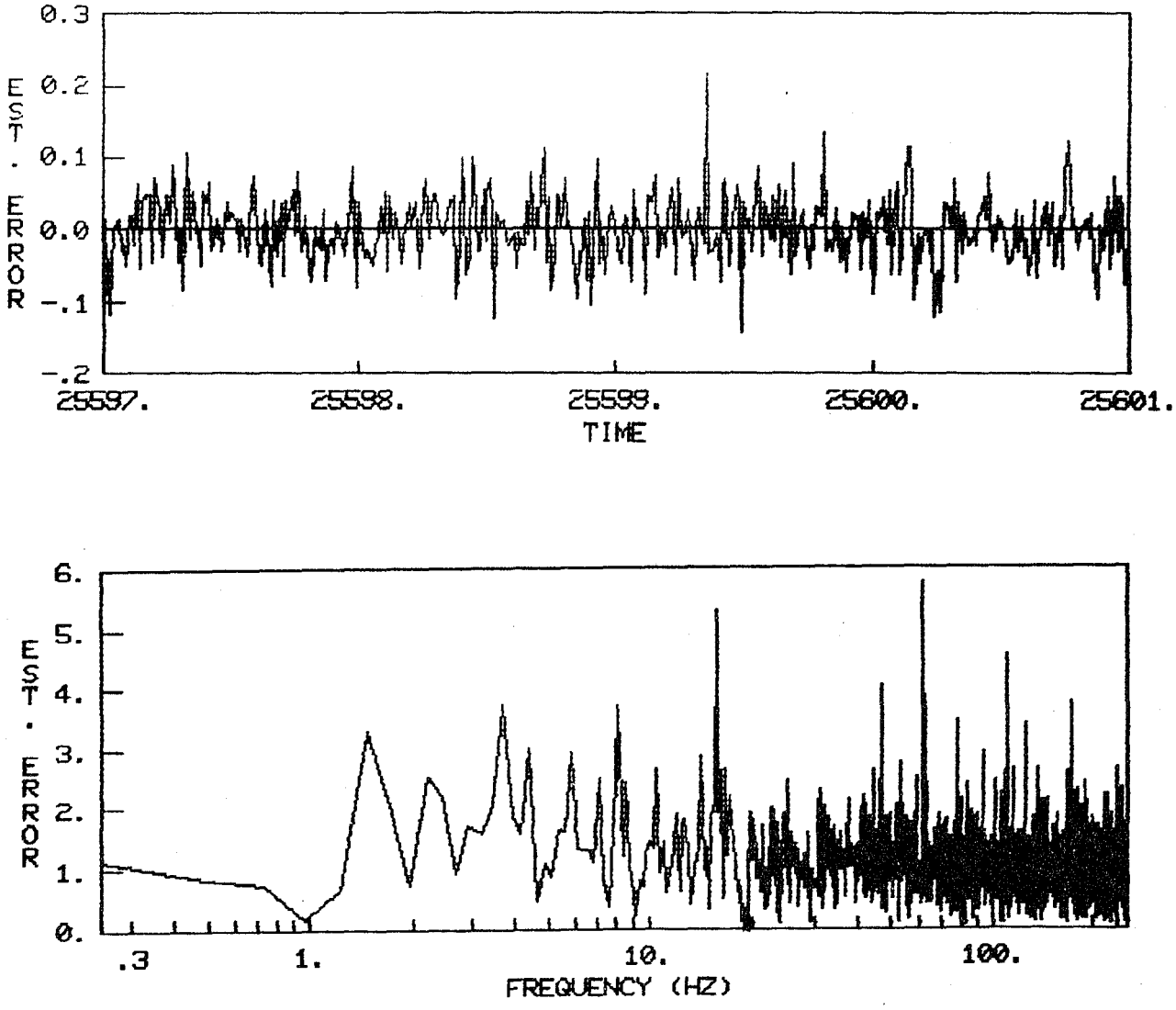


Figure 5-11. Estimate Errors and Error Spectrum ($A(z^{-1})$, $B(z^{-1})$, and $C(z^{-1})$ Polynomials Estimated)

The conclusion from this flight data with a very low level of turbulence or input excitation is consistent with the conclusions on simulated data in Sections 4.4 and 3.3.2. The excitation level has a very significant effect on estimation accuracy and, of course, the proportion of the input signals that are known significantly affect the parameter accuracy as well.

5.3 FLUTTER INCIDENT RESULTS

During the third flight of the ARW-1 program, the Flutter Suppression System (FSS), the on-board stabilizing control system, was actually operating at one-half the nominal gain. During the acceleration from $M = .80$ to $M = .825$ the first wing-bending asymmetric mode showed increasingly lighter damping, finally resulting in structural failure of the right wing, and together with a partial failure of the parachute recovery system, a crash of the vehicle [14].

The excitation (FSSEXC channel) and ALFSO accelerometer response are shown in Figure 5-12. The last three excitation pulses are full cycle sine wave doublets (chopped here because only every 20-th point has been plotted). Just over twelve seconds (6290 data points) were processed as a single record, up to the last point before accelerometer saturation.

Figure 5-13 shows the aileron positions for this final part of the flight, while Figure 5-14 shows their average sum and difference for the remaining linear portion of the flight.

It is noticeable that the symmetric and asymmetric inputs have comparable input magnitudes due to the flutter suppression asymmetric feedback attempting to stabilize the anti-symmetric modes.

At approximately $T = 26138.5$ seconds, the test pilot was instructed to terminate the test and the throttle was reduced due to the light damping observed in the previous time traces

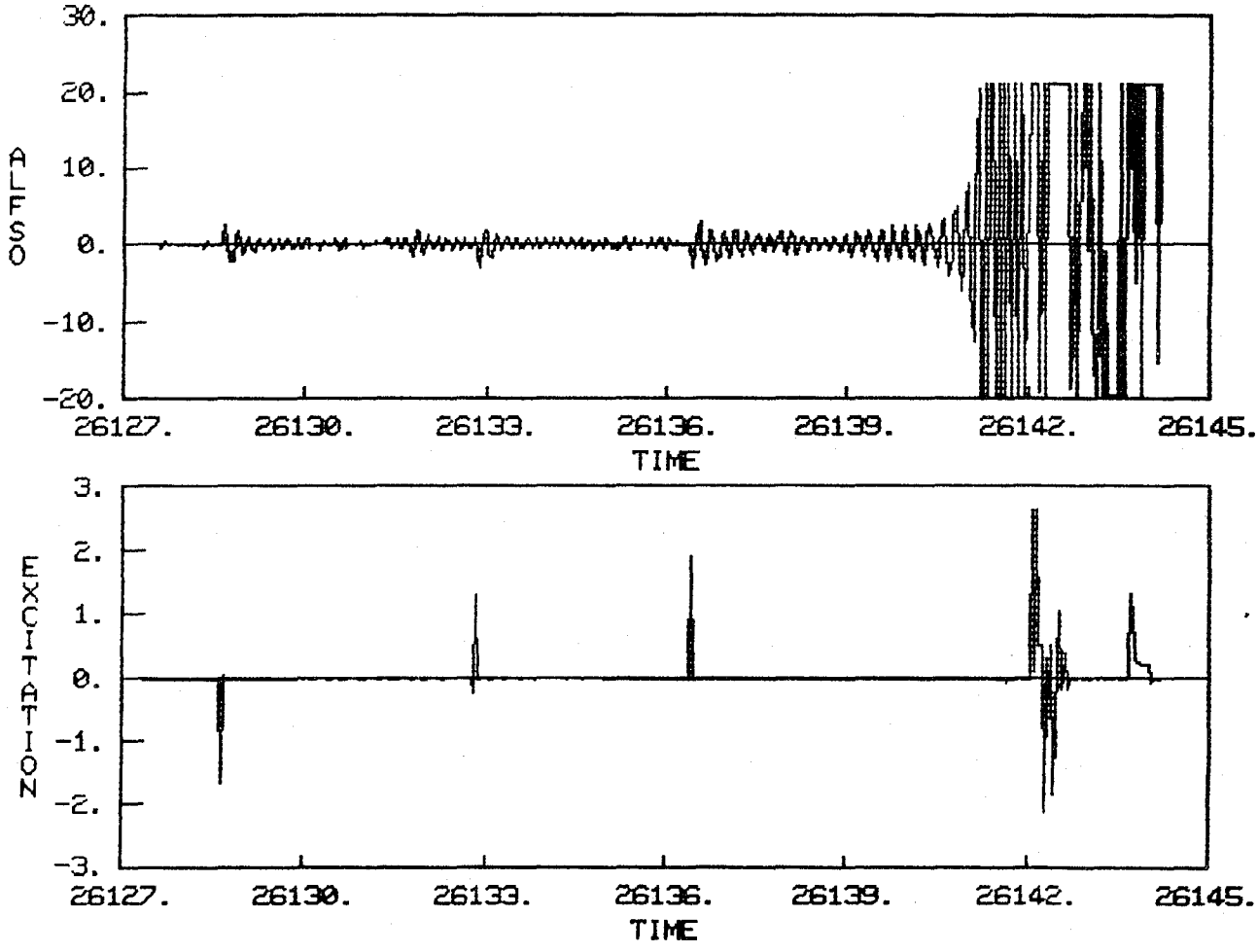


Figure 5-12. Flutter Incident Excitation and Response

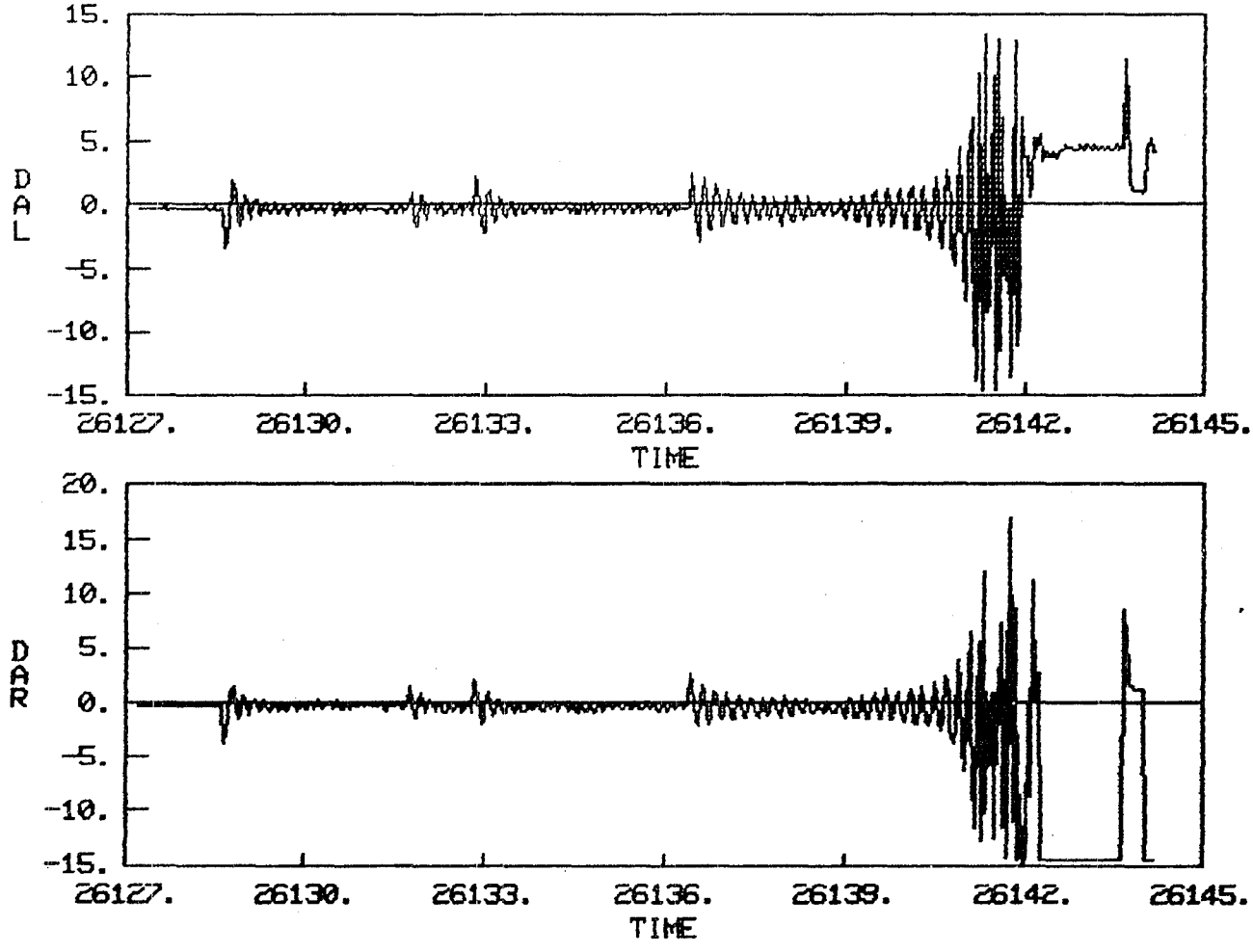


Figure 5-13. Aileron Positions for the End of the ARW-1 Flight 3

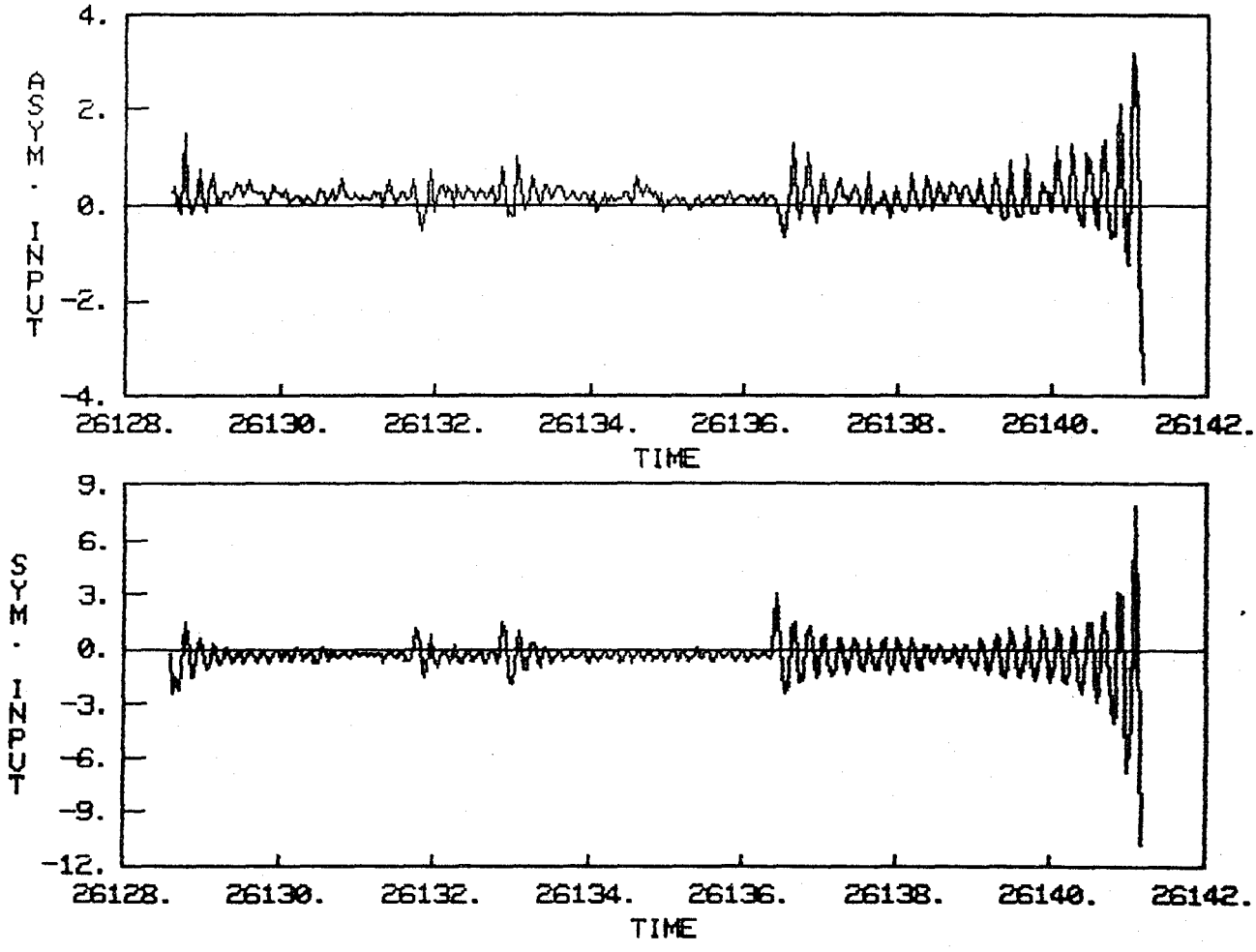


Figure 5-14. Inputs for the Last Linear Portion of the DAST Flight 3

[14]. The real-time damping estimator indicates a significant reduction trend in damping, approximately two seconds before this, as seen in Figure 5-15. Such a serious warning, this early, is quite difficult if not indiscernable from the time trace shown in Figure 5-12. The last damping estimate (modal parameters were output every 100 points or .2 seconds, using the last 190 points for the last estimate) is $\xi = -.02218$ for the symmetric wing bending mode at 19.16 Herz. Before the drop in frequency at the flutter condition (see Figure 5-15), the flutter mode frequency was estimated as $\hat{\omega}_n = 125$ rad/sec, which correlates exactly with post flight processing results by other techniques (see Gilyard [14, Figure 12]).

Figure 5-15 shows the initial convergence period for RPEM of 0.5 - 1.0 seconds. It is possible that the damping dip at T = 26132 seconds is still due to convergence since some of the other discrete modes modeled (an eight mode model) changed at this point (after 1500 data points). The break up in natural frequency and clear reduction trend in the damping at $T \cong 26136.5$ seconds is definitely due to the changing flutter characteristics. All the modes modeled in the overparameterized system are changing smoothly at this point.

It should be pointed out that the asymptotic forgetting factor in RPEM can be fixed at less than one to make the algorithm track models with changing characteristics with higher accuracy. Giving the operator the option to select a past window size of considered data is a possibility when real-time implementation is performed.

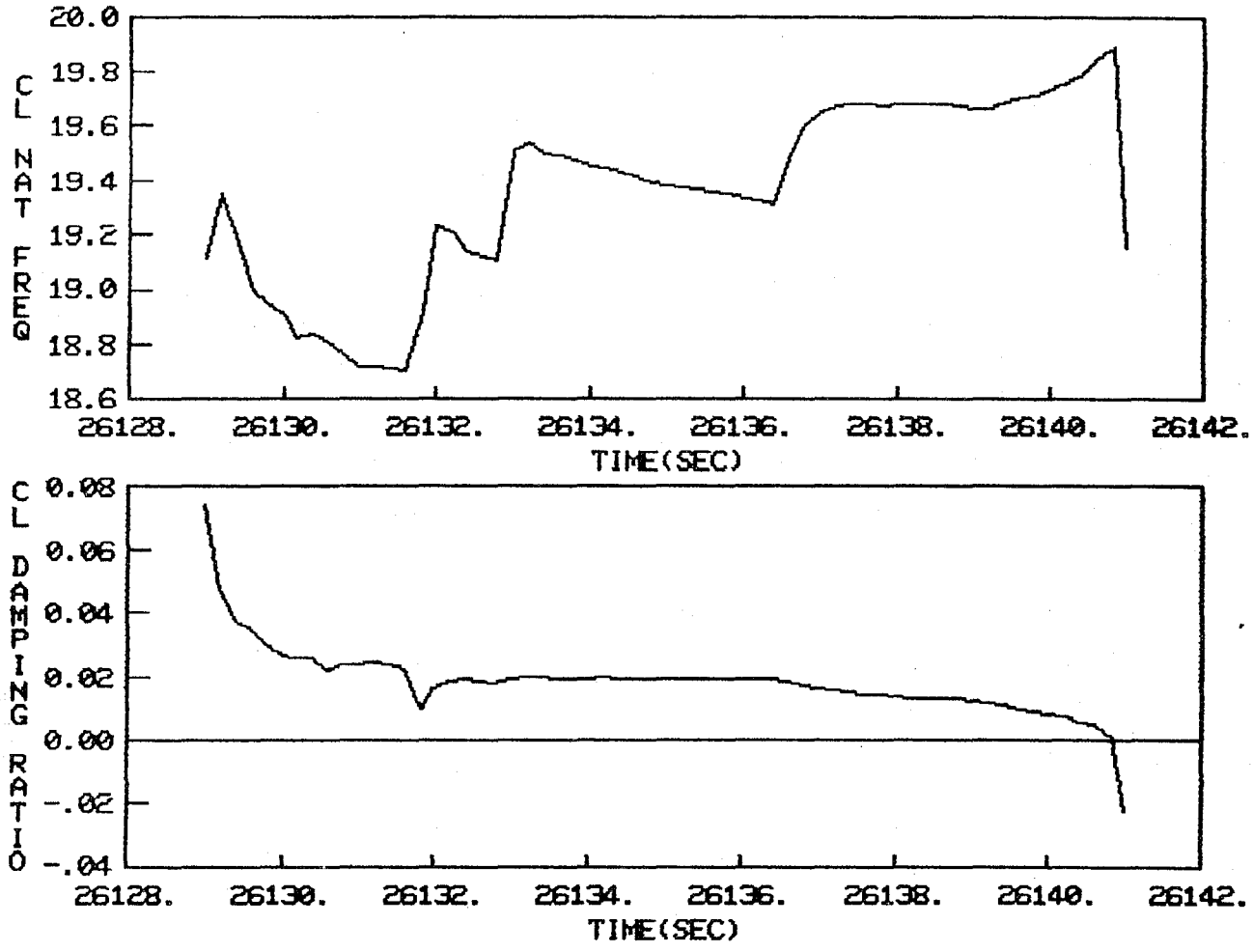


Figure 5-15. Estimated Frequency and Damping of the Flutter Mode (End of the ARW-1 Flight 3)

SECTION 6

NEAR REAL-TIME FLUTTER ANALYSIS (NRTFA) PROGRAM DOCUMENTATION

The delivered ANSI FORTRAN-77, Near Real-Time Flutter Analysis Program (NRTFA) is so named because, while providing a working tool to analyze real time flutter analysis algorithms, it lacks the real-time executive, real-time input/output buffered interfaces and real-time graphics drivers. Thus the program reads a large data block, then emulates real-time operation by repeatedly outputting new frequency and damping estimates to the operator console screen. This section

1) Explains the choice of identification algorithm implemented in NRTFA,

2) Documents the program structure and subroutines, as well as fixed algorithm initialization parameters,

3) Describes the memory resources and timing comparisons for typical flutter models,

4) Discusses extensions necessary for actual real-time implementation, and

5) Gives installation instructions.

User information is documented in Appendix A, and Appendix B gives the FORTRAN listings.

6.1 THE NRTFA ALGORITHM

Studies documented in the previous three sections support the conclusion that the Recursive Predictive Error Method (RPEM) works extremely well as a stand alone algorithm or as a parameter

start-up routine for a block-recursive maximum likelihood (ML) algorithm. The block-recursive ML technique gives excellent results where several passes can be made over the data block. This may not be acceptable for the timing requirements of a real-time algorithm. Implementation of a block recursive algorithm with a single pass over the data could enhance identification speed over pure recursive methods; however, the convergence characteristics of such an algorithm require more research. Therefore, the NRTFA program was configured with RPEM as a stand alone algorithm which could be restarted at any point by the operator.

The equation error, standard ARMA or full ARMAX models can be selected by the user (see Appendix A), as described in Section 2.

6.2 THE NRTFA PROGRAM

The NRFTA subroutine calling structure is given in Figure 6-1.

The NRTFA program structure is shown in Figure 6-2.

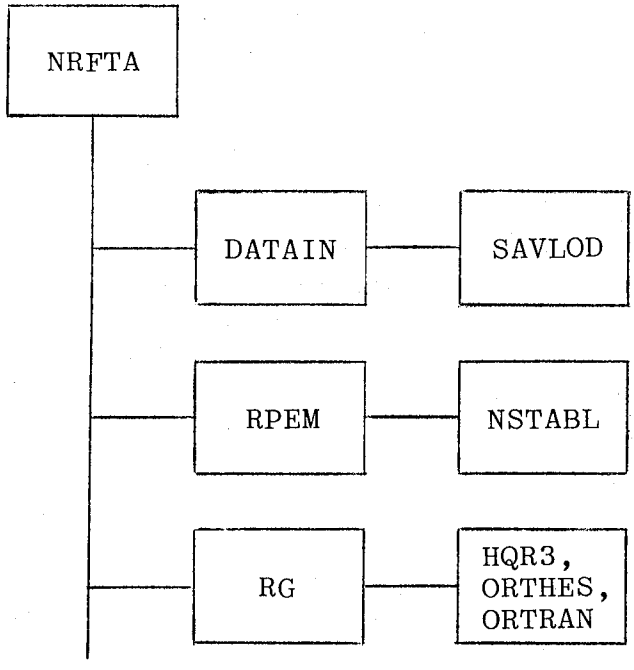


Figure 6-1. Near Real-Time Flutter Analysis Calling Sequence.

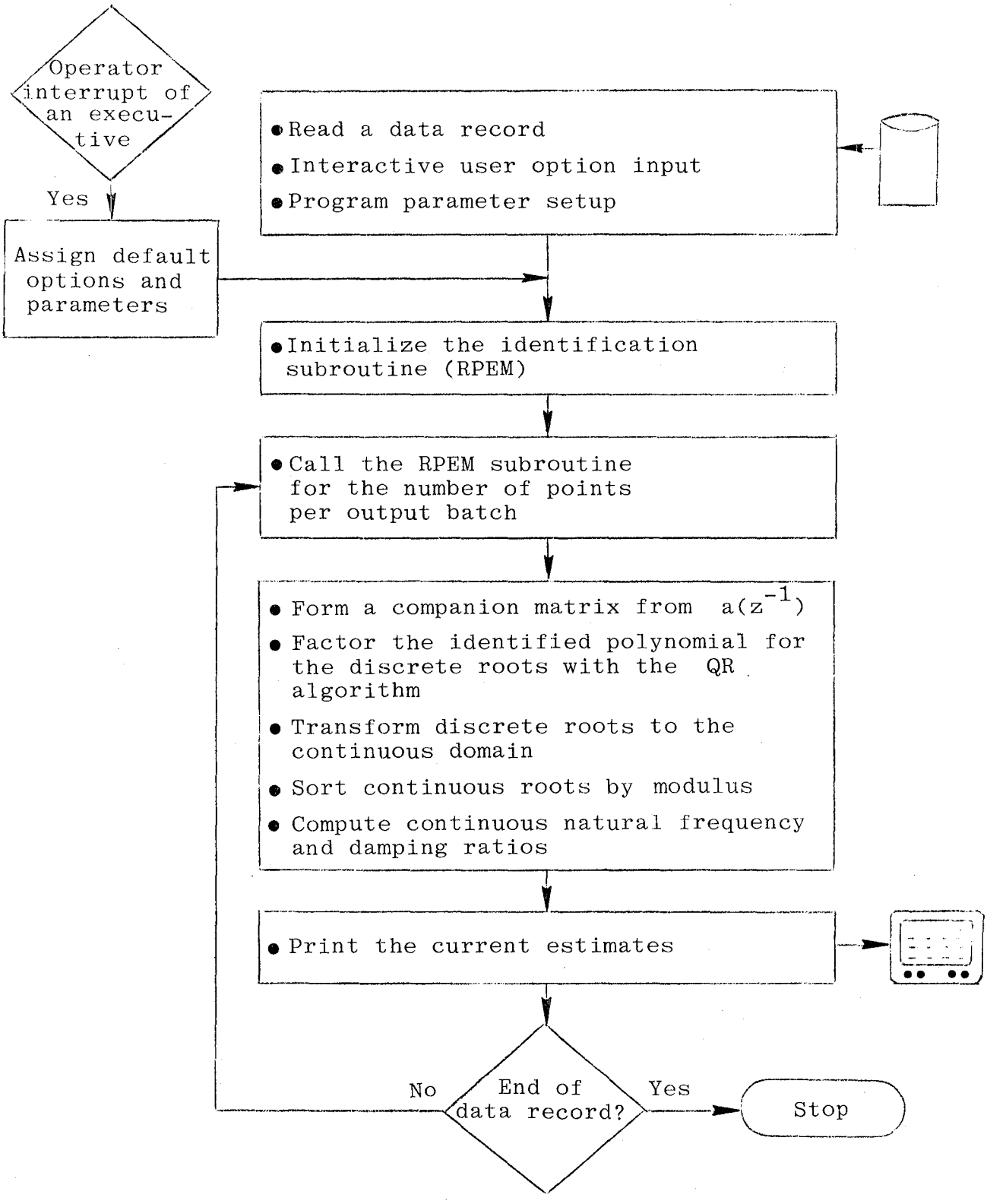


Figure 6-2. Near Real-Time Flutter Analysis Flowchart

Detailed descriptions of the options and input/output arguments are given with comment statements in the subroutines (see Appendix B); a brief description of the routines in Fig. 6-1 is:

NRFTA	The main program which allows the user to set up the identification problem and calls the principal computational and input/output subroutines.
DATAIN	Opens an input and output data file and calls SAVLOD to read the data records.
RPEM	Performs an update of the ARMAX model parameters and their covariance for one additional input and/or output data point.
NSTABL	Performs a stability check on the $C(z^{-1})$ polynomial using the Schur-Cohn test.
RG	Finds the eigenvalues of a real general matrix, used to factor the $A(z^{-1})$ polynomial by operating on the associated companion matrix.
HQR3, ORHES, ORTRAN	Supporting eigen-decomposition routines which transform a matrix to its Hessenberg form, and subsequently to its real triangular form (Schur form) with the unsymmetric QR algorithm.

There are a number of small supporting subroutines which have been documented elsewhere [15].

There are several default values used to initialize RPEM that have been fixed in NRTFA; their values or the way theyare initialized could easily be changed in the source code. RPEM uses an exponential forgetting factor, discounting past data so as to minimize at time t the sum

$$\sum_{j=1}^{t} \lambda^{t-j} \varepsilon^2(j,\theta),$$

where λ itself is a function of time,

$$\lambda(j) = (1-a_\lambda^j)\lambda_f + a_\lambda^j\lambda_o, \quad \text{computed recursively as}$$

$$\lambda(j+1) = (1-a_\lambda)\lambda(j) + a_\lambda\lambda_f \ .$$

The fixed parameters are the initial forgetting factor $\lambda_o(.9)$, the final forgetting factor $\lambda_f(1)$, and the rate at which λ approaches its final value, $a_\lambda(.97)$. The values in parenthesis are those fixed for flutter application in the early part of NRTFA (see the source listings in Appendix B). The above forgetting factor parameters achieve a value of .99 (from .9) in approximately 75 data points, and .999 in 150 data points.

In computing the regressors by filtering with the $C(z^{-1})$ polynomial (see Section 3.2.2), a contraction factor is applied to the poles of $\hat{C}(z^{-1})$, k_{fac}, which reduces the effect of the filtering. k_{fac} is also varied with time by the equation

$$k_{fac}(j+1) = a_{k_{fac}} k_{fac}(j) + (1 - a_{k_{fac}}) \ ,$$

where k_{fac} is approaching the value of 1.

The initial value is set at $k_{fac} = .01$ with $a_{k_{fac}} = .999$. Thus the algorithm starts out as nearly approximately maximum likelihood (AML) [7], where the residuals are used directly in the regressors, giving good transient convergence characteristics and then slowly approaching the RML2 [2] or standard RPEM algorithm (k_{fac} reaches .9 after approximately 2300 data points), for good asymptotic convergence characteristics.

6.3 MEMORY RESOURCES AND TIMING COMPARISONS

The active memory requirements for tne NRTFA program and the system supporting subroutines on the VAX 11/780 (VMS Version 3.1) are shown below in Table 6-1, for the problem definition given in Table 6-2.

Routine Type	Memory Requirement (Bytes)
Main Program	51,576
Input/Output Routines	2,007
Identification Routines	2,138
Eigensystem Routines	24,138
System Support Routines	2,061
Total Executable Code	81,920

Table 6-1. NRTFA Memory Requirements.

The above memory requirements include storage for the maximum sizes shown in Table 6-2 below.

Parameter	Maximum Size
dim. (θ)	60
Number of modes with an A, B, and C polynomial	10
Number of modes with A and B or A and C polynomial	15
Number of points per data record	2000

Table 6-2. NRTFA Program Parameter Limits
(Effected by Current Array Dimension Bounds)

To characterize the CPU requirements of different functions within the NRTFA program, on the VAX 11/780, the following experiment was performed. RPEM identifications from 200 data points of the A, B, and C polynomials, as well as only the A and B polynomial for 2, 4, and 8 modes were performed. The interval for outputting the frequency and damping estimates was varied to

separate the CPU required for identification from that for factoring the polynomial and outputting the desired estimates. In all cases the order of the A, B, and C polynomials is twice the number of oscillatory modes. The results are tabulated in Table 6-3.

Number of Modes	Number of Points/Batch	Polynomials Identified	CPU for RPEM (sec*)	CPU for Polynomial Factorization and ω_n, ξ output (sec**)
2	20	A,B,C	1.74/200	.27/8
2	5	A,B,C	1.74/200	1.107/32
4	20	A,B,C	4/200	2/8
4	5	A,B,C	4/200	8/32
8	20	A,B,C	14.58/200	11.93/8
8	5	A,B,C	14.58/200	47.72/32
8	20	A,B	7.27/200	11.93/8
8	5	A,B	7.27/200	47.72/32

Table 6-3. NRTFA Timing Study Results

Note that the common factor of the time to read the input data has been subtracted from all cases. These results can be depicted much more clearly, graphically. Figure 6-3 shows the CPU time in seconds required for a single data point parameter update (6n

---

* $1.74/200 \triangleq 1.74$ seconds for 200 updates by the RPEM subroutine.

** .27/8 = .27 seconds for 8 output batch computations. (Factorization of the polymonial, transformation to continuous frequencies and damping ratios, as well as output to a terminal in the format shown in Appendix A.)

parameters of the full ARMAX model for n modes) as a function of the number of modes. Similarly the transformation of the ARMAX model parameters to continuous frequencies and damping ratios is shown in the plot below as a function of the number of modes. Clearly overhead becomes more important with a small number of modes.

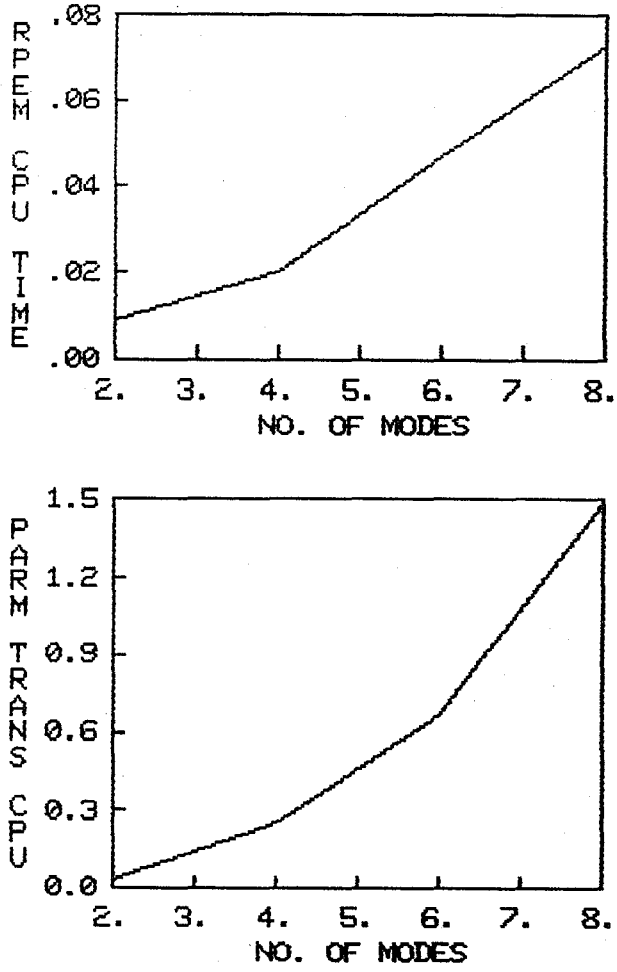


Figure 6-3. CPU for Identification and Discrete Polynomial Factorization and Output
(Full ARMAX Model - 6n Parameters for n Modes)

These values are consistent with operation counts for these two algorithms and the machine cycle time for the VAX 11/780 (VMS with a Floating Point Accelerator). The floating point operation cycle times for typical machines are shown below in Table 6-4.

Table 6-4. Millions of Operations Per Second

Machine	Vendor Rating[1] (MOPS)	Performance Benchmark[2]
IBM 3033	4.00	1.77(D)[3]
Cyber 175	5.06	1.36(S)
CDC 7600	10.	1.24(S)
CDC 6600	2.5	.477(S)
VAX 11/780 VMS/FPA	.831	.190(S)
HP 5451C (HP1000 - E Series)	.08	-
Apollo	-	.0196(S)

[1] "Mainframe KOPS Rating," Datamation 1982

[2] Floating Point (MOPS) for a typical mathematical software problem [16]

[3] (S) or (D) refers to operations in single or double precision

For the 500 sps of the DAST flight data, with, for example, a four mode model, and frequency and damping estimates output four times a second to the operator, the CPU effort is approximately eleven times real time (10 seconds for identification and 1.2 for factoring the polynomials in one second of flight data). These timing results do not include accuracy estimates, which require the mode shapes, but beyond this should not require significant additional work. The above CPU requirements are

not as pessimistic as they may first appear for implementing RPEM in real time to do flutter analysis.

Thirty percent of the work in factoring the polynomials can be eliminated if mode shapes are not desired; however, the identification algorithm is where the reduction is needed for a real-time implementation. For the modes of interest the sampling time could be cut at least in half and with some loss in accuracy the sampling rate could be cut by a factor of five.

The RPEM version implemented here has been a research tool that can be streamlined significantly. Even single subscripting the arrays could improve efficiency by as much as thirty percent. In addition, the implementation of a block recursive algorithm could significantly increase the throughput for a given machine speed. Such considerations are part of the general analysis necessary for actual real-time implementation.

Other considerations for actual real-time implementation include adding a higher level routine to function as an executive so that the operator can restart the procedure with function key at his console, interfacing a data buffering routine with the telemetry system, and displaying the modal parameters and accuracy estimates graphically at the operator's console.

6.4 INSTALLATION INSTRUCTIONS

The NRTFA program is written in ANSI Standard FORTRAN-77, such that conversion to operating systems meeting this standard should be straightforward. The storage requirements are modest, such that an overlay structure is not required on machines such as the Cyber, not having virtual memory. The principal adaptations all have to do with input/output. The subroutine FILES which dynamically opens files for the control and output data requires a suitably modified OPEN command. The subroutine SAVLOD which reads the data files with a specified format into

the buffer arrays should be changed to a convenient format for the particular installation. The subroutines FILES (not shown as it is machine dependent) and SAVLOD are called by the routine DATAIN. This file interface is exactly that of $MATRIX_X$ [1].

The variable EPS in the real general eigensolver (RG) should be set to the host machine precision ($2^{(1-t)}$ where t is the number of binary bits in the mantissa of a floating point number).

This Page Intentionally Left Blank

SECTION 7

SUMMARY

Recent theoretical analysis of recursive identification algorithms has been applied to flutter test monitoring, and implemented in a near-real-time flutter analysis program (NRTFA) shown to be highly reliable. This report documents the algorithm and model forms used, with the numerical analysis and estimation issues considered for its choice. The performance of the recursive predictive error method (RPEM) on simulated data compares well with predicted convergence rates, accuracy and robustness. The results on the DAST flight data show modal estimates that are consistent with NASTRAN and ground test vibration results, with consideration of the flutter suppression system (FSS) (see Gilyard and Edwards [14]).

Timing studies of the configured FORTRAN program show that real-time operation will be possible with careful implementation. The development and documentation of this effort has clarified the issues to be resolved for a full real-time flutter monitoring system. The NRTFA program has been documented sufficiently for program modifications, for adaptations to other computers (Section 6), and for use in current flutter analysis (Appendix A). Subsequent subsections explain these conclusions more fully.

7.1 ALGORITHM CHARACTERISTICS

The RPEM algorithm implemented in the NRTFA program identifies ARMA-type models (see Section 2.2 for the ARMAX model definition), which can be specified for three cases, known inputs,

unknown inputs or both. For the first two cases, and sufficiently high signal-to-noise ratios in the third case, this algorithm is globally convergent. The $A(z^{-1})$ polynomial or equivalently the system dynamics is always identified. With known inputs the $B(z^{-1})$ or equivalently the input distribution matrix is identified; with unknown inputs the $C(z^{-1})$ or equivalently the constant Kalman gain is identified.

Two features have been added to effect good transient convergence behavior. Initially the data is weighted with an exponential forgetting factor which discounts past data. This factor is changed to approach one, hence tapering off this windowing effect after the parameter estimates have stabilized somewhat. The NRTFA program also uses a second initial transient factor which controls the computation of the estimated gradient. This factor constrains the algorithm to be nearly linear regression initially [3], [7] which gives a rapid initial convergence rate, and then slowly tranforms to a full Gauss-Newton minimization of the prediction error [3] for assured asymptotic convergence to the global minimum (for the three cases described above).

7.2 ALGORITHM PERFORMANCE

Colored sensor noise and higher order dynamics are automatically accounted for by an overparameterized ARMA-type model. The significant conclusion about noise effects is that whenever some excitation can be tolerated it is highly desirable for improving estimation accuracy. Unknown inputs still yield reasonable damping estimates for signal-to-noise ratios above approximately ten. Families of curves show the effect of different turbulence, sensor noise and input levels in Figures 3-2 and 3-3. The convergence rates and accuracy of estimated parameters from six different simulations compare favorably to predicted error bounds in Tables 4-3 through 4-5 demonstrating the underlying reliability. RPEM is based on a stability analysis of a

differential equation associated with the behavior of the parameter estimates as a function of time [6]. Recursive algorithms can be made robust to data dropouts and outliers with fairly simple logic (see Section 3.3.1), resulting in improved estimation accuracy (see Table 3-1 for a quantitative comparison).

For known inputs and a low level of turbulence the recursive least squares (RLS - $A(z^{-1})$ and $B(z^{-1})$ only) provides very good results as long as sufficient modes are modeled to cover unknown effects in the vehicle response; for example, with a high-amplitude swept sine input, a three mode (RLS) model did not give good estimates, whereas five and eight mode models did. The unknown input results (see Section 5.2) support the general conclusion stated previously that a sufficient turbulence level must be present to give a signal-to-noise ratio (SNR) of approximately 10 for estimation particularity of the damping parameter. With a low SNR, even low amplitude known inputs increase the estimation of the $A(z^{-1})$ polynomial and hence frequencies and damping ratio. Monitoring of the final portion of the DAST Flight 3 has been emulated with the delivered algorithm, showing clearly where the flutter mode damping starts its trend to ξ=-.022 before saturation of the sensors and controls and eventual failure of the right wing.

7.3 SOFTWARE IMPLEMENTATION

Fixed parameters that influence the convergence characteristics of RPEM are described in Section 6 such that the NASA Dryden personnel can modify the source program (see Appendix B for the listings of principal routines) if so desired. Minimal necessary installation instructions are also given. Appendix A illustrates an example execution of the NRTFA program, with guidelines for input options and description of the modal estimate outputs.

7.4 RECOMMENDATION FOR FURTHER RESEARCH

Work is needed in two directions jointly to achieve semi-automated monitoring of aircraft flutter in real time.

1. Further development and testing of algorithms on several flight records and simulation runs with a variety of errors. Also paramenters like batch size, fading factors and model orders need to be further investigated.
2. Development of a real-time flutter analysis hardware system.

The hardware system could be used on the ground initially but could be extended to be an onboard system as more experience is gained with the algorithms and the computation capability advances.

The goal of such a real-time flutter analysis system would be to provide a dual environment. The real-time environment monitors flutter characteristics for safe flight tests. In addition, portions of the test are isolated for detailed post-flight analysis. The second environment is the post-flight processing capability. This would be accomplished with a flexible identification program to extract more accurate estimates of the frequency and damping estimates on critical maneuvers where the real-time algorithm indicated very light damping or unusual behavior. Improvements to the real-time algorithms could also be made in this "development" or post-flight processing environment, as more experience is gained.

Specific algorithm issues that deserve more attention are a better quantification of estimation error effects due to model order, more experience with convergence parameter setup values, particularly on flight data, the effect and design of prefilters, input design for maximum accuracy, and parameter estimation

accuracy during the convergence transient phase.

The timing results of Section 6 indicate that speed needs to be a significant concern in the further algorithm analysis, quantifying the effects of, for example, not updating the covariance as often, etc. A comparison of the RPEM, extended Kalman Filter (EKF), and maximum likelihood output error formulation for a block recursive or semi-batch mode should identify the best algorithm for the accuracy, convergence, and speed characteristics most suitable for real-time flutter monitoring.

The joint development of the hardware system with further algorithm investigation will be necessary, even though the problem appears to be compute bound, because the actual throughput of real-time systems can not be fully predicted by study alone. Finally, the technology is now available to integrate a fully automatic system and validate its performance.

This Page Intentionally Left Blank

REFERENCES

1. R. A. Walker, S. C. Shah, and C. Z. Gregory, Jr., "MATRIX$_x$: A Data Analysis, Identification, Control Design and Simulation Program," Control Systems Magazine, Vol. 2, No. 4, December 1982.1

2. T. Söderström, L. Ljung, and I. Gustavsson, "A Theoretical Analysis of Recursive Identification Methods," Automatica, 14, 1978, pp. 231-244.

3. L. Ljung and T. Söderström, "Theory and Practice of Recursive Identification," MIT Press, Cambridge, Massachusetts, 1983.

4. K. J. Astrom and T. Söderström, "Uniqueness of the M:aximum Likelihood Estimates of the Parameters of an ARMA Model," IEEE Trans. on Auto. Contr., AC-19, 1974, pp. 769-773 (Cited in [2]).

5. T. Söderström, "Convergence Properties of the Generalized Least Squarest Identification Method," Automatica, 10, 1974, pp. 617-626, (Cited in [2]).

6. L. Ljung, "Convergence of Recursive Estimators," 5th IFAC Symposium on Identification and System Parameter Estimation," Pergamon Press, Darmstadt, F.R.Germany, 1979.

7. V. Solo, "Time Series Recursions and Stochastic Approximation," Ph.D Dissertation, Australian National Univ., September, 1978.

8. L. Ljung, "On Positive Real Transfer Function and Convergence of Some Recursions," IEEE Trans. Auto. Contr., AC-22, August 1977, pp. 539-550.

9. K. W. Iliff, R. E. Maine, and T. D. Montgomery, "Important Factors in the Maximum Likelihood Analysis of Flight Test Maneuvers," NASA TPf-1459, 1979.

10. G. J. Bierman, Factorization Methods for Discrete Sequential Estimation, Academic Press, New York, 1977.

11. L. Ljung and P. E. Caines, "Asymptotic Normality of Prediction Error Estimators for Approximate System Models," Stocastics, 3, 1979, pp. 29-46; or, 17th IEEE Conference on Decision and Control, 1978.

12. P. J. Huber, Robust Statistical Procedures, SIAM Publication No. 27, Philadelphia, Pennsylovania, 1977.

13. R. A. Walker, A. Emami-Naeini, and P. Van Dooren, "A General Algorithm for Solving the Algebraic Ricatti Equation," Proc. of the 21st IEEE Conf. on Decision and Control, Orlando, Florida, December 1982.

14. G. B. Gilyard and J. W. Edwards, "Real-Time Flutter Analysis of an Active Flutter-Suppression System on a Remotely Piloted Research Aircraft," Agard Flight Mechanics Panel Symposium, Cesme, Turkey, October 1982.

15. R. A. Walker, "Computing the Jordan Form for Control of Dynamic Systems," SUDAAR 528, Ph.D. Dissertation, Dept. of Aeronautics and Astronautics, Stanford Univ., 1981.

16. J. J. Dongarra, "Redesigning Linear Algebra Algorithms," Proc. of the International Colloquim on Vector/Parallel Computing in Scientific Applications Conference, Paris, France, March 17-18, 1983.

APPENDIX A

USER'S GUIDE

NEAR REAL-TIME FLUTTER ANALYSIS PROGRAM (NRTFA)

The NRTFA program setup parameters are entered interactively as shown below.

```
   ENTER INPUT FILE NAME...
U

   ENTER OUTPUT FILE NAME...
Y

   DO YOU WANT C(Z) POLYNOMIAL, IE. KALMAN GAIN IDENTIFICATION
   (Y OR NO)?
Y

   DO YOU WANT INPUT POLYNOMIAL B(Z) TO BE IDENTIFIED (Y OR N)?
Y

   ENTER NUMBER OF DATA POINTS TO BE ANALIZED...
200

   NUMBER OF MODES TO BE CONSIDERED?
4

   HOW MANY POINTS PER BATCH?
20
```

The first two prompts are for the excitation and sensor data files, where the file names can be up to 32 characters. The input file U above contains the sequence of control positions, for open loop identification, or the excitation superimposed on a closed loop control system for closed loop identification. The

excitation channel is always read, whether or not the $B(z^{-1})$ polynomial is identified. The output file contains the sequence of accelerometer outputs (the present version of RPEM is a SISO algorithm).

The second group of two prompts above defines which two or all three of the ARMAX polynomial models are identified (see Section 2.2). If there is knowledge of the control channel $B(z^{-1})$ should be identified, and if the gust inputs are known to be small with wideband accelerometer noise then the recursive least-squares (RLS) algorithm ($A(z^{-1})$ and $B(z^{-1})$ identified, without $C(z^{-1})$) can be used. In all other cases the full ARMAX model ($A(z^{-1})$, $B(z^{-1})$ and $C(z^{-1})$) should be used.

The last three prompts of the interactive setup session specify the number of data points to be read into buffer arrays, the number of oscillatory modes (resulting in 2n or 3n parameters) modeled, and how often the frequency and damping estimates are output to the terminal.

The modal estimate output to the terminal resulting from the session above follows:

```
          FIRST MODE              SECOND MODE             THIRD MODE
          FOURTH MODE             FIFTH MODE              SIXTH MODE

  :C1:    5.433   0.153   :C2:    9.672   0.167   :C3:   13.749   0.079
  :C4:   14.665   0.356   :

  :C1:    6.538   0.077   :C2:    9.938   0.095   :

  :C1:    0.493  -0.213   :C2:    7.005   0.008   :C3:   10.038   0.086
  :C4:   13.708   0.016   :

  :C1:    0.535   0.234   :C2:    7.151   0.048   :C3:   10.775   0.124
  :C4:   13.741   0.071   :

  :C1:    0.669   0.057   :C2:    7.141   0.095   :C3:    9.071   0.461
  :C4:   12.724   0.118   :

  :C1:    0.617   0.234   :C2:    6.877   0.132   :C3:    9.878   0.283
  :C4:   13.765   0.092   :C5:   14.519   0.330   :

  :C1:    0.686   0.088   :C2:    6.681   0.163   :C3:    8.927   0.212
  :C4:   13.505   0.132   :

  :C1:    0.675   0.052   :C2:    6.570   0.135   :C3:    9.557   0.177
  :C4:   13.763   0.091   :C5:   14.043   0.218   :

  :C1:    0.659   0.013   :C2:    6.483   0.161   :C3:    9.111   0.119
  :C4:   13.414   0.126   :
FORTRAN STOP
#
```

Estimates for the first twenty points are computed before output begins, hence these 9 modal output parameter groups. A pair of roots are prefixed by whether they are complex (C) or two real (R) roots, along with the mode numbers. The modes are ordered from smallest to largest by natural frequency. For a complex pair the first number is the natural frequency and the second the damping ratio.

For real roots, both are printed. If the discrete roots cannot be realized as real continuous modes they are not printed at all, for example the second group above, where only two of four modes were printed.

This Page Intentionally Left Blank

APPENDIX B

FORTRAN LISTINGS

Listings of the major subroutines of the Near Real Time Flutter Analysis (NRFTA) Program as described in Section 6 follow, ordered according to the calling sequence shown in Figure 6-2.

```
      PROGRAM RTFA
      DOUBLE PRECISION PMAT(60,60),THETA(60),PSID(60),FI(60),
     1 PSI(60),TSTAB(21),WORK(42),FRPEM(60),GRPEM(60),LRPEM(60),
     2 UARRA(10),YARRA(10),UCUR,YCUR,LAMBDA,KFACT,LAMDL,ALMDA,
     3 AKFACT,V,RESID,PINIT,CFACT,PRERR
C
C     ------------------------------------------------------------
C     THIS IS THE MAIN PROGRAM FOR REAL-TIME FLUTTER ANALYSIS BY
C     RECURSIVE PREDICTION METHOD (RPEM).
C     HAVING INPUT U(T) AND OUTPUT Y(T) PARAMETER ESTIMATE THETA
C     IS UPDATED EACH TIME SUBROUTINE RPEM IS CALLED.
C     MODEL STRUCTURE USED IS
C     A(Q**-1 )Y(T)=B(Q**-1 )U(T)+C(Q**-1 )E(T)
C     THETA- VECTOR OF ORDER (NARPM+NBRPM+NC) CONTAINING THE
C     PARAMETER ESTIMATES.
C     THETA=(A(1),A(2),...,A(NARPM),B(1),B(2),...,B(NBRPM),C(1),
C     C(2),...,C(NC))
C     THIS IS AN INTERACTIVE SEMI-BATCH PROGRAM WITH OPTIONS ON
C     TOTAL NUMBER OF DATA POINTS, NUMBER OF POINTS PER BATCH
C     AND IDENTIFYING [A(Z),B(Z),C(Z)], [A(Z),B(Z)] OR [A(Z),
C     C(Z)]. THE FOLLOWING INPUTS ARE NEEDED.
C     NDATP - NUMBER OF DATA POINTS.
C     NMODE - NUMBER OF MODES TO BE IDENTIFIED.
C     NPRBCH - NUMBER OF POINTS PER BATCH.
C
C     AFTER NSKIP POINTS, FOR EVERY NPRBCH POINTS NATURAL FREQUENCY
C     AND DAMPING RATIO  OF EACH MODE ARE CALCULATED AND PRINTED.
C     THE TRANSFORMED DYNAMIC MATRIX
C     F( , )=[-A( ) IN THE FIRST COLUMN , IDENTITY MATRIX]
C     AFTER TRANSFORMING THE EIGENVALUES OF F INTO CONTINOUS
C     DOMAIN THE NATURAL FREQUENCY AND DAMPING RATIO OF COMPLEX
C     ROOTS ARE CALCULATED. IF THE ESTIMATED ROOTS ARE COMPLEX
C     THEN IN OUTPUT :C: WILL BE PRINTED BEFORE FREQUENCY AND
C     DAMPING RATIO OTHERWISE :R: AND LOCATION OF REAL ROOTS
```

```
C     WILL BE PRINTED. TO BE ABLE TO PRINT REAL ROOTS THE
C     LINE BELOW LABEL 120 IN THE MAIN PROGRAM 'KDUM=KDUM3'
C     SHOULD BE DELETED.
C     IF DAMPING RATIO IS NEGATIVE THAT MODE IS UNSTABLE.
C
C     F( , ) - DYNAMIC MATRIX
C     FREQ( )- NATURAL FREQENCY
C     DAMP( )- DAMPING RATIO
C     RLRT( )- REAL ROOTS
C     CLXRT( )- COMPLEX ROOTS
C     DT      - SAMPLING PRIOD
C
C     INPUT PARAMETERS FOR SUBROUTINES HAVE BEEN DEFINED IN THEM.
C
C     SUBROUTINE DATAIN READS INPUT U( ) AND OUTPUT Y( ).
C     SUBROUTINE RPEM UPDATES THETA( ) PARAMETER ESTIMATE.
C     SUBROUTINE RG COMPUTES THE EIGEN-VALUES OF F( , ).
C     ------------------------------------------------------
C
      DOUBLE PRECISION F(20,20),VRG(20,20),U(500),Y(500)
      DOUBLE PRECISION FREQ(20),DAMP(20),CLXRT(20,2),RLRT(20),DT
      DOUBLE PRECISION EPSRG,ER(20),EI(20),SUPD(20),DUMP
      INTEGER INPUT(32),OUTPUT(32),ID(4),NR,NC,JOB,IMG
      INTEGER NLOW,NUP,INFO,NV,NARG,TYPE(20),NDEL(20)
      CHARACTER CHAR(10),ANSWR
      COMPLEX*16 CDUM1,CDUM2
      DATA IMG,JOB/0,500/
      DATA NSKIP/40/
C


C     ---------- EPSRG= MACHINE EPSILON ---------------------
C
      DATA CFACT/1.0D10/,IDIM/60/,EPSRG/2.0D-16/
      DATA NA/20/,NV/20/
      DATA  DT/.2292/
      OPEN(UNIT=5,NAME='INPUT',STATUS='UNKNOWN')
      OPEN(UNIT=6,NAME='OUTPUT',STATUS='UNKNOWN')
C
C    ---------- READING INPUTS ---------------------------------
C
      CALL DATAIN (JOB,IMG,NDATP,U,Y)
      WRITE(6,881)
  881 FORMAT(2X,' DO YOU WANT C(Z) POLYNOMIAL, IE. KALMAN GAIN',
     1          ' IDENTIFICATION (Y OR N)?')
      READ(5,883)ANSWR
      WRITE(6,882)
  882 FORMAT(2X,'DO YOU WANT INPUT POLYNOMIAL B(Z) TO BE ',
     1     'IDENTIFIED(Y OR N)?')
      READ(5,883) ANSRB
  883 FORMAT(A)
      WRITE(6,880)
```

```
  880 FORMAT(2X,' ENTER NUMBER OF DATA POINTS TO BE ANALIZED.')
      READ(5,*)NDATP
      WRITE(6,994)
  994 FORMAT(2X,' NUMBER OF MODES TO BE CONSIDERED? ...')
      READ(5,*)NMODE
      WRITE(6,995)
  995 FORMAT(2X,' HOW MANY POINTS PER BATCH?')
      READ(5,*)NPRBCH
      NBACH=NDATP/NPRBCH
      TWONMD=2*NMODE
C
C     ---------- SET PARAMETERS FOR RPEM SUB. ------------------
C
      NARPM=TWONMD
      NBRPM=TWONMD
      NC=TWONMD
      IF(ANSWR.NE.'Y') NC=0
      IF(ANSRB.NE.'Y') NBRPM=0
      ND=0
      MP=NARPM+NBRPM+NC
      NP1=NARPM+1
      NLOW=1
      NUP=NARPM
      INIT=1
      IER1=1
      IER2=1
      DO 10 I=1,MP
   10 THETA(I)=0.0D0
      PINIT=1.D4
      LAMBDA=.9D0
      ALMDA=1.0
      LAMDL=0.97
      KFACT=0.01
      AKFACT=0.999
      ISTAB1=1
      ISTAB2=0
C
C     ------------INITIAL RPEM CALL ----------------------------
C
      CALL RPEM(THETA,PMAT,NARPM,NBRPM,NC,ND,UCUR,YCUR,LAMBDA,
     1 KFACT,CFACT,ISTAB1,ISTAB2,V,PRERR,RESID,INIT,PINIT
     2 ,IDIM,IER1,IER2,FI,PSI,PSID,FRPEM,GRPEM,LRPEM,WORK,
     3 TSTAB,UARRA,YARRA)


      WRITE(6,993)
  993 FORMAT(11X,'FIRST MODE',11X,'SECOND MODE',11X,'THIRD MODE',
     1     /11X,'FOURTH MODE',10X,'FIFTH MODE ',11X,'SIXTH MODE')
```

```
C     ------------ MAIN LOOP AND BATCH LOOP SET-UP -------------
C
      INIT=0
      IJ=0
      DO 30 J=1,NBACH
      DO 20 I=1,NPRBCH
      IJ=IJ+1
      UCUR=U(IJ)
      YCUR=Y(IJ)
      LAMBDA=ALMDA*LAMBDA+(1.0D0-ALMDA)*LAMDL
      KFACT=AKFACT*KFACT+(1.0D0-AKFACT)
C
C     ---------- RPEM CALL ------------------------------------
C
      CALL RPEM(THETA,PMAT,NARPM,NBRPM,NC,ND,UCUR,YCUR,LAMBDA,
     1 KFACT,CFACT,ISTAB1,ISTAB2,V,PRERR,RESID,INIT,PINIT
     2 ,IDIM,IER1,IER2,FI,PSI,PSID,FRPEM,GRPEM,LRPEM,WORK,
     3 TSTAB,UARRA,YARRA)
   20 CONTINUE
C
C     ---------- TRANSFORMED DYNAMIC MATRIX SET-UP -------------
C
      IF(IJ.LT.NSKIP) GO TO 30
      DO  50 IDUM=1,NARPM
      DO  50 JDUM=1,NARPM
      F(IDUM,JDUM)=0.0D0
      IF(JDUM.EQ.IDUM+1) F(IDUM,JDUM)=1.0
   50 CONTINUE
      DO 40 K=1,NARPM
      F(K,1)=-THETA(K)
   40 CONTINUE
C
C     --------- CALL RG TO COMPUTE EIGENVALUES -----------------
C
      CALL RG(F,NA,NARPM,VRG,NV,NLOW,NUP,ER,EI,TYPE,SUPD,
     1          NDEL,NP1,NBLOCK,EPSRG,INFO)
C
C     ---------- TRANSFORMING ROOTS TO CONTINOUS DOMAIN ---------
C
      K=0
      KDUM=0
      KDUM1=0
   60 K=K+1
      IF(K.GE.NARPM) GO TO 90
      KP1=K+1
      IF(ABS(F(K,KP1)).LT.1.0D-07) GO TO 110
      KDUM=KDUM+1
      CDUM1=DCMPLX(F(K,K),F(K,KP1))
      CDUM2=CDLOG(CDUM1)/DT
      CLXRT(KDUM,1)=DREAL(CDUM2)
      CLXRT(KDUM,2)=DIMAG(CDUM2)
      K=K+1
      GO TO 60
```

```
  110 IF(ABS(F(K,K)).GE.1.0D0) GO TO 60
      IF(F(K,K).LT.0.0D0) GO TO 150
      KDUM1=KDUM1+1
      RLRT(KDUM1)=DLOG(F(K,K))/DT
      GO TO 60
  150 KDUM=KDUM+1
      CLXRT(KDUM,1)=DLOG(-F(K,K))/DT



      CLXRT(KDUM,2)=3.1415/DT
      GO TO 60
   90 CONTINUE
      DO 80 K=1,KDUM
      IF(ABS(CLXRT(K,2)).GT.1.0D-07) GO TO 70
      FREQ(K)=CLXRT(K,1)
      DAMP(K)=CLXRT(K,1)
      CHAR(K)='R'
      GO TO 80
   70 FREQ(K)=DSQRT(CLXRT(K,1)*CLXRT(K,1)+CLXRT(K,2)*CLXRT(K,2))
      DAMP(K)=-CLXRT(K,1)/FREQ(K)
      CHAR(K)='C'
   80 CONTINUE
C
C     ---------- SORTING THE ROOTS ----------------------------
C
      KDUM3=KDUM
      DO 140 K=1,KDUM-1
      DO 140 KP1=K+1,KDUM
      IF(ABS(FREQ(K)-FREQ(KP1)).LT.0.0001)KDUM3=KDUM3-1
      IF(ABS(FREQ(K)-FREQ(KP1)).LT.0.0001)FREQ(KP1)=1.0D+10
      IF(FREQ(K).LT.FREQ(KP1)) GO TO 140
      DUMP=FREQ(K)
      FREQ(K)=FREQ(KP1)
      FREQ(KP1)=DUMP
      DUMP=DAMP(K)
      DAMP(K)=DAMP(KP1)
      DAMP(KP1)=DUMP
      ANSWR=CHAR(K)
      CHAR(K)=CHAR(KP1)
      CHAR(KP1)=ANSWR
  140 CONTINUE
      KDUM=KDUM3
      IF(KDUM.GT.NMODE) GO TO 120
      DO 130 K=1,KDUM1
      KDUM=KDUM+1
      FREQ(KDUM)=RLRT(K)
      DAMP(KDUM)=RLRT(K+1)
      IF(K.GT.KDUM1)DAMP(KDUM)=0.0
  130 CHAR(KDUM)='R'
  120 IF(KDUM.GT.NMODE) KDUM=NMODE
C
```

```
C      ---------- OUTPUT PRINT-OUT ------------------------------
C
       KDUM=KDUM3
       WRITE(6,883)
       WRITE(6,992)(CHAR(K),K,FREQ(K),DAMP(K),K=1,KDUM)
  992 FORMAT(3(3X,'|',A1,I1,'|',2F8.3 ))
   30 CONTINUE
       STOP
       END
$      '


       SUBROUTINE DATAIN(JOB,IMG,NDATP,U,Y)
C
C      ---------- SUBROTINE DATAIN ------------------------------
C
       DOUBLE PRECISION Y(1),U(1)
       INTEGER ID(4),NR,NC,JOB,IMG
       INTEGER INPUT(32),OUTPUT(32)
C
C      ---------- INPUT FILE NAMES
C
       WRITE(6,9000)
 9000 FORMAT(/,' ENTER INPUT FILE NAME...')
       READ(5,9010,END=2000) INPUT
 9010 FORMAT(32A1)
       WRITE(6,9030)
 9030 FORMAT(/,' ENTER OUTPUT FILE NAME...')
       READ(5,9010,END=2000) OUTPUT
C
C      ---------- ASSIGN FILES
C
       CALL FILES(2,INPUT)
       CALL FILES(11,OUTPUT)
C
C      ---------- LOAD MATRIXX FILE
C
       CALL SAVLOD(2,ID,500,NR,NC,IMAG,JOB,U,U)
       CALL SAVLOD(11,ID,500,NR,NC,IMAG,JOB,Y,Y)
         NDATP=NR
C
2000     CONTINUE
C
C      ---------- CLOSE FILES
C
       CALL FILES(-2,INPUT)
       CALL FILES(-11,OUTPUT)
       RETURN
       END
       SUBROUTINE SAVLOD(LUNIT,ID,MA,M,N,IMG,JOB,XREAL,XIMAG)
C
```

```
C     ---------- SUBROUTINE SAVLOAD ----------------------------
C
      INTEGER LUNIT,ID(4),M,N,IMG,JOB
      DOUBLE PRECISION XREAL(1),XIMAG(1)
C
C     IMPLEMENT SAVE AND LOAD
C     LUNIT = LOGICAL UNIT NUMBER
C     ID = NAME, FORMAT 4A1
C     M, N = DIMENSIONS
C     IMG = NONZERO IF XIMAG IS NONZERO
C     JOB = 0     FOR SAVE
C         = SPACE AVAILABLE FOR LOAD
C     XREAL, XIMAG = REAL AND OPTIONAL IMAGINARY PARTS
C
C     SYSTEM DEPENDENT FORMATS
  101 FORMAT(4A1,3I4)
  102 FORMAT(4Z18)
      IF (JOB .GT. 0) GO TO 20
C
C     SAVE
C
   10 WRITE(LUNIT,101) ID,M,N,IMG
      DO 15 J = 1, N
        K = (J-1)*MA + 1


        L = K + M - 1
         WRITE(LUNIT,102) (XREAL(I),I=K,L)
         IF (IMG .NE. 0) WRITE(LUNIT,102) (XIMAG(I),I=K,L)
   15 CONTINUE
      RETURN
C
C     LOAD
C
   20 READ(LUNIT,101,END=30,ERR=29) ID,M,N,IMG
      IF (M*N .GT. JOB) GO TO 30
      DO 25 J = 1, N
         K = (J-1)*M+1
         L = J*M
         READ(LUNIT,102,END=30) (XREAL(I),I=K,L)
         IF (IMG .NE. 0) READ(LUNIT,102,END=30) (XIMAG(I),I=K,L)
   25 CONTINUE
      RETURN
   29 WRITE(6,*) 'ERROR IN READING FILE'
C
C     END OF FILE
C
   30 M = 0
      N = 0
      RETURN
      END
```

```
      SUBROUTINE RPEM(THETA,P,NA,NB,NC,ND,U,Y,LAMBDA,K,C,
     C  ISTAB1,ISTAB2,V,EPS,EPS1,INIT,PO,IDIM,IER1,IER2,
     C  FI,PSI,PSID,F,G,L,WORK,TSTAB,UARRA,YARRA)
C
C       ---------- RPEM SUBROUTINE ----------------------------
C
C       RECURSIVE PREDICTION ERROR METHOD
C       THIS IS A MODIFIED VERSION OF THE ROUTINE ORIGINALLY
C       DEVELOPED BY T.SODERSTROM. THE MODIFIED FORM HANDLES
C       VARIABLE SIZES OF POLYNOMIALS A,B,AND C.THE ROUTINE
C       PROVIDES VARIOUS OPTIONS AS DESCRIBED BELOW.
C       THE SUBROUTINE PERFORMS THE MODIFICATION OF THE
C       PARAMETER ESTIMATES THETA FOR ONE SAMPLING INTERVAL
C       A NEW CALL TO RPEM MUST BE MADE FOR EVERY NEW
C       SAMPLING INTERVAL
C       MODEL STRUCTURE USED IS
C       A(Q**-1 )Y(T)=B(Q**-1 )U(T-ND)+C(Q**-1 )E(T)
C       THETA - VECTOR OF ORDER (NA+NB+NC) CONTAINING THE
C               PARAMETER ESTIMATES.
C       THETA=(A(1), ...,A(NA),B(1),...,B(NB),C(1),....,C(NC))
C       THETA IS CHANGED IN THE SUBROUTINE
C       P -  SYMMETRIC MATRIX OF ORDER(NA+NB+NC)
C       P  IS USED IN THE U-D FORM.(THIS U IS DIFFERENT
C       FROM THE INPUT VARIABLE U(T).THE ARGUMENT OF THE
C       ROUTINE CONTAINS U(T)).
C       P=U*D*U (TRANSPOSED) WITH D DIAGONAL AND U UPPERY
C       TRIANGULAR. THE ELEMENTS OF D ARE STORED IN THE
C       DIAGONAL OF P. THE ELEMENTS OF U ARE STORED IN THE
C       UPPER TRIANGULAR PART OF P. P IS CHANGED IN THE
C       SUBROUTINE.
C       U -THE LAST INPUT VALUE
C       Y - THE LAST OUTPUT VALUE
C       LAMBDA - THE FORGETTING FACTOR (TO BE ENTERED)
C                0.LT.LAMBDA.LE.1 .
C       K - THE CONTRACTION FACTOR
C           0.LT.K.LE.1.
C           THIS FACTOR CONTRACTS THE ROOTS OF THE
C           C - POLYNOMIAL . (THIS IS TO BE ENTERED)
C           THIS FACTOR IS USED IN FILTERING OF THE DATA
C       C - PARAMETER USED FOR THE REGULARIZATION (SHOULD BE
C           CHOSEN RATHER LARGE. TO BE ENTERED)
C           THIS LIMITS THE MAXIMUM VALUE OF THE DIAGONAL
C           ELEMENTS OF THE P - MATRIX
C       ISTAB1 - FLAG(TO BE ENTERED) FOR STABILITY TEST OF C(Z)
C                IF ISTAB1=0,NO MONITORING (STABILITY TEST AND
C                STEP SIZE REDUCTION) IS PERFORMED.
C                IF ISTAB1.NE.0, MONITORING IS PERFORMED
C       ISTAB2 - INTEGER AT RETURN GIVING THE NUMBER OF STEP
C                SIZE REDUCTIONS PERFORMED
C       V - LOSS FUNCTION- SUM OF SQUARED PREDICTION ERRORS.
C           TRANSIENT PHASE IS INCLUDED
```

```
C       EPS - GIVES THE PREDICTION ERROR ON RETURN
C       EPS1 - GIVES THE RESIDUAL ON RETURN PROVIDED UPDATE USES
C                RESIDUALS BY IER1=1
C       INIT - FLAG TO BE USED FOR STARTING THE RECURSION. IF
C              INIT=0, ALL PARAMETERS ARE UPDATED. IF INIT.NE.0,
C              APPROPRIATE INITIAL VALUES ARE FIRST SET AND THE
C              PARAMETERS ARE UPDATED USING THE AVAILABLE DATA
C              U,Y
C       PO - SCALAR PARAMETER USED TO GIVE AN INITIAL VALUE(TO
C            BE ENTERED WHEN INIT.NE.0)
C            IF INIT.NE.0, P=PO*I
C       IDIM - DIMENSION PARAMETER

C       IER1 - FLAG TO BE ENTERED. IF IER1=0, PREDICTION ERRORS
C              ARE USED IN PLACE OF THE RESIDUALS. IF IER1.NE.0,
C               THE ALGORITHM USES THE RESIDUALS.
C       IER2 - FLAG TO BE ENTERED. IF IER2=0, FILTERING IS
C              NOT USED IN THE ALGORITHM. IF IER2.NE.0,
C              FILTERING WITH POLYNOMIAL C(Z) IS PERFORMED.
C       ABS(ND) .LT. 10
C       IF IER1=0 AND IER2=0, THE ALGORITHM REDUCES TO THE
C       EXTENDED LEAST SQUARES CASE.
C
C       IF IER1=0, AND NB=NC=0, THE SUBROUTINE REDUCES TO A
C       SIMPLE RECURSIVE LEAST SQUARES ESTIMATE OF AR MODEL.
C
C       IF IER2=0, THE ALGORITHM PERFORMS AS RML1
        DOUBLE PRECISION THETA(1),P(IDIM,1),PSID(1),
     C  FI(1),PSI(1),TSTAB(1),WORK(1),F(1),G(1),U1,Y1,E1,AL
     C  ,CI,EPS,EPS1,ALFA,PO,V,C,U,Y,AMY,DD,S,BETA,GAMMA
          DOUBLE PRECISION LAMBDA,K,L(1),UARRA(1),YARRA(1)
            NN=NA+NB+NC
C        TEST FOR INITIALIZATION
         IF(INIT.EQ.0) GO TO 100
C
         V=0.D0
         DO 10 I=1,NN
         DO 10 J=1,NN
10       P(I,J)=0.D0
         DO 15 I=1,10
         YARRA(I)=0.D0
15       UARRA(I)=0.D0
         DO 20 I=1,NN
         P(I,I)=PO
         L(I)=0.D0
         FI(I)=0.D0
20       PSI(I)=0.D0
        NS=3*NC
        DO 30 I=1,NS
30      PSID(I)=0.D0
         RETURN
C
```

```
100   NDS=ND
      IF (NDS .LT. 0) GO TO 103
      IF (ND .LT. 1) GO TO 102
       DO 101 I=1,ND
        J=NDS+2-I
101    UARRA(J)=UARRA(J-1)
102    UARRA(1)=U
       U=UARRA(ND+1)
       GO TO 108
103    NDS=-NDS
       DO 105 I=1,NDS
        J=NDS+2-I
105    YARRA(J)=YARRA(J-1)
       YARRA(1)=Y
       Y=YARRA(NDS+1)
108    CONTINUE
COMPUTE PREDICTION ERROR
         EPS=Y
         DO 110 I=1,NN
110      EPS=EPS-FI(I)*THETA(I)
C        COMPUTE NEW PARAMETER ESTIMATES
         AMY=1.
C        TEST FOR NEED OF MONITORING
        IF(NC.EQ.0)GO TO 200
         IF(ISTAB1.EQ.0)GO TO 200


         ISTAB2=0
120     DO 130 I=1,NC
        NI=NA+NB+I
130      TSTAB(I+1)=THETA(NI)+L(NI)*EPS*AMY
         TSTAB(1)=1.
C        TEST FOR STABILITY OF C(Z)
        CALL NSTABL(TSTAB,NC,WORK,IST)
         IF(IST.EQ.0)GO TO 200
         AMY=AMY/2.
         ISTAB2=ISTAB2+1
         IF(ISTAB2 .GT. 25) RETURN
         GO TO 120
C        UPDATE PARAMETER ESTIMATES
200      DO 210 I=1,NN
210      THETA(I)=THETA(I)+L(I)*EPS*AMY
        IF(IER1.EQ.0)GO TO 250
C        COMPUTE RESIDUALS
         EPS1=Y
         DO 220 I=1,NN
220      EPS1=EPS1-FI(I)*THETA(I)
250     IF(IER1.EQ.0)EPS1=EPS
C
```

```
C       COMPUTE FILTERED SIGNALS AND STORE IN PSID ARRAY
         Y1=Y
         U1=U
         E1=EPS1
        IF(IER2.EQ.0)GO TO 670
        IF(NC.EQ.0)GO TO 670
        DO 620 I=1,NC
        CI=THETA(NA+NB+I)*K**I
        Y1=Y1+CI*PSID(I)
        U1=U1-CI*PSID(NC+I)
620     E1=E1-CI*PSID(2*NC+I)
        IF(NC.EQ.1)GO TO 650
C       UPDATE PSID VECTOR
        DO 630 I=2,NC
        I1=NC+2-I
        PSID(I1)=PSID(I1-1)
        I2=NC+NC+2-I
        PSID(I2)=PSID(I2-1)
        I3=NC+NC+NC+2-I
630     PSID(I3)=PSID(I3-1)
650     PSID(1)=-Y1
        PSID(NC+1)=U1
        PSID(NC+NC+1)=E1
C       UPDATE VECTORS FI AND PSI
670     CONTINUE
        IF(NA.EQ.1)GO TO 720
C
C
        DO 700 J=2,NA
        I=NA+2-J
         FI(I)=FI(I-1)
700     PSI(I)=PSI(I-1)
720     FI(1)=-Y
        PSI(1)=-Y1
        IF(NB.EQ.0)GO TO 750
        IF(NB.EQ.1)GO TO 740
        DO 730 J=2,NB
        I=NA+NB+2-J
        FI(I)=FI(I-1)
730     PSI(I)=PSI(I-1)
740     FI(NA+1)=U
        PSI(NA+1)=U1


750     IF(NC.EQ.0)GO TO 780
        IF(NC.EQ.1)GO TO 770
        DO 760 J=2,NC
        I=NA+NB+NC+2-J
        FI(I)=FI(I-1)
760     PSI(I)=PSI(I-1)
770     FI(NA+NB+1)=EPS1
        PSI(NA+NB+1)=E1
```

```
780     CONTINUE
C        COMPUTE GAIN VECTOR L, UPDATE P AND V
         DO 810 I=2,NN
         J=NN+2-I
         ALFA=PSI(J)
         J1=J-1
         DO 800 KK=1,J1
800      ALFA=ALFA+P(KK,J)*PSI(KK)
         F(J)=ALFA
810      G(J)=P(J,J)*ALFA
         G(1)=P(1,1)*PSI(1)
         F(1)=PSI(1)
C
         ALFA=LAMBDA+F(1)*G(1)
         GAMMA=0.
         IF(ALFA.GT.0.)GAMMA=1./ALFA
         IF(G(1).NE.0.)P(1,1)=GAMMA*P(1,1)
         DO 830 J=2,NN
         BETA=ALFA
         DD=G(J)
         ALFA=ALFA+DD*F(J)
         IF(ALFA.EQ.0.) GO TO 835
         AL=-F(J)*GAMMA
         J1=J-1
         DO 82<`$=1,J1
         S=P(I,J)
         P(I,J)=S+AL*G(I)
820      G(I)=G(I)+DD*S
         GAMMA=1./ALFA
         P(J,J)=BETA*GAMMA*P(J,J)/LAMBDA
         P(J,J)=MIN(P(J,J),C)
830      CONTINUE
C
         V=V+EPS**2/ALFA
835      CONTINUE
         DO 840 I=1,NN
840      L(I)=G(I)/ALFA
         RETURN
         END
$
```

```
      SUBROUTINE NSTABL(A,N,W,IST)
C     TEST FOR STABILITY
      DOUBLE PRECISION A(1),W(1),AL
C
      IST=1
      N1=N+1
      DO 1 I=1,N1
      W(I)=A(I)
1     W(N1+I)=0.
      K=0
10    IF(K.EQ.N)GO TO 99
      NK1=N-K+1
      DO 11 J=1,NK1
11    W(N1+J)=W(NK1-J+1)
      IF(W(N1+NK1).EQ.0.)GO TO 98
      AL=W(NK1)/W(N1+NK1)
      IF(ABS(AL).GE.1.0)GO TO 98
      NK=N-K
      DO 12 J=1,NK
12    W(J)=W(J)-AL*W(N1+J)
      K=K+1
      GO TO 10
98    RETURN
99    IST=0
      RETURN
      END
$
```

```
      SUBROUTINE RG(A,NA,N,V,NV,NLOW,NUP,ER,EI,TYPE,SUPD,
C
C     ---------- RG SUBROUTINE ---------------------------------
C
     1 NDEL,NP1,NBLOCK,EPS,INFO)
      INTEGER NA,N,NV,NLOW,NUP,NBLOCK,TYPE(N),NDEL(NP1)
      DOUBLE PRECISION EPS,A(NA,N),V(NV,N),ER(N),EI(N),SUPD(N)
C
C      RG FINDS THE EIGEN VALUES OF A MATRIX. ALSO IT HAS CAPABILITY
C         OF FINDING EIGEN VECTORS TOO. TO BE ABLE TO FIND EIGEN-
C        VECTORS SUBROUTINE ELMBTR WHICH HAS BEEN COMMENTED OUT MUST
C        BE CALLED.
C
C     ON ENTRY
C
C        A        CONTAINS THE INPUT MATRIX
C
C        NA,N     LEADING DIMENSION AND COLUMN DIMENSION OF A
C
C        NLOW,    LOW AND HIGH INDEX OF DIAGONAL POSITIONS OF THE
C        NUP      PORTION OF A TO BE REDUCED TO SCHUR FORM
C
C     ON RETURN
C
C        A        DESTROYED ON OUTPUT
C
C        V        CONTAINS THE PRINCIPAL VECTORS OF A
C
C        NV       LEADING DIMENSION OF V
C
C        ER,EI    REAL AND IMAGINARY PARTS OF THE N EIGENVALUES OF A
C
C        TYPE     (INTEGER) FLAG ARRAY INDICATING THE TYPE OF EIGENVALUE
C                 (0 => REAL, 1 => POS. CONJUG, 2 => NEG CONJUG)
C
C        SUPD     SUPERDIAGONAL COUPLING ELEMENTS (INDEXED ON THE COLUMN
C                 POSITION)
C
C        NDEL     FLAG ARRAY RECORDING THE LOCATION AND SIZE OF THE
C                 BLK TRIANGLES, (NDEL(I+1) IS THE BOTTOM CORNER OF BLK I
C
C        NBLOCK   NUMBER OF JORDAN MATRICES FOUND
C
```

```
C         INFO    = 0 NORMAL RETURN
C
C                 = 1 LACK OF CONVERGENCE IN INITIAL QR DECOMPOSITION
C                     TYPE(I) = -1 ARE THE UNCONVERGED EIGENVALUES
C
C     ---------- EPS IS THE RELATIVE MACHINE PRECISION, I.E. 2**(1-T)
C                WHERE T IS NUMBER OF BINARY DIGITS OF FLOATING PT
C                MANTISSA (FOR IBM 16**(1-14) = 2.22D-16)
C
      IORDER=0
      FNORM=0.0D0
      DO 10 I=1,NA
      DO 10 J=1,N
      FNORM=FNORM+A(I,J)*A(I,J)


   10 CONTINUE
      CALL ORTHES(NA,N,NLOW,NUP,A,SUPD)
      CALL ORTRAN(NA,NV,N,NLOW,NUP,A,SUPD,V)
      CALL HQR3(A,NA,N,NLOW,NUP,IORDER,V,NV,ER,EI,TYPE,SUPD,EPS)
C
C     ... SET UP FLAGS FOR ELMBTR
C
      NDEL(1)=0
      NBLOCK=1
      DO 100 I=1,N
      SUPD(I)=0.0D0
      IF(TYPE(I).LT.0) INFO=I
      IF(TYPE(I).EQ.2) GO TO 100
      NBLOCK=NBLOCK+1
      NDEL(NBLOCK)=NDEL(NBLOCK-1)+TYPE(I)+1
  100 CONTINUE
      NBLOCK=NBLOCK-1
C
C     ... ELIMINATE TO BLOCK TRIANGLAR FORM
C     ..........TO SAVE SOME CPU TIME..............
C
C    CALL ELMBTR(A,NA,N,V,NV,ER,EI,TYPE,SUPD,NDEL,NP1,
C     1      NBLOCK,FNORM,EPS)
      RETURN
      END
$
```

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-170412		

4. Title and Subtitle	5. Report Date
Real-Time Flutter Analysis	March 1984
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Robert Walker and Naren Gupta	ISI Report 24
	10. Work Unit No.

9. Performing Organization Name and Address	11. Contract or Grant No.
Integrated Systems, Inc. 151 University Avenue Suite 400 Palo Alto, California 94301	NAS4-2955
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Contractor Report - Final
National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code
	RTOP 505-31-44

15. Supplementary Notes

NASA Technical Monitor: Glenn B. Gilyard, Ames Research Center, Dryden Flight Research Facility, Edwards, CA 93523.

16. Abstract

This report addresses the important algorithm issues necessary to achieve a real-time flutter monitoring system; namely, the guidelines for choosing appropriate model forms, reduction of the parameter convergence transient, handling multiple modes, the effect of overparameterization, and estimate accuracy predictions, both online and for experiment design. An approach for efficiently computing continuous-time flutter parameter Cramer-Rao estimate error bounds has been developed. This enables a convincing comparison of theoretical and simulation results, as well as off-line studies in preparation for a flight test. Theoretical predictions, simulation and flight test results from the NASA Drones for Aerodynamic and Structural Test (DAST) Program are compared.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Flutter analysis Flutter testing Parameter identification Recursive identification	Unclassified-Unlimited STAR category 05

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price*
Unclassified	Unclassified	124	A06

For sale by the National Technical Information Service, Springfield, Virginia 22161.

End of Document