NASA Contractor Report 3933

Real-Time Flutter Identification

Richard Roy and Robert Walker

CONTRACT NAS2-11451
OCTOBER 1985

NASA

NASA Contractor Report 3933

Real-Time Flutter Identification

Richard Roy and Robert Walker
Integrated Systems, Inc.
Palo Alto, California

Prepared for
Ames Research Center
Dryden Flight Research Facility
under Contract NAS2-11451

National Aeronautics
and Space Administration

**Scientific and Technical
Information Branch**

1985

LIST OF TABLES

LIST OF SYMBOLS

$a_{\{x,y,z\}}$	axial, lateral, and normal specific forces at the pilot location
$A(j\omega)$	frequency dependent cost on system
A_{mp}	six-vector of the amplitude of the of the vibration responses at the pilot location due to equal forces in all six directions at the hub for the m^{th} mode
A_i	lagged output coefficient matrices in autoregressive model
A_{1_S}, B_{1_S}	cyclic swashplate controls
b_i	numerator coefficient of frequency-shaping filter in discrete form
$B(j\omega)$	frequency dependent cost on system imputs
B_i	lagged input coefficients of matrices in autoregressive model with exogenous inputs
$C(k)$	controller feedback gain matrix
D	feedforward output distribution matrix
F	system dynamics matrix
$F_{\{x,y,z\}}$	axial, lateral, and normal forces at the rotor hub
G	control distribution matrix
h_i	frequency shaping filter numerator coefficients in continuous time domain
H	measurement distribution matrix
K	symmetric stiffness matrix
$\tilde{K}$	generalized stiffness matrix
$K(k)$	Kalman gain matrix
L,M,N	roll, pitch, and yaw torques at the rotor hub
M	symmetric mass matrix

$\tilde{M}$	generalized mass matrix
$P(k), P(k\|k)$	filtered state (parameter) estimate error covariance matrix at indexed time point K
$P(j\omega)$	frequency shaping filter transfer function in continuous time domain (SISO)
$P_\eta(j\omega)$	matrix square-root of frequency dependent output cost function in continuous time domain
$Q(k)$	process noise covariance matrix
R	measurement noise covariance matrix
$R_\nu(\kappa)$	innovations covariance matrix
R_{mp}	approximate amount of vibration excitation of mode m at the rotor hub assuming equal forces and moments in all directions
RSS	root-sum-square
T	matrix relating inputs and outputs in the static model of vibration valid at a single frequency only
T	force distribution matrix
$u(k)$	dynamic system control imput vector
$v(k)$	measurement noise vector
VEQ	equivalent airspeed
$x(k)$	dynamic system state vector
$y(k)$	vector of system output measurements
z	Frequency variable in discrete time domain
$\varepsilon(\kappa), \nu(\kappa)$	predicted data residuals (innovations)
ζ	damping coefficient (>0⇒ stability)
η	vector of generalized coordinates
θ	vector of parameters to be estimated in
$\theta_{c,tr}$	{collective, tail rotor} controls

$\lambda(k)$	"forgetting factor"
$\phi(k)$	vector of lagged inputs and outputs
ϕ_{mp}	vector of modal deflections at the hub
ψ	rotor shaft angle
ω	frequency variable in continuous time
ω_m	frequency of m^{th} vibration mode
Ω	fundamental vibration frequency (N/rev)
Ω_ρ	pre-warped Ω (bilinear transform)
Ω_r	rotor angular velocity
$(\cdot)_{fs}$	frequecncy shaping filter state/parameter
$(\cdot)^T$	vector/matrix transpose
$\hat{(\cdot)}$	estimated quantity

SECTION 1
INTRODUCTION

One of the important factors in the design and testing of high performance aircraft is that of aeroelastic stability. Quite often the aircraft performance envelope is limited by the flutter boundary, the set of flight conditions at which aeroelastic instability occurs. Active techniques for flutter suppression are currently under investigation; however, in order to safely test these techniques as well as to identify the flutter boundary itself, a reliable technique for real-time monitoring of the aeroelastic stability of the aircraft is required.

Under a recent contract [1] sponsored by the National Aeronautics and Space Administration Dryden Flight Research Facility, Integrated Systems, Inc. implemented a recursive prediction error (RPEM) algorithm to estimate flutter mode parameters of test aircraft in real-time. The RPEM algorithm estimates the coefficients of the numerator and denominator polynomials in a transfer function description of the system being identified. The modal frequencies and damping coefficients are estimated by computing the roots of the denominator polynomial. The algorithm was tested on simulated data and then applied to actual flight test data from the Drones for Aerodynamic Structural Testing (DAST) program. The simulated data tests involved known inputs as well as unknown random inputs (turbulence). The results of the study are documented in detail in [1].

The RPEM algorithm has several deficiencies however. The first and probably most important deficiency is the inherent single-input single-output (SISO) nature of the algorithm. Though RPEM can be extended to the multi-input multi-output (MIMO) case, the extensions result in significant over-parameterization problems. Not only does this result in increased computational requirements, but it raises issues of parameter identifiability and convergence. Another critical problem with RPEM is the inherent requirement for 'synchronous' operation. Non-uniform data rates and the 'data drop-out' problem necessitate the use of special robust estimation techniques. Since the intended application involves real-time processing of

data, recognizing and efficiently handling data drop-outs and spurious data points (or outliers) is a requirement.

Further problems with the RPEM algorithm applied to real-time flutter parameter monitoring include a potentially large sensitivity of the parameters of interest (frequencies and damping coefficients) to variations in the parameters being estimated (polynomial coefficients). The choice of model form, i.e. the transfer function or polynomial fraction description, results in an input/output parameterization which is not 'physically meaningful' and is nonlinearly related to the parameters of the internal state description desired. Though this may seem to be something of a mathematical abstraction, it is an important issue, in that, to make the estimation algorithm adaptive, a certain amount of 'tuning' is required. Tuning of adaptive algorithms is the engineering art of selecting parameters which govern trade-offs between speed of adaptation and estimation error, and this tuning is simpler when performed with physically meaningful parameters.

To develop an algorithm for providing reliable real-time estimates of flutter mode parameters, the following issues were deemed important:

1) appropriate modeling of the multi-input multi-output characteristics of the flutter identification problem,

2) estimation of the parameter estimate error variances as well as of the parameters themselves,

3) detection handling of data outliers in a real-time data processing environment,

4) capability to predict the onset of instability (and an associated uncertainty) with sufficient lead-time to allow for preventative actions to be taken,

5) and finally, algorithm flexibility and adaptability to various flight test situations such as unknown inputs and different numbers of important modes.

The MOdal Parameter IDentification (MOPID) algorithm developed and tested and discussed in this report directly addresses these issues. It overcomes, to various degrees, the inherent deficiencies of other algorithms recently applied to the flutter parameter identification problem. A brief discussion of the various algorithms applied to the flutter parameter identification problem is presented in Section 2.

1.1 SUMMARY OF APPROACH

The initial effort in this study involved the selection of candidate algorithms for real-time flutter parameter identification. The candidate algorithms included Ljung's [2] modified Extended Kalman Filter (EKF), an over-parameterized SISO RPEM algorithm, and a full EKF formulation parameterized in terms of the frequencies and damping coefficients of the dominant modes. These algorithms were then compared with respect to their ability to successfully address the issues raised above. The full EKF algorithm was selected for final implementation based primarily on its superior performance in tracking closely spaced modes (a common occurrence in the flutter identification problem) and its physically meaningful (MIMO) model parameterization. A small extension also resulted in the capability to estimate time-to-instability and its variance for each mode identified. This estimate provides valuable information during the course of a flight test, and potentially could be used in an automated instability prevention system.

The model for the flutter dynamics is based on the continuous equations for second order under-damped linear systems. The parameters in the model are the frequency and damping coefficient of the modes, i.e. the parameters of interest in the real-time flutter monitoring application. Analytic integration of these equations enables asynchronous digital operation, a feature which is required for efficient handling of potential data drop-outs and outlier problems. The algorithm tuning parameters are basically the measurement noise variance, a parameter easily estimated by inspection of the data and a priori instrument calibration, and the process noise variance density, a parameter directly related to the model uncertainty and the anticipated time-variation of the parameters being estimated. This basically allows for the inclusion of considerable a priori information which may be available concerning the expected variations of frequencies and damping coefficients with flight conditions such as Mach number and dynamic pressure. These tuning parameters also open the possibility of closed-loop adaptive operation in which the tuning parameters are made functions of the flight conditions, eg. Mach number and dynamic pressure.

Prefilters were implemented in order to avoid the necessity of data bias estimation. For the purposes of flutter mode identification, the definition of data biases is extended to include not only the measurement

device nominal zero-level, but the low-frequency center-of-mass motion of the aircraft detected in the instruments (accelerometers) as well. Elimination of these effects is mandatory to ensure unbiased estimates of the parameters of interest.

The algorithm was tested on simulated data to verify the code and to investigate its performance under various conditions. Simulated data closely resembling data for the two flight tests for which data were available were analyzed. Though extensive simulations were not performed, the results obtained verified algorithm convergence and indicated an ability to accurately track closely-spaced time-varying modes. Analysis of actual flight test data included a known input case (DAST) as well as one in which the only excitation source was turbulence (F-16). The results of the DAST data analysis indicate that moderately accurate estimates of time-to-instability can be obtained with proper tuning. The results of the F-16 data analysis manifest interesting behavior indicative of possible control surface activity. However, since no collateral information were available, the analysis was performed assuming only random inputs. Simulation data with similar spectral characteristics to the F-16 turbulence excited test data were processed and showed desirable estimation performance.

1.2 REPORT ORGANIZATION

Section 2 presents an overview of candidate algorithms for real-time parameter identification. Previous approaches to the problem of real-time flutter parameter monitoring are discussed, including the RPEM and modified EKF techniques. The full EKF as an algorithm for real-time parameter identification is discussed in Section 3. Section 4 presents the extensions to the algorithm for stability prediction and addresses the issue of data prefiltering for bias removal. The results of the simulated data analyses are presented in Section 5, followed in Section 6 by the flight test data analysis results for both the DAST and F-16 flight tests. An overview of the program organization is given in Section 7 along with some relevant operations counts, leaving the details of the input definitions and operator selections to the MOPID User's Guide in Appendix A.

SECTION 2

REAL-TIME ALGORITHMS FOR FLUTTER PARAMETER IDENTIFICATION

This section summarizes some of the important features of the real-time flutter parameter identification problem. It gives an overview of previous approaches summarizing their strengths and weaknesses and concludes with a detailed description of recursive predictive error methods (RPEM). The strengths and weaknesses of RPEM for both the single-input single-output (SISO) and multiple-input multiple-output (MIMO) systems are discussed.

2.1 FLUTTER PARAMETER IDENTIFICATION PROBLEM CHARACTERISTICS

Several aspects of the real-time flutter parameter identification problem are important in determining algorithm requirements. The primary goal is the real-time monitoring of the stability of the system being tested. The key issue is the determination of an appropriate measure of the stability and methods for obtaining estimates and estimated variances thereof, given the a priori information and data which are available. The reliable estimation of these parameters involves consideration of such factors as the number of modes (or size of the approximate dynamical model), the need to properly account for data outliers as well as data drop-outs, and the ability to adapt to changing operating conditions.

2.1.1 Stability Monitoring

There are several ways to quantify the stability of a linear (or suitably linearized) system. For SISO systems, gain and phase margins are commonly used measures of stability. Several problems arise, however, in the interpretation of these quantities once the system has become unstable, and furthermore, they are complex and for that reason not potentially useful quantities for multi-input multi-output (MIMO) systems. Fundamentally, the quantities of interest are the locations of the natural frequencies of the

system under investigation (in the s-plane) and specifically their 'distance' from the stability boundary. The commonly used damping coefficient is basically a frequency normalized distance from the $j\omega$-axis which is normally chosen as the stability boundary. Thus, an accurate (i.e. low variance) estimate of the damping coefficient of the flutter modes is a desirable algorithm output for real-time monitoring. If the estimates are of the current conditions based on the past information, the estimation (or prediction) of future values of the damping coefficient is left to the operator. However, if estimates of the rates of change of the stability measures are available, reliable prediction of future values is possible and provides the operator with information regarding estimates of the future stability of the system.

Section 3 discusses the inclusion of the rate of change of the damping coefficient into the continuous state-space model form. The use of these estimates in extending the algorithm to estimate quantities such as future values of the damping coefficients and a 'time-to-instability' is discussed in Section 4. The results presented in Sections 5 and 6 aptly demonstrate the value of these stability measures.

2.1.2 Number of Modes

Classical flutter often involves more than just a single natural frequency of the aeroelastic structure. Such structures are quite complex, involving many natural modes of oscillation, all of which change with changing flight conditions. It is not uncommon to find that, as a function of increasing dynamic pressure for example, two modes coalesce near the stability boundary, then bifurcate with one of the modes crossing into the unstable region (the right-half plane). The key point is that in such applications, it is likely that more than one mode may be important in the accurate characterization of the system dynamics (stability) in a certain region in the frequency domain. The underestimation of the number of important modes usually results in over-optimistic estimates of system stability which will in turn dramatically degrade the algorithm's ability to accurately predict future stability.

2.1.3 Algorithm Start-up and Reset Requirements

Although a priori modal analysis is available for nearly every flight test vehicle, the modal results are approximate at best and therefore one requirement of an algorithm is that it have a robust start-up procedure. Secondly, the algorithm should be capable of being reset during a flight test should it get stuck in a local minimum which is known not to be a global minimum (such as higher harmonics of a fundamental frequency) or diverge because of some abrupt change that it was not able to track. In order to facilitate verification of proper initialization and/or respecification of parameters, they should be as physically meaningful as possible. This is best exemplified by choosing a parameterization in terms of frequencies and damping coefficients rather than coefficients in a polynomial whose roots are the characteristic frequencies! This helps prevent errors during algorithm set-up.

2.1.4 Parameter Variations

The basic objective in the testing of aeroelastic structures is to obtain information from which an accurate description of the system dynamics can be reconstructed throughout the entire flight envelope. Since aeroelastic dynamics are inherently nonlinear, current methods involve linearization of the system at various points throughout the flight envelope, with particular emphasis placed on regions near the stability boundary. As the flight conditions change, so do the parameters in the linearized models. Thus, it is important that any real-time algorithm have the capability to adapt to changing conditions; specifically the algorithm should at least be capable of tracking variations of the parameters in its dynamical model as a function of time. This requirement is most easily satisfied by a class of algorithms known as recursive algorithms. The parameters are updated each time a measurement of the system output becomes available. Though batch processing algorithms can be modified to operate in a pseudo real-time environment by decreasing the batch size and performing sequential batch analysis, the computational requirements are in general too great, and the issue of how often to update estimates and the appropriate batch size is

difficult to resolve in real-time. There is an obvious trade-off between timeliness of estimates, their estimated variance, and computational load. It should be mentioned, however, that batch algorithms are capable of yielding lower variance estimates of desired parameters since they are in effect including all the information in the batch interval in estimating the parameters at each point in the interval (i.e fixed-interval smoothing). The increased computational cost and attendant delay in producing estimates of the critical parameters is not currently justified, however, in light of the limited improvement in estimation accuracy, especially in the 'current' parameter estimates. In fact, the parameter estimates at the endpoint of the interval are not improved at all by smoothing since they already contain all the information over the entire interval.

The characteristics of the flutter identification problem discussed above affect the choice of model form and its parameterization, the choice of a method for updating the parameter values as new data become available, and also influence the choice of numerical technique. These issues are discussed further in a review of some recent approaches given below. Gilyard and Edwards [3] discuss some of the issues involved in using FFT based techniques for on-line flutter parameter estimation. A significant point that they make is that since flight conditions are not exactly repeatable, output responses cannot be overlaid to average out the effects of noise. Therefore, recent techniques have tended to use time-domain based models which allow incorporation, in a statistical sense, of the effects of both measurement noise and turbulence or unknown input noise.

2.2 OVERVIEW OF PREVIOUS APPROACHES

Russo and his colleagues at Grumman Data Systems [4] applied a complete flight-testing maximum likelihood parameter identification technique to the flutter estimation problem. They used a modal coordinate continuous state-space model form and propagated a Kalman filter for the discrete measurement model. This required propagating sensitivities with respect to all of the parameters of the Kalman filter equations as well as the equations themselves. They experimented with different model structures

choosing them based on analysis of power spectral densities. This approach required multiple passes through the data to obtain reasonable parameter estimates. The parameters are updated after each pass using a modified Newton method. This is a classic approach to a powerful identification method which is fundamentally, however, an off-line batch, rather than an on-line recursive technique. Another factor is that extensive computation is required to propagate the Kalman filter sensitivity equations. The advantage of the method is that it works in physical modal coordinates so that no factorization of polynomials is required to determine the damping of the identified modes. A discussion of the use of an array processor was included by the authors, however current array processors are not particularly well suited to iterative time-domain equations. There has been research done on specialized processors that are well suited to these sorts of covariance propagations, however, other algorithms address the problem much more directly with a recursive rather than a semi-block method.

Wendler of Lockheed [5] used a recently derived identification algorithm and implemented it in a manner which does not require factorization of a polynomial. A lattice algorithm was employed to identify the parameters of an autoregressive (AR) model of the input-output relationship. The lattice algorithm identifies reflection coefficients which are coordinates of an orthogonal basis and are numerically well-conditioned. A least squares regression of each reflection coefficient against dynamic pressure was then performed to extrapolate and predict near-instability. The autoregressive model order was determined using an energy threshold and counting peaks in a periodogram. It should be pointed out, however, that this is not an on-line approach. Furthermore, the lattice algorithm is designed for stationary processes and under certain conditions yields only stable estimates of the system. As the system approaches the flutter boundary, the performance of the algorithm degrades and it is incapable of yielding accurate estimates past the flutter boundary. It should also be noted that in the implementation discussed, no provision was made for the inclusion of exogenous inputs. The algorithm was designed to handle the unknown input case only though this need not be the case.

An advantage of the lattice formulation is that it can be extended to actually increase the model order as well as updating the parameters on-line. However, while the AR parameterization is perhaps better conditioned

numerically than an autoregressive moving average (ARMA) formulation, it shares a number of model form difficulties with the ARMA formulation discussed in the next subsection.

Molusis and Kleinman [6] used an autoregressive moving average model to perform recursive maximum likelihood identification of parameters of a second-order system for on-line rotorcraft elastic mode identification. From the second-order model parameter estimates, they estimated both the frequency and damping coefficients and their variances. A bandpass filter was used to separate the desired mode from possible interfering modes. When closely spaced modes were present, this approach experienced obvious difficulties. Bandpass filters cannot separate closely spaced modes. Since the autoregressive moving average with exogenous input model, or ARMAX model, is basically identifying the coefficients of the characteristic polynomial, and since it is well-known that the roots of such polynomials are very sensitive to the coefficients, unwanted contributions from closely spaced modes can lead to large errors in estimated pole locations.

2.3 RECURSIVE PREDICTION ERROR METHODS

Recursive prediction error methods (RPEM) constitute a class of algorithms which are based on the minimization of a prediction error. This error is most often the difference between the measurement and a one-step ahead prediction of the measurement. RPEM can be applied to a variety of model forms including those with known, or exogenous inputs as well as random inputs. The choice of model form and parameterization depend upon the details of the problem to be solved.

Independently from Molusis and Kleinman [6], Walker and Gupta [1] applied this algorithm to the real-time flutter parameter estimation problem. The analysis addressed the difficulty of multiple modes by overparameterization of the model; basically estimating a number of parameters in excess of the number required. This was also done in an attempt to minimize the sensitivity of the desired root locations to the coefficients being estimated. Simulations were performed for two-mode models with both known and unknown

inputs. In addition, a novel approach was developed to compute the Cramer-Rao estimate of the variance of the flutter parameters. Flight data were also processed for conditions with unknown inputs (turbulence excitation) and with swept sine-wave and pulse or doublet inputs. The problem of data outliers was addressed using robust estimation techniques. Software implementing this approach was delivered to NASA for further analysis.

2.3.1 A Modified EKF Extension of RPE Methods

Lennart Ljung [2] has shown the equivalence of a modified extended Kalman filter algorithm for identifying parameters of a linear model, with the ARMAX model form used in recursive prediction error methods. There were three reasons for evaluating this modified EKF for the real-time flutter identification problem.

1) It is capable of handling the multiple-input multiple-output (MIMO) nature of the problem.

2) The modified EKF is potentially computationally less expensive than a full EKF.

3) The modified EKF does not require that the model form be an ARMAX form. A modal form can be used to estimate frequency and damping coefficients directly rather than polynomial coefficients.

The equations for this modified EKF are given below.

$$x(t+1) = F(t)x(t) + G(t)u(t) + K(t)\nu(t)$$

$$y(t) = H(t)x(t) + \nu(t)$$

$$\nu(t) = y(t) - \hat{y}(t)$$

$$\hat{\Lambda}(t) = \hat{\Lambda}(t-1) + \gamma(t)[\nu(t)\nu^T(t) - \hat{\Lambda}(t-1)]$$

$$R(t) = R(t-1) + \gamma(t)[\psi(t)\,\hat{\Lambda}^{-1}\psi(t) - R(t-1)]$$

$$\hat{\theta}(t) = \hat{\theta}(t-1) + \gamma(t)\,R^{-1}(t)\psi(t)\,\hat{\Lambda}^{-1}(t)\,\nu(t)$$

Update F, G, H, K, $\bar{M}$, D as functions of $\hat{\theta}(t)$, and then obtain:

$$\hat{x}(t+1) = F_t\hat{x}(t) + G_t u(t) + K_t v(t)$$

$$\hat{y}(t+1) = H_t\ \hat{x}(t+1)$$

$$W(t+1) = [F_t - K_t H_c]W(t) + \bar{M}_t - K_t D_t$$

$$\psi(t+1) = W^T(t+1)H_t^T + D^T(\hat{\theta}(t),\ x(t+1))$$

where

$$\psi^T(t,\theta) = \frac{d}{d\theta}\,\hat{y}\,(t|\theta)\ ; \qquad D(\theta,\ x) = \frac{\partial}{\partial\theta}\,[H(\theta)x]\Big|_{\theta=\hat{\theta}}\ ;$$

$$W(t,\theta) = \frac{d}{d\theta}\,\hat{x}(t,\theta)\ ; \qquad \bar{M} = \frac{\partial}{\partial\theta}\,[\hat{x}(t+1,\ \theta)]\Big|_{\theta=\hat{\theta}}\ .$$

A forgetting factor (λ) determines a weighting factor (γ) in the equations which effectively weights recent data more heavily than past data. A γ enters the covariance equation as well as the parameter updates equations, and directly effects the parameter convergence rate.

Initial studies with the MIMO modified EKF algorithm performed well on several simulation examples. These results were achieved without estimating the parameters in the the Kalman gain matrix in the innovations form of the equations even though process noise was present. This approach can be used as long as the plant remains stable. For unstable plants, the Kalman gain must be estimated to prevent filter divergence. This, however, results in an unacceptable increase in the number of parameters which must be estimated since the algorithm is to run in a real-time environment. The solution to this problem was the implementation of a full extended Kalman filter which propagates the covariance of the states as well as the parameters, including the correlation between the two, rather than only propagating the covariance of the parameters as the modified EKF does. The Kalman gain matrix is calculated as a function of various other parameters in the algorithm rather than being estimated on-line.

There are several aspects of the full EKF formulation worthy of further comment. In the modified EKF formulation, estimation of the parameters in the Kalman gain matrix does not guarantee the stability of the resulting filter equations. This implies the necessity for periodic testing of the filter stability requiring an eigendecomposition or singular-value decomposition of a matrix the size of the state dimension. In either case this amounts to a significant computational burden. The full EKF propagates the

associated Riccati equation which guarantees the stability of the resulting filter equations. Secondly, the process noise variance density matrix used in the 'tuning' of the full EKF has a number of degrees of freedom equal to the dimension of the state vector, whereas the modified EKF equivalent which is the forgetting factor is a scalar quantity. The result is that in the modified EKF, the prediction of the state covariance is performed by multiplicative modification of the filtered covariance matrix (by a scalar times the identity matrix) whereas selective parameter variance augmentation can be performed in the full EKF algorithm. This allows the inclusion of a priori information concerning the relative dynamics of the parameter variations (at least in a statistical sense). The remainder of this report develops this approach and its software implementation.

SECTION 3

THE EXTENDED KALMAN FILTER AS A REAL-TIME FLUTTER PARAMETER IDENTIFIER

This section discusses the application of extended Kalman filtering to the problem of real-time flutter parameter identification with emphasis on frequency and damping coefficient estimation. The equations governing the algorithm are presented without rigorous derivation. Derivations can be found in the references ([10],[13],[14]) and are not repeated here for the sake of brevity. The section begins with a discussion of the simplified model chosen to approximate the complex nonlinear dynamics of aeroelastic structures. The equations are given in continuous-time state space form and the conversion to discrete-time is performed. The augmentation of the state with the parameters to be identified is discussed and the Kalman filter equations are presented. The implementation of these equations in square-root form is also discussed. Issues such as prefiltering of the data and algorithm extension for stability prediction are deferred to the next section.

3.1 FLUTTER DYNAMICS MODELING

A key concept in the development of simplified dynamical models is that of prediction. One of the primary reasons for developing such models is to be able to predict the outputs of the underlying system given the inputs. Quantitative measures of the model's ability to predict the future are often used as criteria for the selection of model forms and their parameterization. The more complex the model, the greater its ability to accurately predict future outputs of the system. The price of increased accuracy, however, is an increased computational load.

A fundamental trade-off between prediction accuracy and model (computational) complexity exists as attempts are made to predict farther and farther into the future. For most physical systems, as the interval over which the models are required to predict decreases, so does the relative error. Furthermore, in applications such as real-time flutter

monitoring, corrective information in the form of measurements of the system outputs is available and can be used in a recursive manner to keep the prediction errors small. In such situations, simplified models can be entirely adequate if the input information rate is sufficient, since the overall objective is to monitor the stability of the system. Linearized models can give more than adequate predictions, obviating the need for complex predictive models of the phenomenon.

From the class of linear models, the choice of a continuous-time second-order system parameterized in terms of its natural frequency (ω) and its damping coefficient (ζ) is appropriate. Equation 3-1 gives an input-output continuous-time state-space description of a single mode dynamical system with two inputs and two outputs. The effect of possible process noise terms (i.e. disturbances and unknown inputs) is also indicated, and measurement noise is included in the output equation.

$$\begin{aligned} \dot{\underline{x}}(t) &= F_x^c \, \underline{x}(t) + G_u^c \, \underline{u}(t) + G_w^c \, \underline{w}^c(t) \\ \underline{z}(t) &= H_x^c \, \underline{x}(t) + D_u^c \, \underline{u}(t) + \underline{v}^c(t) \end{aligned} \qquad (3\text{-}1a)$$

where the superscript "c" is used to denote continuous-time. For a 2-input, 2-output, single-mode system (2,2,1) with independent noise processes forcing each state, the system matrices can be written in the following form:

$$F_x^c = \begin{bmatrix} 0 & 1 \\ -\omega^2 & -2\zeta\omega \end{bmatrix}$$

$$G_u^c = \begin{bmatrix} g_{11} & g_{12} \\ g_{21} & g_{22} \end{bmatrix}$$

$$G_w^c = \begin{bmatrix} g_{w_1} & 0 \\ 0 & g_{w_2} \end{bmatrix}$$

(3-1b)

$$H_x^c = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix}$$

$$D_u^c = \begin{bmatrix} d_{11} & d_{12} \\ d_{21} & d_{22} \end{bmatrix}$$

$$\underline{w}^c \sim N(0, \mathrm{diag}\{q_{x_i}\})$$
$$\underline{v}^c \sim N(0, \mathrm{diag}\{r_i\}).$$

The inclusion of another mode would increase the state dimension to four (4). The systems dynamics matrix (**F**) becomes block diagonal; a 2x2 block with the form given in equation 3-1 describes the dynamics of each mode. Notably the number of parameters in the control distribution matrix (**G**) and the measurement distribution matrix (**H**) double, and in general increase linearly with the number of states.

Since the measurement devices can be accelerometers mounted on the wings near the control surfaces used for modal suppression, a direct feedthrough term is included in the measurement equation. This term accounts for the direct effect of control actuator motion on accelerometer outputs. The noise processes ($\underline{\mathbf{v}}$) and ($\underline{\mathbf{w}}$) are assumed to be independent random Gaussian processes with zero-mean and the indicated spectral densities. Though the independence assumption can easily be removed and correlations included, the resulting computational burden is not warranted. Note that measurement noise correlation could be introduced by linear operations on the data prior to identification such as summing and differencing the outputs, but that such operations are not required since the algorithm is fully capable of handling the general MIMO identification problem.

The process noise inputs are included to account for the effects of unknown random disturbances as well as model uncertainties. The random disturbances are generally in the form of turbulence and gusts. The model uncertainties are an attempt to quantitatively and statistically measure the

accuracy with which the simplified model of flutter dynamics is able to predict future outputs.

Since the underlying physical process in the real-time flutter monitoring problem is certainly a continuous-time process, a continuous-time description and parameterization are appropriate. However, the measurements processed are in the form of sampled data vectors. Thus, the real-time flutter parameter identification problem is a continuous-discrete problem. The predicted outputs of the system model are required at discrete times, those at which the measurements are made. This requires integration of the underlying continuous system equations, which can be accomplished in many ways. For the class of models employed, however, analytic integration is by far the most accurate and is not computationally intensive. Integration of the state-space equations in equation 3-1 yields the following set of discrete-time state-space equations:

$$\begin{aligned} \underline{x}(k+1) &= F_x^d\, \underline{x}(k) + G_u^d\, \underline{u}(k) + G_w^d\, \underline{w}^d(k); \\ \underline{z}(k) &= H_x^d\, \underline{x}(k) + D_u^d\, \underline{u}(k) + \underline{v}^d(k); \end{aligned} \tag{3-2a}$$

where the discrete analogs of the continuous-time system matrices can be written as follows:

$$F_x^d = e^{-\zeta\omega\Delta} \begin{bmatrix} \cos(\beta\Delta) + \frac{\zeta\omega}{\beta}\sin(\beta\Delta) & \frac{1}{\beta}\sin(\beta\Delta) \\ -\frac{\omega^2}{\beta}\sin(\beta\Delta) & \cos(\beta\Delta) - \frac{\zeta\omega}{\beta}\sin(\beta\Delta) \end{bmatrix};$$

$$G_u^d = \Gamma_x\, G_u^c;$$

$$G_w^d = \Gamma_x\, G_w^c; \tag{3-2b}$$

$$\Gamma_x = \begin{bmatrix} I_c + \frac{\zeta\omega}{\beta} I_s & \frac{1}{\beta} I_s \\ -\frac{\omega^2}{\beta} I_s & I_c - \frac{\zeta\omega}{\beta} I_s \end{bmatrix}$$

$$H_x^d = H_x^c ;$$

$$D_u^d = D_u^c ;$$

and the following definitions have been employed for notational simplicity;

$$I_c = \int_o^\Delta e^{-\zeta\omega\tau} \cos(\beta\tau)\, d\tau;$$

$$I_s = \int_o^\Delta e^{-\zeta\omega\tau} \sin(\beta\tau)\, d\tau;$$

$$\Delta = t_{k+1} - t_k;$$

$$\beta = \omega\sqrt{1-\zeta^2}$$

Note that in performing the integration, the noise processes are 'integrated' as well. However, under the assumption that they are white Gaussian noise processes, the integral can not be performed in the traditional sense; nor is it required to be performed since the mean value, or expectation, is zero. In tuning of the filter, however, the value of the spectral density is input and it is assumed to be the variance density of the continuous-time noise process. In order for the resulting state estimate error variances to be approximately the same for the continuous and and discrete systems at the sampling times, there is a factor of the sampling time which appears in the formulation, i.e.

$$q^d \sim q^c\Delta;$$

where q^c is the continuous process noise variance density, and q^d is the corresponding discrete process noise variance.

Given the equations describing the simplified dynamical model, and given the set of measurements, the basic idea is to use the measurements at each sample time to correct the estimated outputs of the simplified model to more closely correspond to the 'true' system outputs. Taking into account the stochastic nature of the input and output processes, an 'optimal' estimation problem can be formulated whose solution is the well-known Kalman

filter. There are extensive references on the subject so the derivation of the equations will not be repeated here. However, the underlying concept is that during the prediction step information is lost (the covariance of the states increases due to the addition of the process noise covariance) and that during the measurement update step information is gained, or extracted from the measurements. Remembering these intuitive concepts is helpful when adjusting the tuning parameters as will be discussed later.

Thus far, it has been assumed that the parameters ω and ζ are known, as well as the control distribution matrix elements g_{ij}. If they are not known, they too must be estimated along with the dynamic states. Augmenting the state vector with these parameters to be estimated (identified) is the additional idea behind the extended Kalman filter (EKF). The term extended is used to indicate that the estimation problem is now a nonlinear one. In the case of the control and measurement distribution matrix parameters, it is bilinear in the states and parameters; but in the frequencies and damping coefficients the nonlinearity is transcendental (cf. sine, cosine and exponential functions).

Augmenting the state vector $\underline{x}$ with a vector $\underline{\theta}$ containing parameters to be identified also requires the specification of a dynamical model for predicting the parameters as a function a time. If the parameters are truly time-invariant (as the connotation of the word 'parameter' suggests), then in discrete-time the state transition matrix is simply the identity matrix. If a more detailed model is appropriate or desired, it can be included. In an effort to provide reliable real-time estimates of <u>future</u> values of the flutter mode stability, a slightly more complex model than the zero time-derivative model is employed. The parameter vector is augmented further by the addition of a damping coefficient velocity (ζ_v) state; i.e. instead of assuming that the rate of change (or time derivative) of the damping coefficient is zero, it is assumed to be a constant (ζ_v) to be estimated as well. The dynamical model for ζ_v is assumed to be the zero time-derivative model. Assuming the same model for the frequency as well results in the following set of continuous-time equations:

$$\begin{bmatrix} \dot{\zeta} \\ \dot{\zeta}_v \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \zeta \\ \zeta_v \\ \omega \end{bmatrix} + \begin{bmatrix} w_\zeta^c \\ w_{\zeta_v}^c \\ w_\omega^c \end{bmatrix} \qquad (3\text{-}3)$$

The integration of these equations yields the following set of difference equations for the prediction of the flutter parameters:

$$\begin{bmatrix} \zeta(k+1) \\ \zeta_v(k+1) \\ \omega(k+1) \end{bmatrix} = \begin{bmatrix} 1 & \Delta & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \zeta(k) \\ \zeta_v(k) \\ \omega(k) \end{bmatrix} + \begin{bmatrix} w_\zeta^d \\ w_{\zeta_v}^d \\ w_\omega^d \end{bmatrix}. \qquad (3\text{-}4)$$

Finally, assuming that the G-matrix and H-matrix elements to be identified are also time-invariant, we have:

$$\dot{p} = w_p^c \qquad (3\text{-}5)$$

which in discrete-time can be written:

$$p(k+1) = p(k) + w_p^d(k). \qquad (3\text{-}6)$$

The process noise is included to account for possible parameter time-variations which are unknown. The variance density of these processes is directly related to the expected variation of the parameters as a function of time. The ability to change the variance density as a function of time allows for tracking of step-like changes in the parameters (such as is the case in the flutter problem when the aircraft configuration changes due to a release of stores). This is discussed in greater detail in the sequel.

3.2 EXTENDED KALMAN FILTER EQUATIONS

The equations described in the previous subsection are stochastic in nature and are models for the evolution of the 'true' state of the system. However, the true state is not known nor can it be computed; the idea is to obtain a 'best' estimate of the true state given all of the available information. If a 'best' estimate of the state at the present time is assumed to be available, the best estimate of the state at a future time, assuming no measurements are available during the time interval, is found by taking the expected value of the equations above. Combining the state and parameter propagation equations for 2-modes, 2-inputs, and 2-outputs, the full (2,2,2) system of equations can be written as shown in equation 3-7. The symbol "^" is used to denote an estimated quantity. Variables without ^'s are assumed to be known or given. Unspecified elements of the matrices are assumed to be zero. Note that in the formation of the state vector, no account was taken of the critical issue of which parameters are actually estimable from the given information. This issue will be discussed in detail later in this section. For the (2,2,2) problem formulated, this state vector contains the union of all possible sets of parameters to be identified.

$$
\begin{bmatrix}
\hat{x}_{1_1} \\ \hat{x}_{2_1} \\ \hat{x}_{1_2} \\ \hat{x}_{2_2} \\ \hat{\zeta}_1 \\ \hat{\zeta}_{v_1} \\ \hat{\omega}_1 \\ \hat{\zeta}_2 \\ \hat{\zeta}_{v_2} \\ \hat{\omega}_2 \\ \hat{g}_{11_1} \\ \hat{g}_{21_1} \\ \hat{g}_{11_2} \\ \hat{g}_{21_2} \\ \hat{g}_{12_1} \\ \hat{g}_{22_1} \\ \hat{g}_{12_2} \\ \hat{g}_{22_2} \\ \hat{d}_{11} \\ \hat{d}_{12} \\ \hat{d}_{21} \\ \hat{d}_{22}
\end{bmatrix}_{(k+1)} =
\left[\begin{array}{cc|c}
F^d_{x_1} & & \\
 & F^d_{x_2} & 0 \\
 & F^d_{\zeta\omega} & \\
 & F^d_{\zeta\omega} & \\
\hline
0 & & I
\end{array}\right]
\begin{bmatrix}
\hat{x}_{1_1} \\ \hat{x}_{2_1} \\ \hat{x}_{1_2} \\ \hat{x}_{2_2} \\ \hat{\zeta}_1 \\ \hat{\zeta}_{v_1} \\ \hat{\omega}_1 \\ \hat{\zeta}_2 \\ \hat{\zeta}_{v_2} \\ \hat{\omega}_2 \\ \hat{g}_{11_1} \\ \hat{g}_{21_1} \\ \hat{g}_{11_2} \\ \hat{g}_{21_2} \\ \hat{g}_{12_1} \\ \hat{g}_{22_1} \\ \hat{g}_{12_2} \\ \hat{g}_{22_2} \\ \hat{d}_{11} \\ \hat{d}_{12} \\ \hat{d}_{21} \\ \hat{d}_{22}
\end{bmatrix}_{(k)}
$$

$$+ \begin{bmatrix} \Gamma_{x_1} & 0 \\ 0 & \Gamma_{x_2} \\ \hline & 0 \end{bmatrix} \begin{bmatrix} g_{11_1} & g_{12_1} \\ g_{21_1} & g_{22_1} \\ g_{11_2} & g_{12_2} \\ g_{21_2} & g_{22_2} \end{bmatrix} \begin{bmatrix} u_1(k) \\ u_2(k) \end{bmatrix} \tag{3-7}$$

where

$$F^d_{\zeta\omega} = \begin{bmatrix} I & \Delta & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \quad ;$$

and

$$\Gamma_{x_i} = \Gamma_x(\omega_i, \zeta_i).$$

In order to reduce the complexity of the EKF equations which follow, some notation is helpful. First, let $\underline{X}$ denote the entire state vector composed of the dynamic states $\underline{x}$ and the parameters $\underline{\theta}$ to be estimated.

$$\begin{aligned} \underline{X}^T &= [\underline{x}^T, \underline{\theta}^T]; \\ \underline{\theta}^T &= [\zeta_1, \zeta_{v_1}, \omega_1, \ldots, g_{ij_k}, \ldots, h_{lm_n}, \ldots] \end{aligned} \tag{3-8}$$

As before, the symbol "^" denotes an estimated quantity. The notation $(j|k)$ is used to indicate that the associated quantity is an estimate at time j given data up to and including time k. Thus, $(i+1|i)$ indicates a one step ahead prediction and $(i|i)$ a filtered estimate. The entire state transition matrix is given by ϕ. P is used to denote the state estimate error covariance matrix; **K** the Kalman gain matrix; and $\underline{\nu}$ the innovations or predicted data residuals. The time-update, or prediction step, of the EKF is then given by the following set of equations.

$$\hat{\underline{X}}\,(k+1|k) = \hat{\phi}(k+1,k)\underline{X}(k|k) + \hat{G}_u^d\,\underline{u}(k)$$

$$P(k+1|k) = E\{[\underline{X}(k+1) - \hat{\underline{X}}(k+1|k)][(\underline{X}(k+1) - \hat{\underline{X}}(k+1|k)]^T\} \qquad (3\text{-}9)$$

$$= [\partial_{\underline{X}}\phi]\; P(k|k)\; [\partial_{\underline{X}}\phi]^T + Q^d(k+1,k)$$

where the notation $\partial_\alpha\phi = \frac{\partial\phi}{\partial\alpha}$ has been used.

Though complex to derive, the partial derivatives can be calculated analytically. The derivations are not given, however the expressions for the derivates are, as they are important factors in the EKF equations.

$$\partial_{\underline{X}}\phi = \begin{bmatrix} \partial_{\underline{X}}[F_x^d\,\underline{x} + G_u^d\underline{u}] \\ \hline 0 \qquad F_{\zeta\omega}^d \qquad 0 \\ \\ 0 \qquad 0 \qquad I \end{bmatrix}$$

Defining;

$$c(\beta\Delta) = \cos(\beta\Delta)\ ,$$
$$s(\beta\Delta) = \sin(\beta\Delta)\ ,$$
$$\beta = \omega\sqrt{(1-\zeta^2)}\ ;$$

we can write (omitting the mode subscript "i" for clarity):

$$F_x^d = \begin{bmatrix} e^{-\xi\omega\Delta} & c(\beta\Delta) + \frac{\zeta\omega}{\beta}\,s(\beta\Delta) & \frac{s(\beta\Delta)}{\beta} \\ & -\frac{\omega^2 s(\beta\Delta)}{\beta} & c(\beta\Delta) - \frac{\zeta\omega}{\beta}\,s(\beta\Delta) \end{bmatrix}$$

$$= e^{-\zeta\omega\Delta}\mathbf{f}.$$

For $\alpha \in \{\omega_i, \zeta_i\}$, we have;

$$\partial_\alpha F_x^d = (\partial_\alpha e^{-\zeta\omega\Delta})\,\mathbf{f} + e^{-\zeta\omega\Delta}\;\partial_\alpha\mathbf{f}\ ,$$

where;

$$\partial_\alpha\mathbf{f} = \begin{bmatrix} \partial_\alpha c(\beta\Delta) + \partial_\alpha\frac{\zeta\omega s(\beta\Delta)}{\beta} & \partial_\alpha\frac{s(\beta\Delta)}{\beta} \\ -\partial_\alpha\frac{\omega^2 s(\beta\Delta)}{\beta} & \partial_\alpha c(\beta\Delta) - \partial_\alpha\frac{\zeta\times s(\beta\Delta)}{\beta} \end{bmatrix}.$$

Using;

$$\partial_\alpha c(\beta\Delta) = -\Delta s(\beta\Delta)\partial_\alpha \beta$$

$$\partial_\alpha s(\beta\Delta) = \Delta c(\beta\Delta)\partial_\alpha \beta$$

$$\partial_\omega \beta = \sqrt{(1-\zeta^2)}$$

$$\partial_\zeta \beta = \frac{-\zeta\omega^2}{\beta},$$

the partials are calculated;

$$\partial_\omega \mathbf{F}_x^d = -\zeta\Delta \mathbf{F}_x^d + e^{-\zeta\omega\Delta} \begin{bmatrix} -\Delta s(\beta\Delta)\sqrt{(1-\zeta^2)} + \zeta\Delta c(\beta\Delta) & \frac{\beta\Delta c(\beta\Delta) - s(\beta\Delta)}{\beta\omega} \\ -\frac{\omega s(\beta\Delta) + \omega\beta\Delta c(\beta\Delta)}{\beta} & -\Delta s(\beta\Delta)\sqrt{(1-\zeta^2)} - \zeta\Delta c(\beta\Delta) \end{bmatrix}$$

$$\partial_\zeta \mathbf{F}_x^d = -\omega\Delta \mathbf{F}_x^d + e^{-\zeta\omega\Delta} \begin{bmatrix} -\Delta s(\beta\Delta)\partial_\zeta\beta + \frac{\omega s(\beta\Delta)}{\beta} + \zeta\omega\partial_\zeta\frac{s(\beta\Delta)}{\beta} & \partial_\zeta\frac{s(\beta\Delta)}{\beta} \\ -\omega^2\partial_\zeta\frac{s(\beta\Delta)}{\beta} & -\Delta s(\beta\Delta)\partial_\zeta\beta - \frac{\omega s(\beta\Delta)}{\beta} - \zeta\omega\partial_\zeta\frac{s(\beta\Delta)}{\beta} \end{bmatrix}$$

where $$\partial_\zeta \frac{s(\beta\Delta)}{\beta} = -\frac{\zeta\omega^2}{\beta^2}\left[\Delta c(\beta\Delta) - \frac{s(\beta\Delta)}{\beta}\right]$$

Since the discrete-time input distribution matrix G_u^d is a function of the modal parameters via the term Γ_x, its partials with respect to these parameters are required as well. Using the fact that $\frac{\omega}{\beta}$ is not a function of ω, we can write

$$\partial_\omega \Gamma_x = \begin{bmatrix} \partial_\omega I_c + \frac{\zeta\omega}{\beta}\partial_\omega I_s & -\frac{\partial_\omega \beta}{\beta^2} I_s + \frac{\partial_\omega I_s}{\beta} \\ -\frac{\omega}{\beta} I_s - \frac{\omega^2}{\beta}\partial_\omega I_{s_s} & \partial_\omega I_s - \frac{\zeta\omega}{\beta}\partial_\omega I_s \end{bmatrix}$$

and,

$$\partial_\zeta \Gamma_x = \begin{bmatrix} \partial_\zeta I_s + \frac{\zeta\omega}{\beta}\partial_\zeta I_s + \partial_\zeta(\frac{\zeta\omega}{\beta})I_s & \frac{\partial_\zeta I_s}{\beta} - \frac{\partial_\zeta \beta}{\beta^2} I_s \\ \frac{\omega^2}{\beta^2}(I_s\partial_\zeta\beta - \beta\partial_\zeta I_s) & \partial_\zeta I_c - I_x\partial_\zeta(\frac{\zeta\omega}{\beta}) - \frac{\zeta\omega}{\beta}\partial_\zeta I_s \end{bmatrix} ;$$

where further shortening notation by defining $c = c(\beta\Delta)$ and $s = s(\beta\Delta)$:

$$\partial_\zeta I_c = \frac{1}{\omega} - \frac{\Delta}{\omega} e^{-\zeta\omega\Delta}(\beta s - \zeta\omega c)$$

$$+ \frac{1}{\omega^2} e^{-\zeta\omega\Delta}(c\partial_\zeta\beta + \beta\partial_\zeta s - \omega c - \zeta\omega\partial_\zeta c)$$

$$\partial_\zeta I_s = \frac{\partial_\zeta\beta}{\omega^2} + \frac{\Delta}{\omega^2} e^{-\zeta\omega\Delta}(\zeta\omega s + \beta c)$$

$$- \frac{1}{\omega^2} e^{-\zeta\omega\Delta}(\omega s + \zeta\omega\partial_\zeta s + c\partial_\zeta\beta + \beta\partial_\zeta c)$$

$$\partial_\omega I_c = \frac{\zeta}{\omega^2} + \frac{e^{-\zeta\omega\Delta}}{\omega^2}(\zeta\Delta + \frac{2}{\omega})(\zeta\omega c - \zeta\omega\partial_\omega c)$$

$$+ \frac{1}{\omega^2} e^{-\zeta\omega\Delta}(s\partial_\omega\beta + \beta\partial_\omega s - \zeta c - \zeta\omega\partial_\omega c)$$

$$\partial_\omega I_s = -\frac{\beta}{\omega^3} + \frac{e^{-\zeta\omega\Delta}}{\omega^2}(\zeta\Delta + \frac{2}{\omega})(\zeta\omega s + \beta c)$$

$$- \frac{1}{\omega^3} e^{-\zeta\omega\Delta}(\zeta s + \zeta\omega\partial_\omega s + c\partial_\omega\beta + \beta\partial_\omega c) .$$

If $\underline{\theta}$ contains elements of the control distribution matrix g_{ij_k}, then further partial calculations are required. The terms required are:

$$\partial_\alpha(\Gamma_x G_u^d \underline{u}) = \Gamma_x(\partial_\alpha G_u^d)\underline{u} .$$

Denoting the (i,j)-th element of a matrix by $(\cdot)_{ij}$ and using the standard repeated index summation convention, we have;

$$\partial_\alpha(\Gamma_x G^d_{\underline{u}}) = (\Gamma_x)_{ij}(\partial_\alpha G^d_{\underline{u}})_{jk}(\underline{u})_k \ .$$

Now for $\alpha = g_{lm}$,

$$(\partial_\alpha G^d_{\underline{u}})_{jk} = \delta^{lm}_{jk}$$

where $\delta^{lm}_{jk} = \begin{matrix} 1 \ ; & j = l \text{ and } k = m \\ 0 \ ; & \text{otherwise} \end{matrix}$.

Therefore;

$$\partial_{g_{lm}}(\Gamma_x G^d_{\underline{u}}\ \underline{u}) = (\Gamma_x)_{il}\ (\underline{u})_m \ .$$

The discrete process noise covariance matrix required for the time-update of the state estimate error covariance matrix is calculated as follows:

$$Q^d(k+1,k) = \begin{bmatrix} Q^d_{x_1} & & & & & \\ & Q^d_{x_2} & & 0 & & \\ & & Q^d_{\zeta\omega_1} & & & \\ & & & Q^d_{\zeta\omega_2} & & \\ & 0 & & & & \\ & & & & & Q^d_p \end{bmatrix}$$

where;

$$Q^d_{x_i} = \int_0^\Delta F^d_{x_i}(\tau)\ G^c_w\ Q^c_{x_i}\ G^c_w\ F^{d^T}_{x_i}(\tau)\ d\tau$$

$$Q^c_{x_i} = \text{diag}\{q_{x_1}, q_{x_2}\};$$
$$\qquad\qquad\quad i \quad i$$

which can be approximated by:

$$Q^d_{x_i} \approx G^c_w Q^c_{x_i} G^{c^T}_w \Delta + F^c_{x_i} G^c_w Q^c_{x_i} G^{c^T}_w \frac{\Delta^2}{2} + G^c_w Q^c_{x_i} G^{c^T}_w F^{c^T}_{x_i} \frac{\Delta^2}{2} + F^c_{x_i} G^c_w Q^c_{x_i} G^{c^T}_w F^{c^T}_{x_i} \frac{\Delta^3}{3} + \ldots$$

for the dynamic states. For the parameter states, the process noise covariance matrix can be calculated exactly as follows (since **F** is nilpotent):

$$\mathbf{Q}^{d}_{\zeta\omega_i} = \begin{bmatrix} q_{\zeta_i}\Delta + q_{\zeta v_i}\frac{\Delta^3}{3} & q_{\zeta v_i}\frac{\Delta^2}{2} & 0 \\ q_{\zeta v_i}\frac{\Delta^2}{2} & q_{\zeta v_i}\Delta & 0 \\ 0 & 0 & q_{\omega_i}\Delta \end{bmatrix}$$

$$\mathbf{Q}^{d}_{p} = \text{diag}\,\{q_p\Delta\} \quad \text{for} \quad p \in \{g_{ij},\ h_{ij},\ d_{ij}\}\ .$$

Note that the gradient of the state propagation equations with respect the entire state vector is required. Since the equations are linear in the dynamic states $\underline{x}$, the gradient with respect to these states is the estimated state transition matrix. The terms involving gradients with respect to the modal parameters are more complex, but can be computed analytically. It is precisely these gradient terms which provide the necessary coupling between the inputs and outputs (measurements) of the system and the parameters to be identified (the flutter parameters).

The remaining step is to perform the measurement update incorporating the new information in the measurements into the current best estimate of the states. This filtering step is given by the following system of equations:

$$\begin{aligned} \hat{\underline{X}}(k+1|k+1) &= \hat{\underline{X}}(k+1|k) + \mathbf{K}(k+1)\ \underline{\nu}(k+1) \\ \underline{\nu}(k+1) &= \underline{z}(k+1) - \hat{\underline{z}}(k+1|k) \\ \hat{\underline{z}}(k+1|k) &= \hat{\mathbf{H}}^{d}_{x}(k+1|k)\ \hat{\underline{x}}(k+1|k) + \hat{\mathbf{D}}^{d}_{u}(k+1|k)\ \underline{u}(k)\ . \end{aligned} \tag{3-10}$$

Defining:

$$\underline{H}_{\underline{X}} = \{\partial_{\underline{X}}[H_x^d \underline{x} + D_u^d \underline{u}]\}_{\underline{X}} = \hat{\underline{X}}(k+1|k) \ ,$$

the equation for the gain matrix becomes;

$$K(k+1) = P(k+1|k)H_{\underline{X}}^T(H_{\underline{X}}P(k+1|k)H_{\underline{X}}^T + R)^{-1} \ ;$$

and the recursion is complete with the following formula for updating the covariance matrix;

$$P(k+1|k+1) = (I - K(k+1)H_{\underline{X}})\ P(k+1|k) \ .$$

Note, for the 2-mode example, assuming mode 1 has a fixed measurement distribution matrix and assuming that direct-feedthrough is being estimated for both measurements:

$$\underline{H}_{\underline{X}} = \left[\begin{array}{cccc|c|cccccc} h_{11} & h_{12} & h_{13} & h_{14} & & 0 & 0 & 0 & 0 & u_1 & u_2 & 0 & 0 \\ & & & & 0 & & & & & & & & \\ \hat{h}_{21} & \hat{h}_{22} & \hat{h}_{23} & \hat{h}_{24} & & \hat{x}_1 & \hat{x}_2 & \hat{x}_3 & \hat{x}_4 & 0 & 0 & u_1 & u_2 \end{array}\right] \ ,$$

where the indexing on the dynamic states has been simplified in an obvious manner in order to reduce the notational complexity.

Equations 3-9 and 3-10 comprise the EKF formulation of the flutter parameter identification problem. They are recursive in time, incorporate a priori knowledge about expected parameter variations as a function of time, are fully MIMO, and can operate in an asynchronous environment since there is no requirement that the measurement update interval be kept fixed. Computational savings can be and are realized when the data rate is constant however. The only remaining tasks are the determination of the parameters to be identified (model structure determination), the specification of the initial conditions, and the determination of the process noise covariance density matrix.

3.2.1 Specification of Initial Conditions

In general, reasonably accurate initial conditions are available for the flutter frequencies and damping coefficients in the form of predictions from ground vibration tests and finite element modeling programs. These parameters turn out to be important in determining the algorithm performance. Since the estimation problem as formulated is nonlinear, the estimates and the associated estimated variances are random variables which are dependent upon the initial conditions to varying degrees. For parameters appearing linearly in the problem such as the G-matrix and H-matrix parameters, the gradients with respect to these parameters are fixed (i.e. the measured system inputs) rather than functions of parameters being estimated. Since the nonlinearities are manifest in gradients which are themselves functions of the state being estimated, the estimation problem is not overly sensitive to the initial conditions for the G-matrix and D-matrix parameters. Since the H-matrix elements appear bilinearly in the estimation equations, the sensitivity to their initial conditions is somewhat greater than for the G-matrix or D-matrix parameters.

The gradients of the state transition matrix are, however, strong functions of the frequency and damping coefficient for each mode. If, for example, the initial damping coefficient estimate is of the wrong sign, filter divergence can occur. Since the system can initially be assumed to be stable, this problem should rarely arise. A less obvious potential problem is that of initial frequency estimates which are closer to harmonics (multiples) of the actual frequencies present than to the frequencies themselves. The harmonics actually represent local minima in the sense of optimizing or minimizing the data residuals, and the estimated frequencies can get 'trapped' there. For this reason, care must be exercised in the selection of the initial frequency estimates, and more importantly in the allowable range of frequencies. However, with a priori information concerning the locations of the modal frequencies, it should be possible in most circumstances, not only to provide the algorithm acceptable initial frequency estimates, but to set reasonable limits on the frequency variations which circumvent the problem of harmonic minima.

As far as initial conditions for the remaining parameters and the dynamic states are concerned, there is in general no a priori information available whatsoever. In this situation, a zero value is an appropriate initial estimate. A notable exception to this is the case of estimation in the presence of unknown inputs only. If there are no deterministic (known) inputs present, and estimation of frequencies and damping coefficients is being performed with outputs only, initializing the states and all elements of **H** to zero is not advised. Since the gradients of the measurement equations with respect to the dynamic states and **H**-matrix parameters depend linearly on the **H**-matrix parameter and dynamic state estimates respectively, zero gradients will result. The null gradients will in turn yield zero gains, which imply no state correction regardless of the residuals present. The state and parameter estimates will never change from their initial value zero ! This singularity is a consequence of the bilinear and homogenous nature of the nonlinear measurement equations when no exogenous inputs are present. In this situation, initializing an element of **H** for each mode to some non-zero value is necessary.

Once appropriate initial conditions have been determined, the problem of obtaining reasonable estimates of their associated variances remains. As a 'rule-of-thumb', the initial sigma (the square-root of the variance) is set to approximately one-third of the maximum expected deviation from the estimated initial condition; thus, if the initial condition is expected to lie in the interval [0,6] and no further a priori information is available, then 3 is an appropriate initial condition with variance $(\sigma^2) = 1$. As far as the dynamic states are concerned, zero initial conditions with large a priori sigmas are appropriate since a priori state information is rarely available. Fortunately, the sensitivity of the parameter estimates to variations in the initial state estimate error variances is quite small for all reasonable values of the variances. This is due to the model structure in which the outputs are linear combinations of the states and thus contain a great deal of information concerning the 'true' values of the states.

Overly large initial dynamic state estimate error variances can lead to potential problems in the case where **H**-matrix parameters are part of the state being estimated/identified. (The term 'identification' is commonly associated with the 'parameter' states, and 'estimation' with the 'dynamic'

states.) Concentrating for the moment on the measurement equations and their partial derivatives, the bilinear nature of the equations is clear and implies that the covariance propagation will contain terms proportional to the square of the state/parameter estimates times the associated parameter/state estimated variance. Large state estimate error variances, coupled with large H-matrix parameter variances can result in large state and parameter estimate transients, which in turn can cause covariance divergence. In the real-time environment, this is a situation to be avoided.

The initial variances for the flutter mode parameters ($\sigma_\zeta^2, \sigma_\omega^2$) should pose little problem, since as aforementioned, reliable initial conditions will usually be available. However, as is discussed in more detail in the next subsection, large transients in the frequency estimates can result in the filter 'locking-on' to local minima at harmonics of the frequencies present in the outputs. This is a situation which is usually easy to detect, but should be avoided in the real-time environment (presuming of course that there are no harmonics actually present due to nonlinearities for example) since the corrective action will usually involve a 'reset' and subsequent loss of all information processed up to that point! As a 'rule-of-thumb', the initial sigma should be no larger than 25% of the initial frequency estimate.

3.2.2 Identifiability and Parameter Selection

In the previous subsection, the basic guidelines for determination of appropriate initial conditions were discussed. The discussions were based on the assumption that an appropriate model structure had been determined. The determination of an appropriate model structure for a given problem is critical for ensuring reasonable algorithm performance, and basically involves selecting the set of parameters to be identified and the set of parameters to be fixed (and their values). If the estimation problem is over-parameterized (too many parameters), then some of the parameters (or combinations thereof) are not identifiable from the information available (measurements). Filter divergence in a subspace of the parameter space

being searched is highly likely. Since the problem is nonlinear, this will most likely result in total filter divergence, necessitating a reformulation of the problem.

In the example given in equation 3-7, the entire set of parameters was included in the state vector $\underline{X}$. That these are not all simultaneously identifiable from measurements of the inputs and outputs alone is easily verified by noting that in the transfer function description of the equations describing the input-output relationships, only products of g_{ij} and h_{lm} appear. In the SISO case, the residue is the observable/identifiable quantity; the g and h parameters can not be uniquely determined without a further constraint (eg. setting h=1). In the MIMO case, the situation is a bit more complex, but again only the residues are observable. This implies that the dimension of the observable subspace of the G-matrix and H-matrix parameters is less than its maximum value by the number of dynamic states being estimated. Thus, a row of **H** or a column of **G** can be fixed to reasonable values without loss of generality. Zeroing out a row or column is not appropriate since this is tantamount to eliminating an input or output equation from the model while still including the respective measurement. In general, unless contrary a priori information is available, contributions to or from each mode must be accounted for in constraining either columns of **G** or rows of **H**.

Care should be taken in the specification of the g or h parameters when they are not being identified. For example, if a row of **H** is being fixed, the values (assuming 2-modes) [1 0 1 0] seem appropriate a priori, and indeed they are. However, if it happens to be the case that the associated measurement contains predominantly one mode, the states associated with the second mode will be forced to assume exceedingly small values. The appearance of significant amounts of this mode in the other measurement will result in potentially large estimated values for the associated H-matrix parameters. In severe cases, this could potentially lead to numerical problems, especially in the real-time single-precision computing environment. If this problem is suspected, a more judicious constraint should be considered.

3.2.3 Tuning the Process Noise Covariance Density Matrix

The variance densities (q's) of the continuous-time noise processes which are assumed to be 'driving' the true system and are the parameters with which the analyst can 'tune' the filter. Recall the variance densities are directly related to the expected parameter and state variations over time due to unknown sources (such as wind gusts and turbulence or simply a variation due to a change in the operating conditions). A large variance density expresses a large amount of uncertainty in predicting the next state from the previous filtered state estimate. This is precisely the case for example when stores are released from an aircraft and the flutter parameters essentially step to new values. At these times, a large process noise variance density input for the appropriate flutter parameter states is appropriate. During periods when their is little change in operating condition, small values of parameter process noise variance density are warranted. Though subjectively described, these guidelines are a direct means for inclusion of a priori knowledge of the statistics of the underlying physical principles governing the process.

Feedback is provided in the form of the variance of the predicted data residuals. When the tuning parameters are properly set, the theoretical variance of the predicted data residuals (innovations ν) is well approximated by the sample variance. If the innovations are not zero-mean, there is a basic problem inherent in the estimation problem formulation. Thus, the residuals provide a measure for continuously monitoring the performance of the algorithm in real-time processing applications such as the flutter parameter identification problem.

As a practical matter, the variance densities of the dynamic state noise processes are directly related to the amount of turbulence present at the current time and flight condition. If the amount of turbulence can be quantified (measured) in real-time, this information can certainly be used in specifying appropriate values for q_x, the dynamic state process noise variance densities. In the absence of such information, q_x is usually set to yield innovations with the appropriate variance and left unchanged from then on. If the turbulence changes dramatically during the course of an event, q_x can be changed on-line to account for this fact.

As a final note on the tuning of the EKF, the tuning parameters are actually the square-roots of the the associated process noise variance density. By inputting these values in the continuous domain, the effect of non-uniform data rates is taken into account automatically when the integration is performed. The variance densities need not be adjusted as the data interval changes. The units of the continuous density q_x are different, however, involving factors of square-root time (usually seconds), and this should be remembered when setting the values initially. Basically, the product $q_x\sqrt{\Delta}$ should be approximately equal to the expected variation in the associated state from its predicted value based on the simple model employed, taking into account both modeling errors and unknown disturbances such as gusts and turbulence.

3.3 SQUARE-ROOT FILTER FORMULATION AND UNITS NORMALIZATION

To reduce the computational load sufficiently, the majority of the computations are performed in single-precision. While reducing the time required to perform the basic operations of multiply and add, this has the undesirable effect of reducing the dynamic range of allowed estimate values and more importantly their variances. Since the frequencies involved in the flutter parameter identification problem can be on the order of hundreds of radians/second and the damping coefficients on the order of a few percent ($\zeta \approx 0.01$), and since the data rates required to adequately sample these signals can approach 1 KHz (0.001 second sampling interval), the underlying characteristic number associated with these problems can be as large as 10^8 (recall variances are squares of associated sigmas). Linear combination with numbers on the order of unity places the accuracy of a single computation at the single-precision threshold on most machines. Thousands of such operations can result in unacceptable accumulated roundoff errors. Such errors can lead to negative definite covariance matrix estimates (a situation to be avoided at all costs) and filter numerical divergence. There are two methods for alleviating these problems, both of which were implemented in the flutter parameter identification algorithm -- state normalization and square-root covariance propagation.

3.3.1 State and Parameter Normalization

Units normalization redefines the units of the parameters and states being estimated so that both the estimates and their variances are on the order of one. For example, measuring frequency in units of 100's of rads/sec and damping coefficient in percent yields estimates on the order of unity for both parameters in the case of the DAST data (see Section 6) where the frequencies were on the order of 20 Hz and the damping coefficients were only a few percent. On the other hand, for the F-16 data, the frequencies were on the order of rads/sec requiring no normalization and the damping coefficient normalization to units of 10% was appropriate for the estimates obtained.

The normalization of the dynamic states is particularly valuable in the high frequency cases where the 'velocity' states are larger than the associated 'position' states by a factor equal to the natural frequency (in rads/sec) of the mode. Since the covariance calculations involve squares of these factors, potential numerical problems are circumvented via this normalization. Keeping the dynamic state estimates small also results in reasonable H-matrix element estimates, values which would otherwise be exceedingly small (on the order of 1/(velocity state estimate))!

The G-matrix parameter estimates are normalized in one of two ways. Direct units normalization can be performed as with the H-matrix elements or the dynamic states. The need for normalization is a consequence of the continuous model form chosen, wherein the input-output transfer function can be seen (for a SISO system) to be of the form:

$$\frac{z(s)}{u(s)} = \frac{hg}{s^2 + 2\zeta\omega s + \omega^2}$$

For values of h on the order of unity, low frequency components of the input will be suppressed by a factor of $1/\omega^2$ unless g is on the order of ω^2. Normalization of g similar to that used for the frequencies will mitigate the problem.

An alternate approach implemented in the algorithm is to redefine the g_1-parameter to be $\omega g_1'$ and the g_2-parameter to be $\omega^2 g_2'$ and to estimate the

control distribution vector $\underline{g}'$ instead of $\underline{g}$. This increases the complexity of the gradient expressions somewhat, but has the desireable feature of being 'auto-normalizing' in the sense that the normalization is a function of the frequency estimates themselves. This normalization is a reparameterization of the model structure. This reparameterization is especially useful with regards to the process noise distribution matrix. In the low frequency region (such as the case for the F-16 data), the amount of continuous-time process noise variance density required to properly tune the algorithm becomes a strong function of the frequency unless auto-normalization is performed. With auto-normalization engaged, tuning of the algorithm can be performed using values for q_x which are essentially independent of the frequency of the mode being estimated.

There is another useful feature aside from increased numerical stability which is derived from proper units normalization. If the normalization is performed such that the expected values of each of the parameter and state estimates are on the order of unity, detecting either input errors or filter divergence becomes an easy task. Estimates or estimated variances differing significantly from unity are indications of potential problems and/or instabilities.

3.3.2 Square-Root Formulation

The second technique for increasing numerical stability involves propagation of the square-root of the covariances associated with the state estimates, rather than the covariances themselves. This has several desireable consequences in real-time applications such as the flutter parameter identification problem. Since the object being propagated is a square-root of a covariance, in order to form the covariance (not that this is ever required) the appropriate matrix must be 'squared'. The resulting covariance is guaranteed to be positive semi-definite regardless of the nature of the original matrix! Furthermore, the characteristic number of the problem associated with the covariance propagation is essentially halved since the square-root operation halves the exponent in the dynamic range calculation. Thus, for a fixed wordlength, in this case single-precision,

numerical stability can be ensured over essentially twice the dynamic range of the standard EKF algorithm. These features make square-root filtering a desirable concept in real-time applications. As a historical note, such algorithms are used in the space shuttle guidance and control systems and were used in the Apollo program as well with a great degree of success.

The square-root filtering equations can be derived in many ways, some more algorithmic in nature than others. These derivations are left to the references except to indicate that the key concept is that for any positive definite matrix **B** (covariance matrices are in this class), there exists another matrix **A** such that:

$$B = A^T A \ ;$$

where "T" is used to denote matrix transposition. Furthermore, if the matrix **A** is constrained to be upper (or lower) triangular, the factorization is unique. In any case, the matrix **A** is called a 'square-root of **B**'. The idea then is, given a set of equations describing the evolution of the matrix **B**, find a set of equations for the evolution of **A** such that the above relationship between **A** and **B** is satisfied at each step in the evolution.

The algorithm is initialized by computing the square-roots of the initial covariance matrices in the problem. Triangular square-roots are chosen since they are unique and there are fast algorithms for extraction of triangular factors in the literature. For matrices whose structure is initially diagonal and whose entries are initially specified in terms of sigmas rather than variances, this is easily accomplished. For covariance matrices with off-diagonal non-zero entries, the Cholesky factorization algorithm can be used to obtain the appropriate triangular factors.

Once the square-root EKF algorithm has been initialized, the following equations are used to recursively update the estimates and covariances. In order to simplify the notation somewhat, define:

$$P_F = P(k|k) \ ; \qquad \underline{X}_F = \hat{\underline{X}}(k|k);$$

$$P_P = P(k+1|k); \qquad \underline{X}_P = \hat{\underline{X}}(k+1|k);$$

In order to perform the time-update or prediction, form:

$$[(\partial_{\underline{X}}\phi)P_F^{1/2} \quad Q_{\underline{X}}^{1/2}] \ .$$

At this point the objective is to find a set of orthonormal transformations which will lower triangularize the rectangular matrix. The resulting equivalent matrix will be the square-root of the predicted covariance matrix! The technique used to perform the triangularization is a numerically stable procedure known as the Householder transformation. Basically, the transformation zeroes out all elements in the first column of the subject matrix by performing a multi-dimensional rotation about a suitable axis. Performing this rotation on sequentially smaller principal submatrices of the original matrix results in the triangularization desired. Thus, the procedure uses Householder transformations to lower triangularize the augmented matrix yielding:

$$[\ P_P^{1/2} \quad 0\ ] \ .$$

The states are updated as follows:

$$\hat{\underline{X}}_P = \hat{F}^d_x \hat{\underline{X}}_F + \hat{G}^d_u \underline{u}$$

$$\hat{\underline{\theta}}_P = \begin{bmatrix} F^d_{\zeta\omega} & 0 \\ 0 & I \end{bmatrix} \hat{\underline{\theta}}_F$$

The bulk of the computation effort in the prediction is in performing the Householder triangularization of the augmented matrix. The number of operations required is proportional to n^3, where n is the dimension of the state vector.

The equations for performing the measurement update are given below. Though more complex in structure, the philosophy is the same. Form an augmented matrix and triangularize it. The resulting triangular matrix contains the updated quantities required to perform the next prediction.

To perform the measurement update, form:

$$\begin{bmatrix} R^{1/2} & H_{\underline{X}} P_P^{1/2} \\ 0 & P_P^{1/2} \end{bmatrix}$$

and lower triangularize with Householder transformations to obtain:

$$\begin{bmatrix} R_\varepsilon^{1/2} & 0 \\ \bar{\mathbf{K}} & P_F^{1/2} \end{bmatrix} .$$

Invert $R_\varepsilon^{1/2}$ and find $\mathbf{K}$ as follows:

$$\mathbf{K} = \bar{\mathbf{K}} R_\varepsilon^{-T/2} .$$

Finally, update the states:

$$\hat{\underline{X}}_F = \hat{\underline{X}}_P + K\underline{\nu}$$

$$\underline{\nu} = \underline{z} - \hat{\underline{z}}_P .$$

As is the case with the time-update, the majority of the computational effort is in performing the Householder triangularization.

The above equations describe the recursion for the EKF flutter parameter algorithm in square-root form. These equations have been implemented in the MOPID program which performed the analysis presented in the subsequent sections. Advantage was taken of the structure of the flutter parameter identification problem to minimize the number of computations at each iteration resulting in special purpose routines for performing the Householder transformations and the various required matrix multiplications. This was done in an effort to increase the maximum throughput rate which could be achieved. In the next section, algorithm extensions to address the 'real data' processing issues are discussed along with estimation of appropriate performance parameters for real-time monitoring. Sections 5 and 6 present results from simulated and actual flight data processing, and Section 7 discusses the program structure and gives some computation counts.

SECTION 4
ALGORITHM EXTENSIONS

In the previous section, the details of the EKF algorithm applied to the flutter parameter identification problem were presented. Several issues must still be addressed in the real-time aircraft flutter parameter identification environment, however. The measurements used by the algorithm include (but are not limited to) accelerometer (specific force meter) outputs and control surface (aileron) deflection commands or actual positions. These data contain information about center-of-mass motion and non-zero set point control surface deflections as well as the 'high frequency' flutter information. If not properly taken into account, the presence of the center-of-mass motion will result in erroneous (biased) parameter estimates.

A second problem faced with 'real' data is that of data 'outliers'. In the real-time environment, valid data may not be present at each sample time due to hardware/software malfunction. Detection of these data outliers is necessary since they can lead to biased parameter estimates and possible filter divergence.

Another issue, somewhat unrelated to the first two, is that of extending the algorithm to provide sufficient information for presentation to an operator in a real-time environment. The information should be accurate in the minimum variance unbiased sense, and should be timely. Indicating that the closed-loop modes of the system went unstable several seconds ago is not as useful as an indication that the system is approaching instability and is likely to be unstable within the next few seconds. The extensions of the EKF algorithm to address these issues are discussed in this section.

4.1 DATA CONDITIONING

In real-time flutter parameter monitoring, the inputs to the EKF algorithm are the outputs of various measurement devices onboard the test vehicle. In most flutter tests, the system outputs will be measurements

from accelerometers mounted on the aeroelastic surfaces, usually wing-tip accelerometers. The system inputs can be either measurements of actual control surface deflections or measurements of 'commanded' control surface deflections. These system inputs and outputs are the measurements used in the EKF algorithm and will in general contain more than just the high frequency information of interest.

In the real-time environment, the accelerometers will be sensing not only the acceleration of the wing due to the excited structural modes, but the center-of-mass motion of the vehicle as well. To minimize these effects, the test vehicles are usually flown straight-and-level in possibly horizontally accelerated flight during the flutter parameter identification phase of the tests. Thus, center-of-mass (and potential body-rate and angular acceleration) effects are expected to be small, resulting primarily from disturbances filtered by the vehicle's stability augmentation system (SAS).

There are several methods for taking into account the presence of low-frequency information unrelated to the flutter parameters of interest. A direct open-loop approach is to measure the center-of-mass and angular accelerations with suitable measurement devices, and subtract out the appropriate combinations of these from the FSS accelerometers. This method, though conceptually straightforward, is susceptible to potentially disasterous errors due to miscalibration and unmodeled parameter variations.

A second approach is to augment the state vector with a 'bias' parameter for each measurement. The bias parameter could have the same trivial dynamical model as the other data parameters, with process noise used to allow for low-frequency variations over time. If the EKF algorithm is viewed as a 'data filter', inclusion of such states is loosely equivalent to inserting a high-pass filter in the forward path from the data inputs to the parameter/state estimate outputs. The locations of the poles and zeroes of the filter are determined by the solution of a time-varying Ricatti equation involving the stochastic parameters in the problem. Thus, the pole locations are not necessarily time-invariant. Note that this filter removes not only the center-of-mass motion, but the accelerometer zero-g level as well, since from an input-output standpoint, there is no difference between

the two. However, the augmentation of the state vector results in increased computational requirements.

In order the circumvent the need for increasing the dimension of the state vector, a suboptimal but computationally less expensive alternative is employed. A fixed-parameter high-pass filter is used to 'condition' the input data before inclusion in the EKF algorithm. Though this is a 'suboptimal' approach to low-frequency information removal, it satisfies the more immediate goal of computational efficiency while not sacrificing much in the way of optimality. It is suboptimal in the sense that the low-frequency information could be 'optimally' estimated by incorporating a complete six degree-of-freedom estimation algorithm for estimating the vehicle's center-of-mass and rotational motion. However, since such an algorithm is computationally expensive, a simple high-pass filter sufficies for most situations. The only requirement is that the filter pole be appreciably below the lowest frequency of interest.

The high-pass filter employed is a bilinear transform equivalent of a simple first-order lead network given by the following continuous-time transfer function description:

$$F_{HP}(s) = \frac{s}{s + \alpha} \ .$$

The location of the pole α is selectible by the operator, appropriate values being on the order of 1/10 the lowest flutter frequency expected. Formulation in the continuous-time domain allows for non-uniform data rates which is consistent with the requirement for asynchronous operation of the overall real-time algorithm. Using trapezoidal integration techniques for transforming to the discrete-time domain leads to the use of the bilinear transform for conversion from the s-plane to the z-plane. The bilinear transform of $F_{HP}(s)$ gives:

$$F_{HP}(z) = \frac{1}{1 + \Delta\alpha} \left[1 - \frac{\Delta\alpha/(1 + \Delta\alpha/2)}{z - \frac{(1 - \Delta\alpha/2)}{(1 + \Delta\alpha/2)}} \right] ;$$

where $\Delta = t_{k+1} - t_k$. Bode plots of the gain and phase of the discrete transfer function for a pole at 1 Hz are shown in Figure 4-1. The sampling rate

was assumed to be 250 Hz. For frequencies above 10 Hz, the high-pass filter introduces no significant distortion in gain or phase.

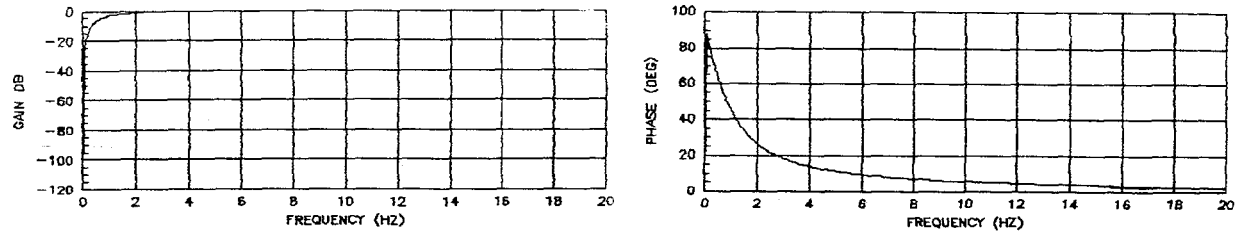


Figure 4-1: Bias Rejection Filter Bode Plots for Pole at 1 Hz.

A key point in the implementation of the bias rejection filter is that both the system inputs and outputs are filtered identically, whether it is required for bias rejection or not. In the closed-loop identification problem where the inputs to the system are chosen to be the control surface excitation commands, the inputs may or may not require bias filtering for removal of any DC-components, however the bias filtering is performed nonetheless in order that the system inputs and outputs (which almost surely require high-pass filtering) experience the same phase-shift at the flutter frequencies, regardless of how small the phase shift is in magnitude.

4.2 DATA OUTLIER REJECTION

In the real-time environment, the possibility of a malfunction of the data collection system is always present and must be taken into account in the design of any robust real-time estimation algorithm. When data which are not accurate measurements of the system inputs or outputs (i.e. bad data points) are input to the EKF algorithm, biased estimates and filter divergence can result. To 'optimally' address this issue of outlier detection and rejection requires the theory of multiple hypothesis testing. In a real-time environment, this approach is not computationally feasible. A sub-optimal computationally feasible approach can be taken however, to impart some robustness to the EKF algorithm employed.

The outlier detection algorithm involves testing the predicted data residuals against their theoretical variance. If a particular data residual or innovation exceeds the square-root of its theoretically predicted variance by an operator selectible factor (usually on the order of 5-10), the data value associated with that residual is declared an outlier. The use of the predicted data residuals restricts this algorithm to detection of output outliers only. Rejection of output outliers is an easy task since the algorithm is designed to run asynchronously. The bad data point is simply deleted from the data vector at the current time only, and the algorithm proceeds as if no output from that particular device were present at the current time. If the outlier were the only measurement at that time, a data drop-out of one sample time results. Prediction of data values is required only for the innovations calculation. The predicted data value is not inserted in place of the outlier, since the new residual would be identically zero resulting in zero state adjustment, and processing of the 'predicted' measurement would decrease the covariance which is not an appropriate action given the decision that the datum was not a valid system output.

The previous discussion was directed primarily towards handling system output measurement outliers. The situation is a bit more difficult with regards to system input measurement anomalies. Since the system inputs are not functions of the states being estimated, there is no analog to the predicted output residuals and their theoretical variance. Furthermore, in order to predict the state at the next measurement time, a measurement of the input to the system at the current time is required since the inputs are not states being estimated. In the absence of a valid input value, assuming there were an efficient way for detecting input outliers, an input value must be assumed. The implementation currently assumes a 'sample-and-hold' strategy in the presence of invalid input data. Linear extrapolation could be used but is less stable in the presence of extended input drop-outs. Furthermore, the validity of the input data is assumed to have been ascertained prior to inclusion in the algorithm by the telemetry processing system. Upon encountering an invalid input data flag, the algorithm uses the previous system input as the current value.

This strategy could be made more complex. It would certainly seem appropriate to increase the process noise variance density on the oscillator

states when invalid inputs are detected. Sliding window fixed-order polynomial predictors could be used to estimate the 'missing' input value, at the expense of an increase in computational load and potential processing delays. A third easily implementable and potentially more stable solution would be to ignore all the data at that time and wait for the next sample time. However, little gain is expected since input data outliers, especially involving commanded inputs, are infrequent.

4.3 REAL-TIME INSTABILITY PREDICTION

The primary objective of any algorithm for real-time flutter monitoring is to provide accurate and timely information concerning the stability of the system being tested. In the locally linearized model of flutter dynamics employed, the damping coefficient estimates provide a direct measure of the local stability of the system at the current operating conditions. For the purposes of this discussion, it is assumed that the system being identified is the one in operation; i.e. if the flutter suppression system (FSS) is ON, then it is assumed that the closed-loop system is being identified. In this configuration, the exogenous input commands to the ailerons are the EKF algorithm system inputs. If the FSS is OFF, it is assumed that the open-loop system is being identified and the appropriate EKF algorithm inputs are the actual aileron positions (deflections). Note that this need not be the case; however, if these conditions are not satisfied, then the system being identified does not accurately reflect the current input-output stability which is desired.

A commonly used measure of the stability of time-invariant linear systems is the 'phase margin'. For single-input single-output (SISO) systems, this concept is well-defined and is the difference between the phase of the system transfer function and 180 degrees at the unity gain crossover point (i.e. the frequency ω such that $|F(\omega)| = 1$). If the magnitude of the phase of the transfer function is greater than 180 degrees at the unity gain frequency, then the phase margin is negative and the system is unstable. For multi-input multi-output (MIMO) systems, however, the concept of phase margin is much more complex, and consequently is not a desirable performance measure for real-time monitoring.

What characterizes MIMO instability most directly is the crossing of one of the system's natural frequencies into the right half-plane. Thus, estimation and therefore prediction of pole locations as a function of time (or a possibly more appropriate independent variable such as dynamic pressure or Mach number) is certainly an appropriate stability performance measure for real-time monitoring. Furthermore, the capability to predict future pole locations is certainly desireable since this opens the possibility of automated instability detection/warning systems which relieves some of the burden of real-time decision making from the operator.

As discussed in detail in Section 3, the capability to predict future values of the damping coefficient of each mode being estimated was facilitated by the addition of a damping coefficient velocity state for each mode. Making the assumption that the model for the damping coefficient as a function of the independent variable is locally linear, prediction of future values of the damping coefficient given all the past information is easily performed. Letting k be the current time and T be the desired prediction interval, the best T-second ahead predictor is given by:

$$\hat{\zeta}(k+T) \approx \hat{\zeta}(k) + T\hat{\zeta}_v(k) \; . \qquad (4\text{-}1)$$

The variance of this estimate is also easily calculable:

$$\hat{\sigma}^2_{\zeta(k+T)} = \begin{bmatrix} 1 & T \end{bmatrix} P_{\zeta\zeta_v} \begin{bmatrix} 1 & T \end{bmatrix}^T;$$

where $P_{\zeta\zeta_v}$ is the covariance of the estimate error of $[\, \zeta \quad \zeta_v \,]^T$. Clearly as the prediction interval increases, the variance increases as well. For large intervals, the variance becomes quadratic in time (or other independent variable) eventually resulting in statistically meaningless estimates. If the model is appropriate, predictions prior to this time are significant estimates of the systems future stability.

A possibly more valuable measure of the systems future stability is an estimate of the time-to-instability (TTI). Again, time can be replaced by a different independent variable such as dynamic pressure if desired. Though a slightly more complex calculation than the simple prediction given in equation 4-1, its value lies in its potential to provide an increasingly

more accurate estimate of a critical system parameter (essentially the flutter boundary) as instability is approached. The estimate of TTI is given by:

$$\hat{TTI}(k) = -\hat{\zeta}(k) / \hat{\zeta}_v(k) . \qquad (4-2)$$

Clearly this is a nonlinear estimator and its implementation requires certain limits to be imposed.

Since there is no lower bound to the magnitude of $\hat{\zeta}_v$, the estimated TTI can become infinitely large. This corresponds to a system estimated to be invariant as far as damping coefficient is concerned. Such systems pose no potential stability problems (unless of course they happen to be already unstable). Placing a reasonable upper bound on estimates of TTI is therefore certainly warranted. For example, in the results discussed in Sections 5 and 6, a bound of 10 seconds was placed on estimated values of TTI. Values in excess of this bound were simply ignored (practically they were set to zero to avoid plot scaling problems). Secondly, the estimate of TTI becomes negative whenever $\hat{\zeta}(k)$ is greater than zero and $\hat{\zeta}_v(k)$ becomes positive. This corresponds to the situation where the estimated system is becoming increasingly more stable and certainly poses no instability problems in the near future. Note that the estimate of TTI also becomes negative when both $\hat{\zeta}(k)$ and $\hat{\zeta}_v(k)$ are negative. In this case, there are certainly more serious problems facing the operator than a negative TTI estimate. Thus, zero is a logical lower bound to the estimate of TTI.

Finally, for the estimate to be meaningful, its estimated sigma should be small. The variance of the TTI estimate is calculated as follows:

$$\hat{\sigma}^2_{TTI} = \left[-\frac{1}{\hat{\zeta}_v} \quad \frac{\hat{\zeta}}{\hat{\zeta}_v^2} \right] P_{\zeta\zeta_v} \left[-\frac{1}{\hat{\zeta}_v} \quad \frac{\hat{\zeta}}{\hat{\zeta}_v^2} \right]^T ;$$

and all estimated quantities are assumed to be at time k. It should be kept in mind that though not explicity indicated as such, the covariance $P_{\zeta\zeta_v}$ is also an estimated quantity since it is a function of other estimates in the problem which in turn are a function of the measurements.

As eluded to several times in the previous discussions, using time as the independent variable in the dynamical model for the damping coefficient and its derivative may not be as appropriate as another choice. For example, extensive experimental evidence suggests that most aerodynamic parameters (stability coefficients in particular) are 'locally' linear functions of dynamic pressure and/or Mach number. Since the damping coefficient of a particular mode depends strongly on these parameters, a more appropriate independent variable would seem to be dynamic pressure or Mach number. For constant altitude flight tests, dynamic pressure is probably to be preferred over Mach number since it is roughly quadratic in airspeed whereas for constant atmospheric density, Mach number is linear in airspeed, and experimental evidence indicates that the variation is more nearly quadratic.

If dynamic pressure were monitored, an appropriate extension of the algorithm to incorporate the damping coefficient dependence upon this parameter would be to include the dynamic pressure as another measurement. The damping coefficient dynamical model would remain unchanged. A measurement model of the form:

$$\rho(k) = h_\rho(k)\, \zeta(k) + b_\rho(k) ,$$

would effectively create the desired dependence of damping coefficient on dynamic pressure. The vector of parameters being identified would then be augmented with the measurement parameters h_ρ and b_ρ and the following dynamical models assumed for the parameters:

$$h_\rho(k+1) = h_\rho(k) + w^d_{h_\rho}(k) .$$

$$b_\rho(k+1) = b_\rho(k) + w^d_{b_\rho}(k) .$$

where ρ is dynamic pressure output. The required partials are entirely analogous to those for the accelerometer measurements.

Note that a measurement of dynamic pressure rate could also be included in the algorithm using exactly the same approach. The measurement would be proportional to ζ_v and would provide exceedingly valuable information on the rate of change of ζ. The variances of the parameters used to

predict future system stability would be greatly reduced. The generalization of these ideas to include other measurement device outputs with strong functional dependence on the frequency parameter related states is obvious.

The interesting feature of this approach is the implied inversion of the functional relationships as currently accepted, i.e. viewing dynamic pressure as a function of damping coefficient rather than damping coefficient as a function of dynamic pressure. Observability issues need to be addressed however, since in this approach there would be, at least conceptually, several measurement equations for each output. Further dynamic state vector augmentation might be required at the expense of increase computational load. Investigation into alternate measurement model forms might be appropriate to avoid this potential increase in dimensionality.

As a final note, with time as the independent variable and a constant acceleration profile (such as was the case for the last minute or so of the actual DAST test), the results of Sections 5 and 6 can be viewed in light of the previous discussion as having a measurement of Mach number (or airspeed) and assuming a linear dependence of the measurement on the damping coefficient. The use of the quadratic model for the damping coefficient in the simulation (cf. Section 5) was an attempt to approximate a linear dynamic pressure dependence in the presence of a linear speed profile.

SECTION 5
SIMULATED TEST CASE RESULTS

This section presents the results of two simulated test cases. The two cases were used to illustrate the performance of the algorithm under conditions similar to those encountered in both of the 'real' data cases analyzed (DAST and F-16 data, cf. Section 6). The value of the simulations lies not only in proof-of-concept and its implementation, but also in providing some insight into such issues as input design for improved parameter identifiability. These issues are discussed in this section and in the next section as well.

The first test case to be discussed is the simulated DAST test. The test was designed to contain many of the aspects present in the real DAST test flight. The input excitation is very similar, and the presence of lightly damped closely spaced modes, one of which eventually becomes unstable, is an attempt to model the last twenty seconds of the (third) DAST test flight. The second test case was designed to simulate the actual F-16 data which were analyzed. The basic objective was to illustrate the performance of the algorithm when exogenous inputs are not present. The only forcing functions are the random disturbances encountered in flight. Since very little information concerning the actual conditions of the real F-16 test flight was available, only a simple test case with fixed modes was used. The results of both tests verify the correctness of the algorithm implementation and to some extent demonstrate proof-of-concept.

5.1 SIMULATED DAST TEST CASE

The objective of the simulated DAST test case was to investigate the performance of the EKF algorithm for flutter parameter identification under conditions similar to those encountered in the real DAST test flight. To this end, a 2-input, 2-mode, 1-output system was simulated using the simulated data generation capability of the program. The two modes were closely

spaced and time-varying in both frequency and damping, with one of the modes becoming unstable approximately one second before the end of the data interval. The exogenous (or deterministic) inputs consisted of a sequence of 'pulses' and chirps, or frequency sweeps. The 'pulses' consisted of a single cycle of a sinewave whose frequency was the average of the frequencies of the simulated modes at the time of the pulse (on the order of 20 Hz). The chirps were logarithmic frequency sweeps from 10 Hz to 40 Hz, with linear tapering at both ends of the sweep interval. Neglecting the tapering, the functional form of the input is:

$$u(t) = A \sin[\frac{\omega_0 \omega_1 T}{\omega_1 - \omega_0} \ln(\frac{\omega_1 T}{\omega_1 - \omega_0} - t)] \ ;$$

where ω_0 and ω_1 are the starting and stopping frequencies (10 and 40 Hz respectively), T is the sweep duration and A is the amplitude (set to 1 second and 1 degree respectively).

Two inputs were applied to the two-mode time-varying linear system. The first input (u_1) consisted of a sequence of pulses and chirps, each occurring in 'pairs'. The sign of the input waveform was changed at the onset of each functional form. This resulted in a sequence of 'plus then minus' pulses followed by 'plus then minus chirps'. As indicated in Figure 5-1, the second input (u_2) was identical to the first except no sign switching was performed. Thus, the two inputs provided 'symmetric' and 'asymmetric' excitation for the two mode system. The input distribution vectors (columns of the G-matrix) for each input were chosen such that their sum and difference placed most of the symmetric and asymmetric input power into separate modes. The 'isolation' was chosen to be approximately 10dB.

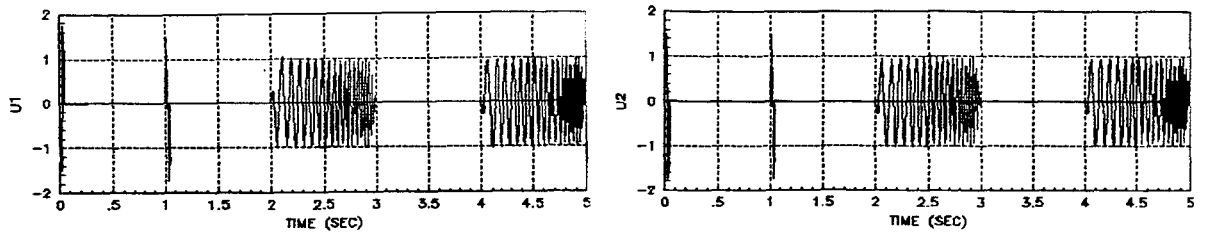


Figure 5-1: Simulated DAST Input Waveforms

Random disturbance inputs were also modeled as 100 Hz low-pass filtered white Gaussian noise processes. Two independent processes were used

with different input distribution vectors to model possible differences in disturbability in the symmetric and asymmetric modes. The variance density of each process was chosen such that the output SNR was approximately 20dB (an amplitude ratio of 10:1) where SNR is defined momentarily as the ratio of output power due to deterministic as well as stochastic inputs to that due to stochastic inputs alone.

The output of the system was chosen to be a unity weighted sum of the 'position' states of the two modes with some direct feedthrough, i.e.

$$z_1 = H\underline{x} + D\underline{u} + v \ ;$$

where H_0 was set to [1 0 1 0], and where v was a normal (0,0.0004) white Gaussian noise process used to model measurement noise in the data acquisition system. The direct feedthrough term was incorporated to model control surface feedforward components in accelerometer outputs. Figure 5-2 shows the two inputs and the output of the simulation over the entire 20 second interval simulated. Other than the compressed time scale and decreased time interval between deterministic inputs, comparison with Figure 6-2 indicates the similarity of the simulated data to the actual DAST flight data.

Technically there are no units affixed to the simulation problem since it is a mere mathematical model. However, by adhering as closely as possible to the conditions of the actual DAST test flight, the inputs can be viewed as aileron commands in degrees, and the output can be viewed as that of an accelerometer in g's. The measurement noise sigma 0.02 was chosen as twice the value for RMS output noise (in g's) given in the specifications for the actual DAST accelerometers.

Estimation Problem Formulation

In order to minimize the nonlinearities in the estimation problem, estimation of the elements of the G-matrix was chosen in favor of estimating those in H. Thus, in addition to the six modal parameter states, eight G-matrix elements and two direct feedthrough terms were identified. The initial conditions for the G and D elements were set to zero with sigma 1 for the G elements and 0.5 for the elements of D. The initial frequencies were chosen to be 15 Hz and 30 Hz with a sigma of 5 Hz for each. The true

initial frequencies were 20 Hz and 25 Hz. The initial damping coefficients (in percent) were set to 10% with sigma 5% for both modes. The true initial values were 4% and 2% respectively.

The measurement noise sigma for the simulated data was chosen to be 0.02 as discussed previously, and the sigma used in the estimation algorithm was set to 0.02 as well. The square roots of the process noise variance densities (q's) for the four dynamic states were chosen to be 1 in units of the states per square-root time. Frequency normalized process noise was not used for this test though it easily could have been. This would have resulted in values of 0.01 being appropriate instead of unity. Values of 0.001 and 0.0001 for the **G** and **D** parameter q's were used even though the true parameter values were not time-varying. These values were used as being representative of values for 'real' data cases where the parameters are expected to be slowly time-varying. Q's of 0.2 were used for the frequency states and 0.0004 was used for the damping coefficient velocity states.

The integration of the simulation was performed at 250 Hz, a decade above the highest frequency mode. Trapezoidal integration was used to more closely approximate the output of a continuous dynamic system. The estimation data rate was chosen to be 250 Hz as well. No data drop-outs were simulated and no spurious data points were added.

Results

Figures 5-2 through 5-6 present the results of the simulated DAST flutter parameter estimation. The predicted data residuals and their theoretical sigma are shown at the bottom of Figure 5-2. Their randomness and concentration within the theoretical one-sigma values indicate that the filter has been properly 'tuned' (i.e., the process noise variance density has been set appropriately). The 'impulse' in the theoretical one-sigma value at one second is due to the nonlinear nature of the estimation of the elements in **G**. Prior to one second, the only input to the system was 'asymmetric'. Thus, no information was present concerning the 'symmetric' input distribution matrix terms and their variances remained near their initial values. When the associated partials (i.e. symmetric input values) became large at one second, the residual variance increased (cf. **HPH'** increased).

Figure 5-3 presents the time histories of the modal parameter estimates, their estimated sigmas, and the true estimate error for each parameter. On each parameter estimate plot, the true time history is also plotted. These are the 'straight lines' on the frequency and damping coefficient velocity plots and the 'parabolas' on the damping coefficient plots. On the plots of the estimate error, the plus and minus estimated one-sigma values are given as well. With the possible exception of the estimate error for ω_2, the errors are compatible with the estimated one-sigma values. When sufficient input excitation is present, the estimate error for the second modal frequency becomes quite small; but when no input is forcing the system, the error increases due to the linearly decreasing nature of the underlying true value coupled with the constant frequency dynamical model used to predict the frequency.

Figure 5-4 is an s-plane plot of the time histories of the estimated pole locations. The parabolic curves are the time histories of the true modal locations. After the initial transients die down, the algorithm quite successfully tracks the poles even into the right half-plane. The erratic nature of the estimated time histories is a consequence of the measurement and process noise disturbances. By increasing the deterministic input power (increased amplitude and/or increased waveform duration and repetition frequency), the variance (both theoretical and actual) of the estimated pole locations can be decreased.

As far as real time flutter parameter monitoring is concerned, Figure 5-5 presents the most relevant estimates. As discussed in Section 4, the reason for estimating damping coefficient velocity was to allow for future values of damping coefficient to be predicted. These calculations are embodied in the estimates of time-to-instability (TTI) (and its variance) and the '5-second ahead' prediction of damping coefficient (and its variance) shown. The straight lines in the TTI plots are the actual values of TTI, and thus have slope -1 and intersect the time axis at 19.1 seconds (the time at which the actual damping coefficient for the first mode becomes negative). The estimates of TTI were limited to a value of 10 seconds. Larger estimates were arbitrarily set to zero in order to avoid plot scaling problems resulting from exceedingly large TTI estimates. The parabola in the 5-second prediction plot at the bottom of the figure is the actual value of the associated damping coefficient displaced backward in time by 5 seconds

(thus it terminates at 15 seconds). The plus and minus one-sigma estimates are indicated as well, and the predictions are seen to be quite good.

Finally, Figure 5-6 gives the estimates for the **G**-matrix and D-matrix parameters and the plus and minus one-sigma estimates. The estimate error for the parameters is not plotted since the true parameter values were constants. With reference to the true values given below, it is easily seen that the estimates are all within their theoretical one-sigma values.

$$\mathbf{G}_0 = \begin{bmatrix} -0.4 & 0.4 & 0.2 & 0.2 \\ 0.2 & 0.3 & -0.3 & -0.3 \end{bmatrix}^T ;$$

$$\mathbf{D}_0 = \begin{bmatrix} -0.3 & 0.0 \end{bmatrix} .$$

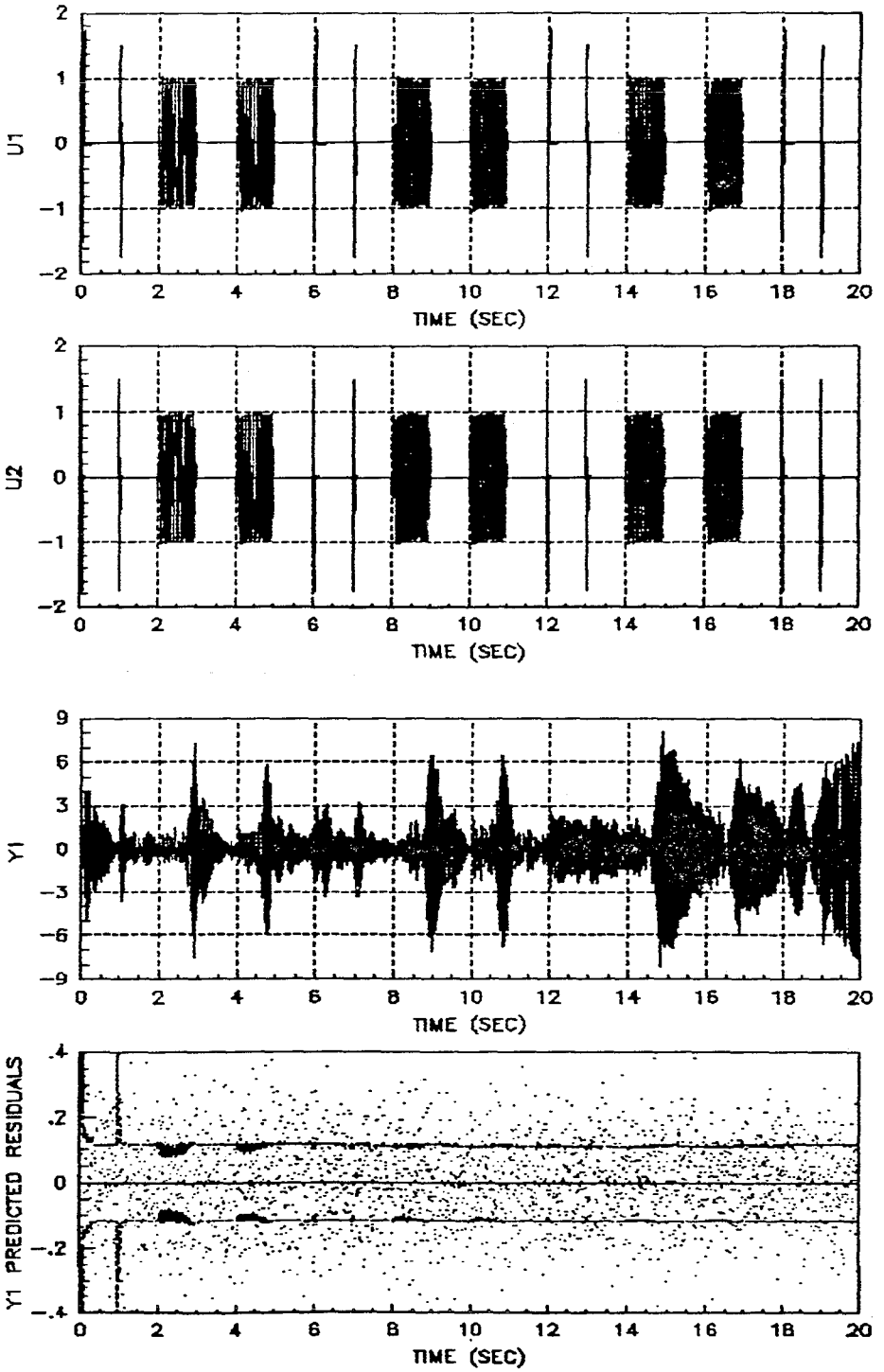


Figure 5-2: Simulated DAST Inputs, Outputs, and Predicted Data Residuals

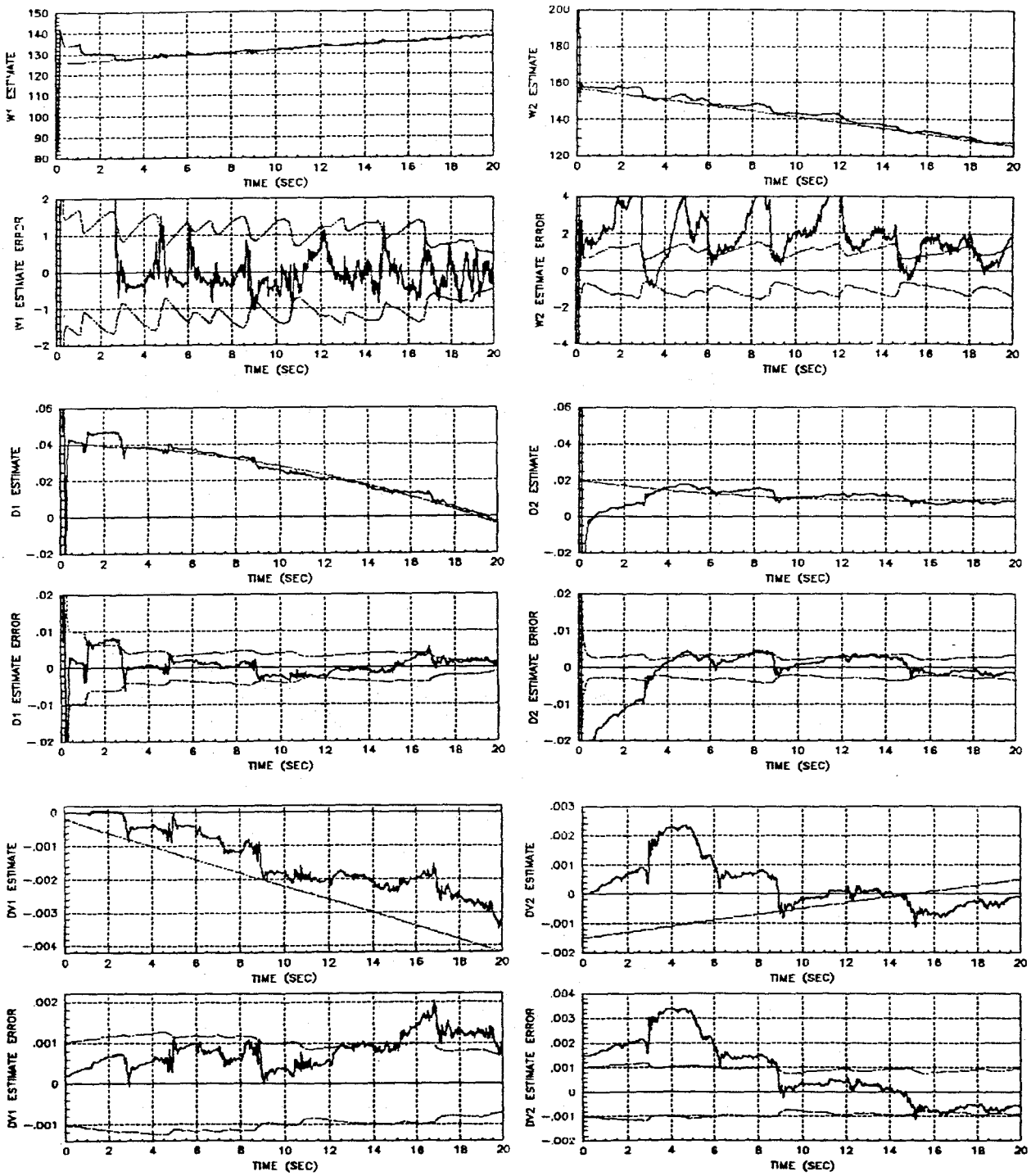


Figure 5-3: Simulated DAST Modal Parameter Estimates, Estimated Sigmas and Estimate Errors

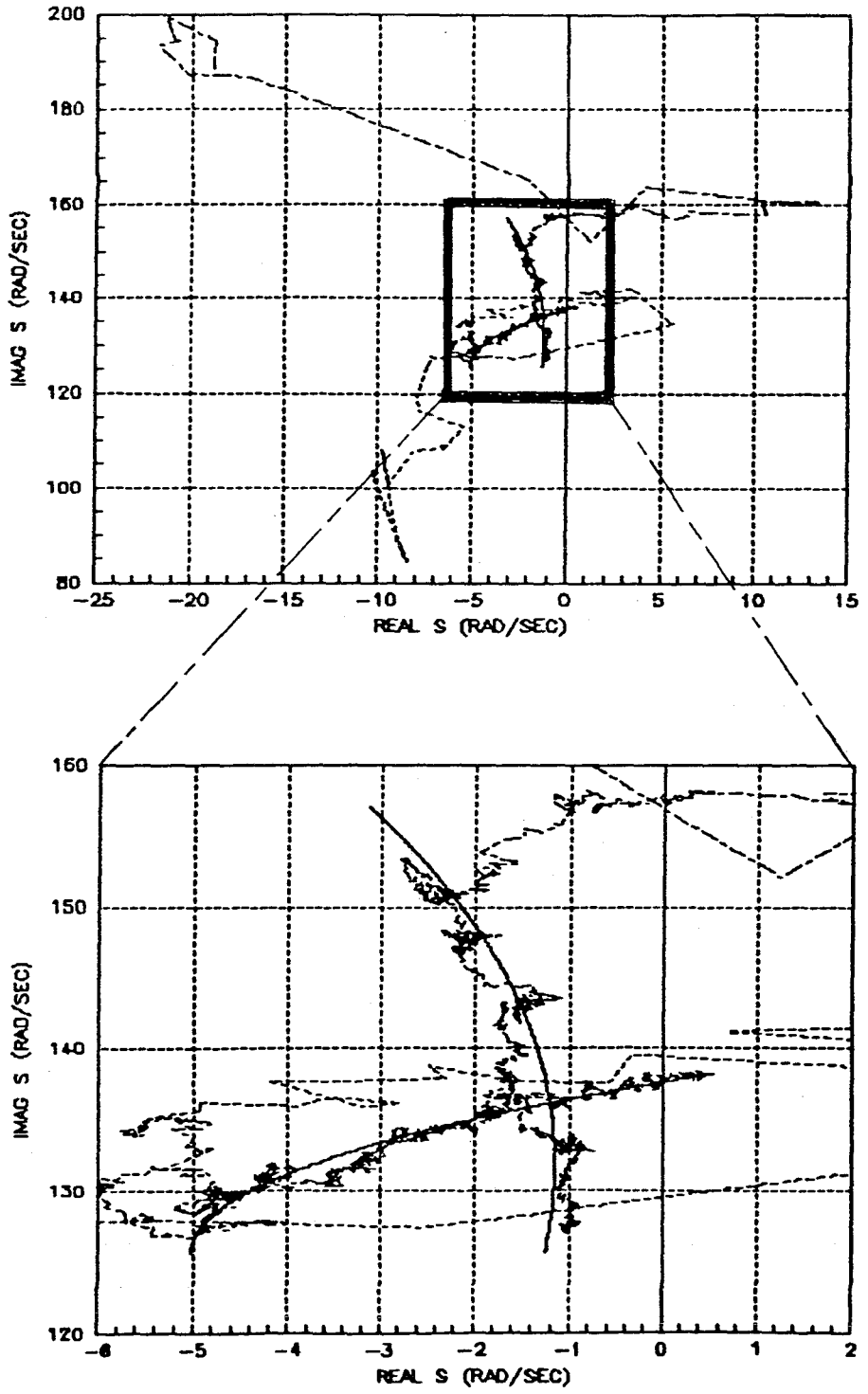


Figure 5-4: Simulated DAST Estimated and Actual Pole Location Time Histories

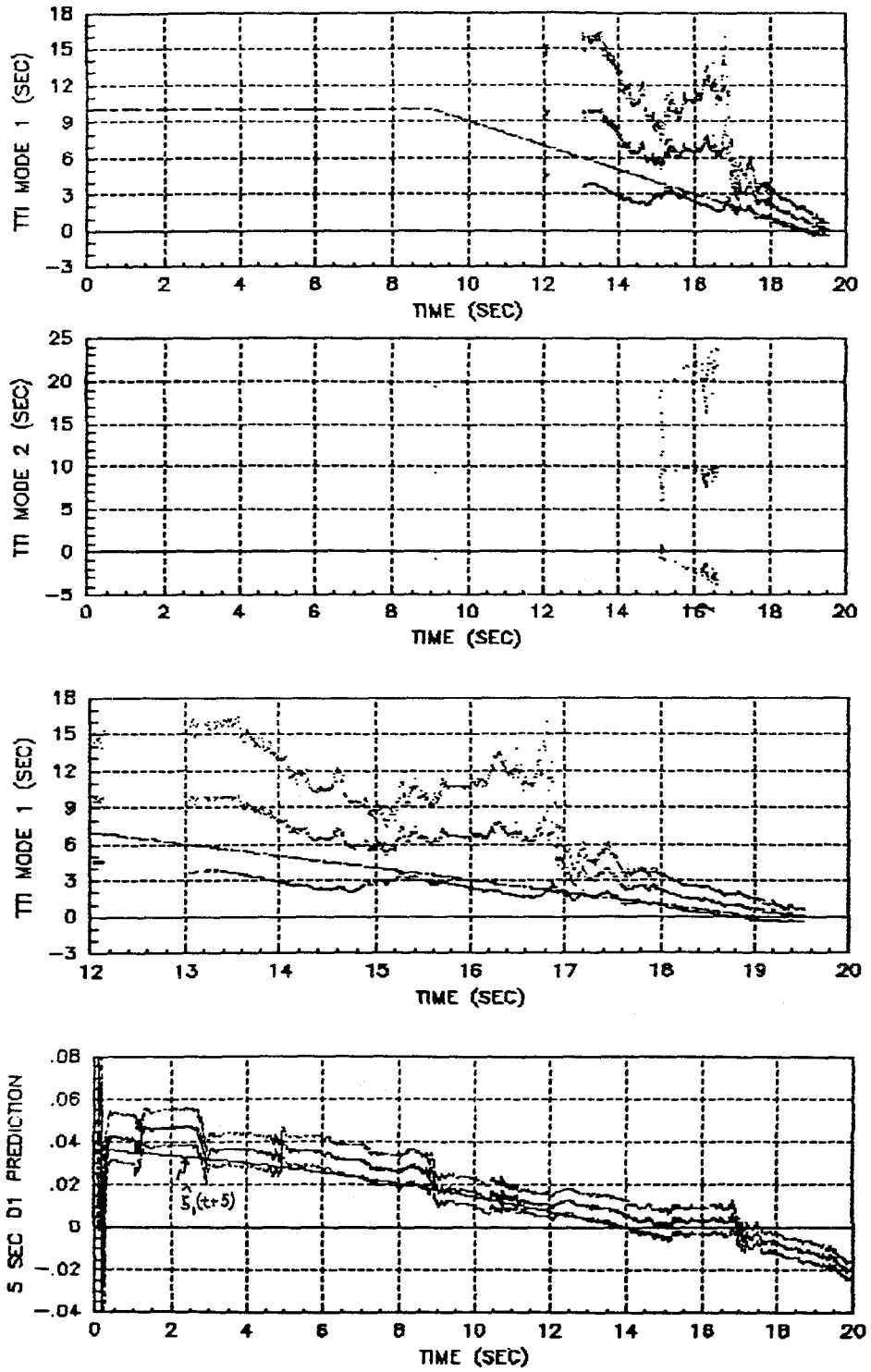


Figure 5-5: Simulated DAST Estimated Time-to-Instability and 5-Second Ahead ζ_1 Prediction

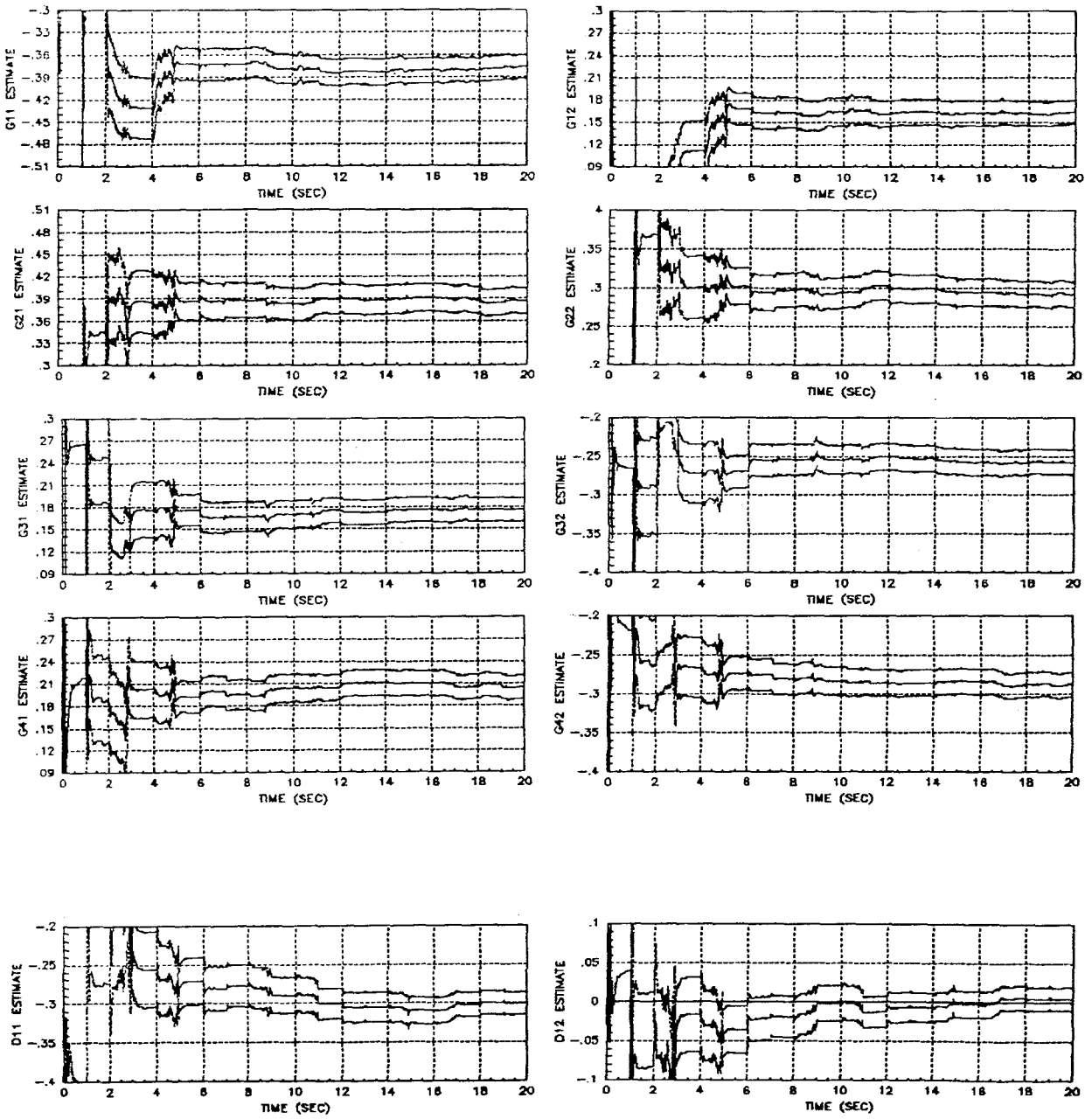


Figure 5-6: Simulated DAST G-matrix and D-matrix Parameter Estimates and Estimated Sigmas

5.2 SIMULATED F-16 TEST CASE

The objective of the simulated F-16 test case was to illustrate the performance of the EKF flutter parameter identification algorithm in the absence of deterministic inputs. The only forcing functions in such situations are the random disturbances normally encountered in flight; disturbances due to wind gusts and turbulence. A two-mode, two-output model was chosen to simulate the conditions of the F-16 test flight. The modal frequencies and damping coefficients were not time-varying for this test; ω_1 and ω_2 were set to 4.5 Hz and 8 Hz (28 rad/sec and 50 rad/sec) and the associated damping coefficients to 8% and 6% respectively.

The inputs to the two-mode model were two independent 50 Hz low-pass filtered white Gaussian noise processes; one driving the 'velocity' state of each oscillator. The square-root of the process noise variance density for both processes was set to 0.025. The outputs were chosen as linear combinations of the four dynamic states whose output power spectra as closely as possible approximated the power spectra of the actual F-16 outputs (cf. Section 6). Measurement noise with sigma 0.02 was added to the outputs as well. The resulting H-matrix was:

$$H_0 = \begin{bmatrix} 3.0 & -0.2 & 0.05 & 0.02 \\ -0.02 & 0.05 & 0.5 & -0.01 \end{bmatrix}.$$

The simulation was integrated at 400 Hz in order to more closely approximate the outputs of a continuous time system. Five seconds of simulation were run before the estimation was begun in order to allow the simulation to reach statistical steady-state. Five seconds were needed since the bias filter pole was set to 0.2 Hz in order to be a factor of 10 below the lowest mode of interest. The estimation data rate was chosen to be 100 Hz, roughly a factor of 10 greater than the highest frequency of interest.

Estimation Problem Formulation

In the estimation problem, all elements of H were estimated with initial values set as follows:

$$\hat{H}(0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

with sigma 1 for each element. Note that as discussed in Section 3, initializing H to zero herein is not appropriate since there are no exogenous inputs forcing the dynamic states. Zero H results in zero partials which lead to zero gains resulting in zero dynamic state estimates for all time. The initial frequency estimates were set to 5 Hz and 10 Hz with sigmas 0.5 Hz and 1 hz respectively. The associated damping coefficient estimates were initialized to 10% with a sigma of 2% for both modes.

The values for the square-root of the dynamic state process noise variance density (q's) were set to 0.025 in units of the state units per square-root time. Frequency normalized process noise was used in this simulation since the ratio of the frequencies of the two modes was approximately 2. Without using frequency normalized sigmas, different values would be required for the two modes (differing by the same frequency ratio of 2). This leads to potential problems however, since the process noise sigmas are a priori assigned to specific modes. Should the estimated frequencies of the two modes 'cross' due to random effects, the process noise variances would become inappropriate for the modes. Frequency normalization overcomes this problem by adjusting the variances proportional to the modal disturbability. A value of 0.0001 was also used for the q's on the elements of H even though the actual values were not time-varying. As in the DAST simulation, this was done to more closely approximate the conditions of the actual F-16 test flight. Q's of 0.0005 were used for the frequency states and 0.000005 was used for the damping coefficient velocity states.

Results

Figures 5-7 through 5-11 summarize the results of the simulated F-16 test case. Figure 5-7 shows the simulated measurements and the associated

predicted data residuals and their theoretical sigmas. The amplitudes of the outputs are seen to be comparable to those of the actual F-16 outputs (cf. Section 6). Figure 5-8 shows the power spectra corresponding to each of the plots in Figure 5-7. The output power spectra are shown to provide a meaningful comparison with the actual F-16 data whose power spectra are given in Section 6. The power spectra of the predicted data residuals are presented to demonstrate that the EKF algorithm has extracted 'all' of the information possible from the measurements. The spectra are flat with the exception of the effects of the bias filter at DC.

Figure 5-9 shows the time histories of the estimated pole locations in the s-plane. The two X's in the figure indicate the true locations of the poles. As is intuitively obvious, there is more information in the data concerning the frequency of the pole locations than the damping coefficient. This results in the observed variations in damping coefficient being much larger than those in frequency. Figure 5-10 gives the time histories of the frequency and damping coefficient estimates and their estimated sigmas as well. Again the relative information content with respect to frequency and damping coefficient is clear from the 'plus and minus' one-sigma values indicated. Furthermore, the rate of decrease of the one-sigma values for the damping coefficient estimates is quite small, indicating that little information is being acquired. The overall rate at which information is accumulated by the algorithm is directly proportional to the input process noise power to measurement noise power ratio; thus, increasing the disturbance power will increase the rate of convergence of the sigmas and decrease the final steady-state sigmas as well.

Figure 5-11 presents the estimates and estimated sigmas for the elements in both rows of the H-matrix. The final H-matrix parameter estimates were:

$$\hat{H}(40|40) = \begin{bmatrix} 1.3 & 0.6 & 0.3 & 0.05 \\ -0.4 & -0.06 & -1.9 & -0.4 \end{bmatrix},$$

Comparison of these asymptotic estimates with the true values given above indicates that there is very little similarity, and for this reason the parameter estimate errors are not shown. The primary reason for the nonzero asymptotic parameter estimate errors is mismodeling of the (stochastic) inputs. Two independent process noise sequences were used as inputs in the

simulation as discussed above. However, the model used in the estimation problem formulation effectively assumes the presence of four independent noise processes by allowing tuning for each of the oscillator states separately. The result is that at each measurement update, the filter is given freedom to adjust the position and velocity states of each oscillator independently. The interdependence of the states which is a consequence of the natural oscillator equations of motion is partially destroyed; more specifically, the derivative-integral relationship specified by the dynamical model is no longer exactly satisfied. Equivalently, the phase relationship between the two states is partially destroyed. In effect, the problem has been overparameterized stochastically, and this leads to the observed steady state H-matrix parameter estimate errors. The overparameterization also manifests itself in the slow convergence of the damping coefficient estimates and the relatively large steady state estimate error sigmas.

A second factor contributing to the H-matrix parameter estimate errors is the difference in the simulation and estimation data rates. By using an integration rate in the simulation of the data which was a factor of four larger than the estimation data rate, effectively different bandwidth noise processes were being used. This contributed further to the stochastic mismodeling.

Another factor contributing to the H-matrix parameter estimate error is the presence of the bias rejection, or high-pass filters. By removing the low-frequency content in the data, information concerning the location of the input-output zeroes near the origin is suppressed relative to high frequency components. Thus, disparity between estimated H-matrix elements and the 'true' values is to be expected. However, since fidelity of the H-matrix parameter estimates is not the primary goal in the modal parameter identification problem, the lack of convergence to the "true" values is of no real concern. Note that the modal parameter estimates do not suffer from this lack of identifiability; and they are the parameters of interest.

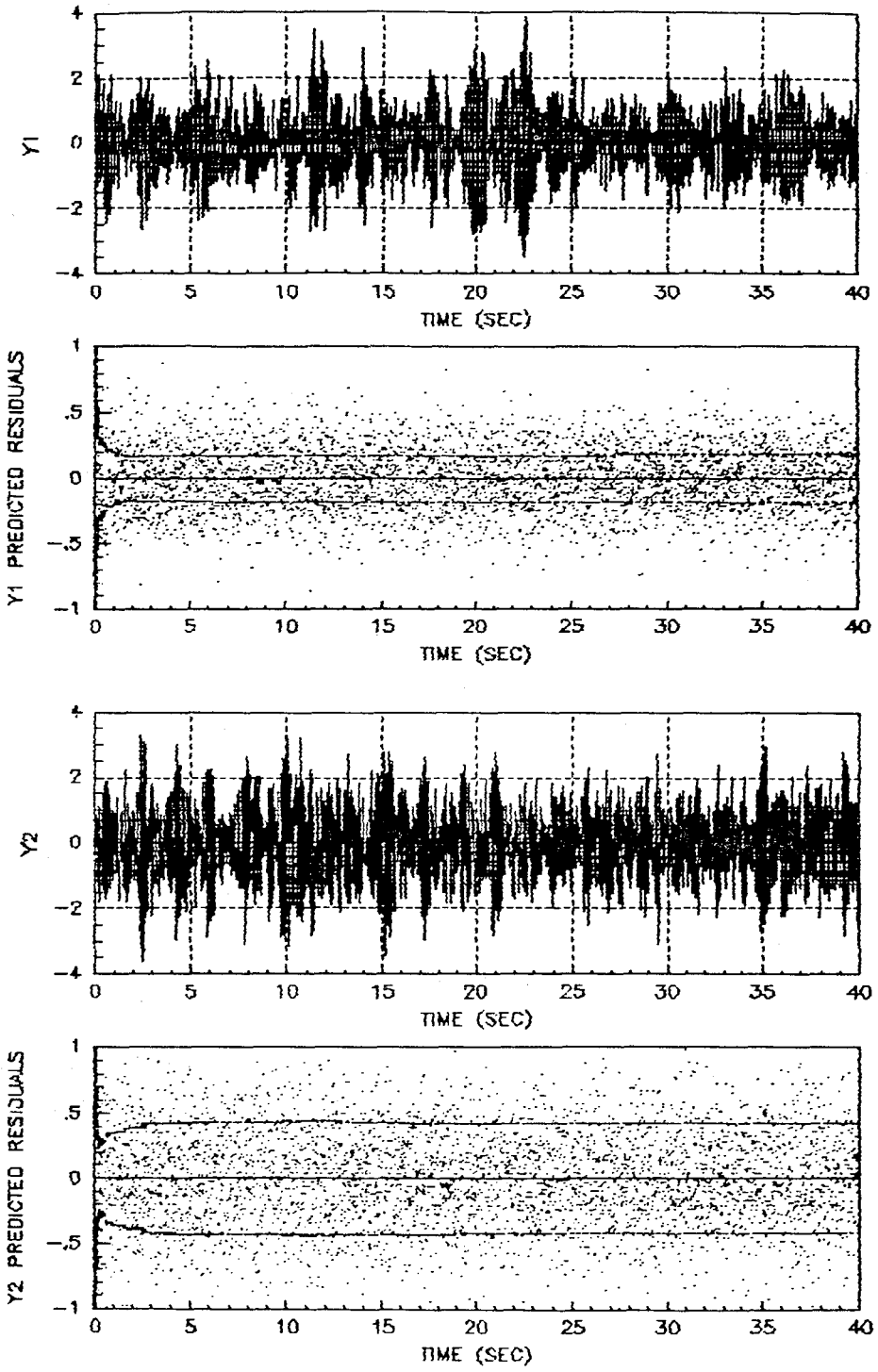


Figure 5-7: Simulated F-16 Outputs and Predicted Data Residuals

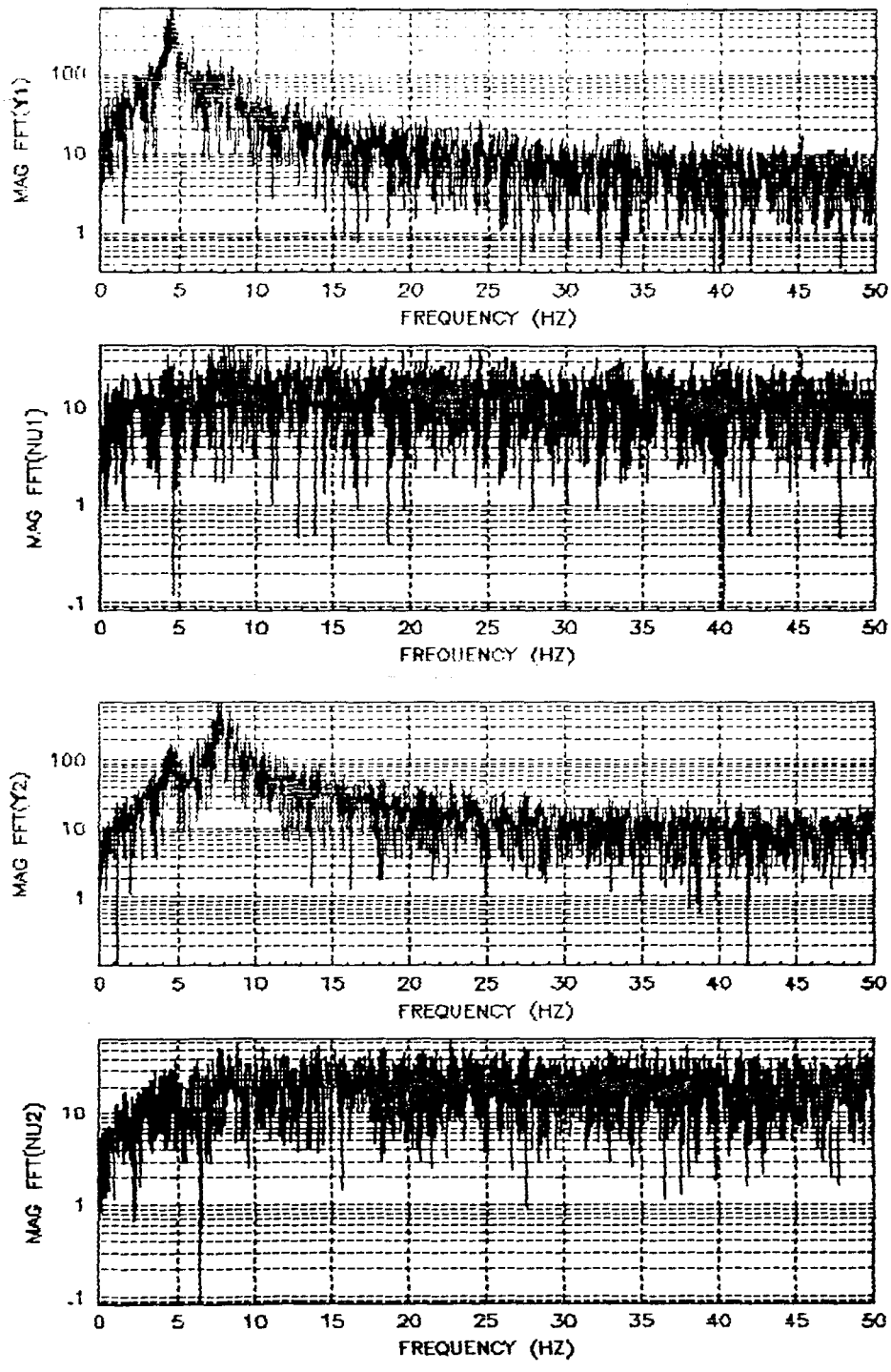


Figure 5-8: Simulated F-16 Output and Predicted Data Residual Power Spectra

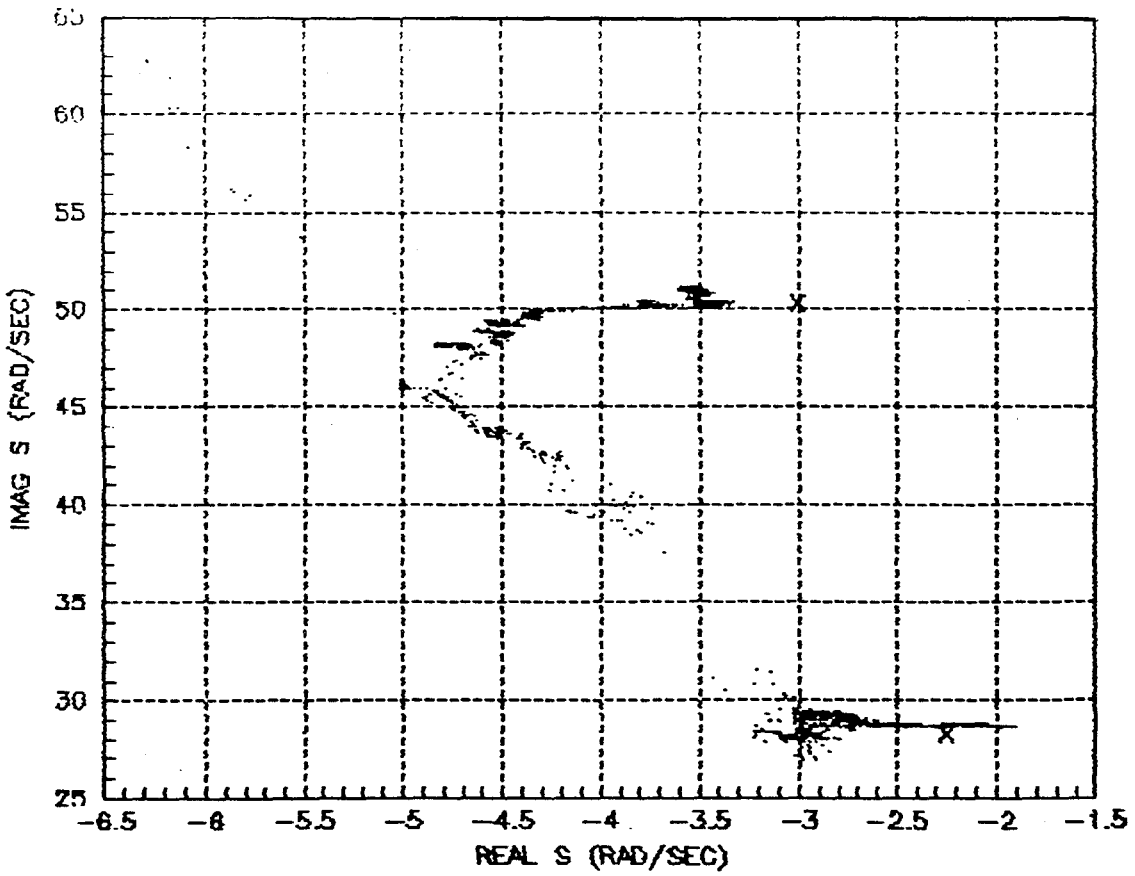


Figure 5-9: Simulated F-16 Estimated Pole Location Time Histories

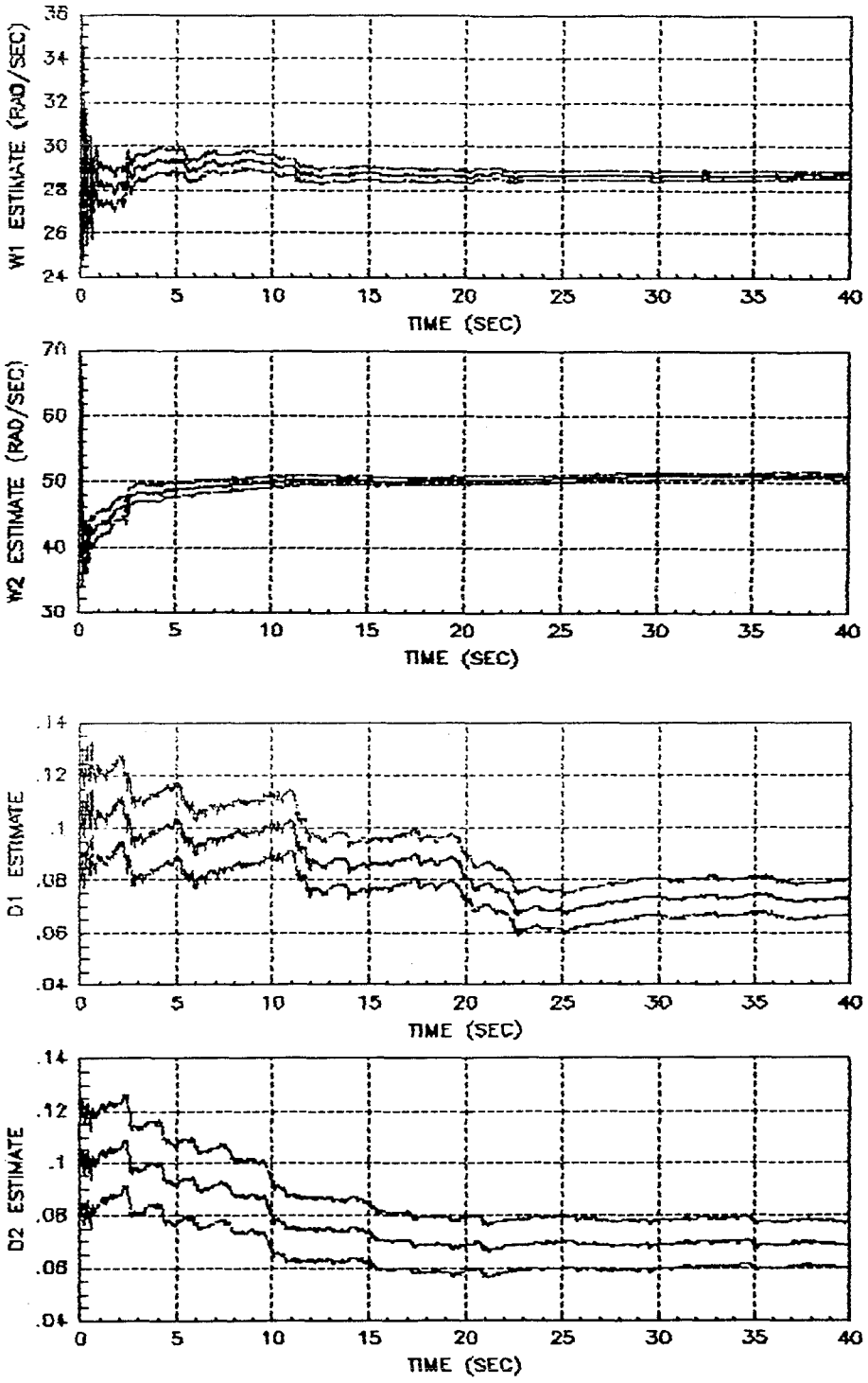


Figure 5-10: Simulated F-16 Estimated Modal Parameters and Estimated Sigmas

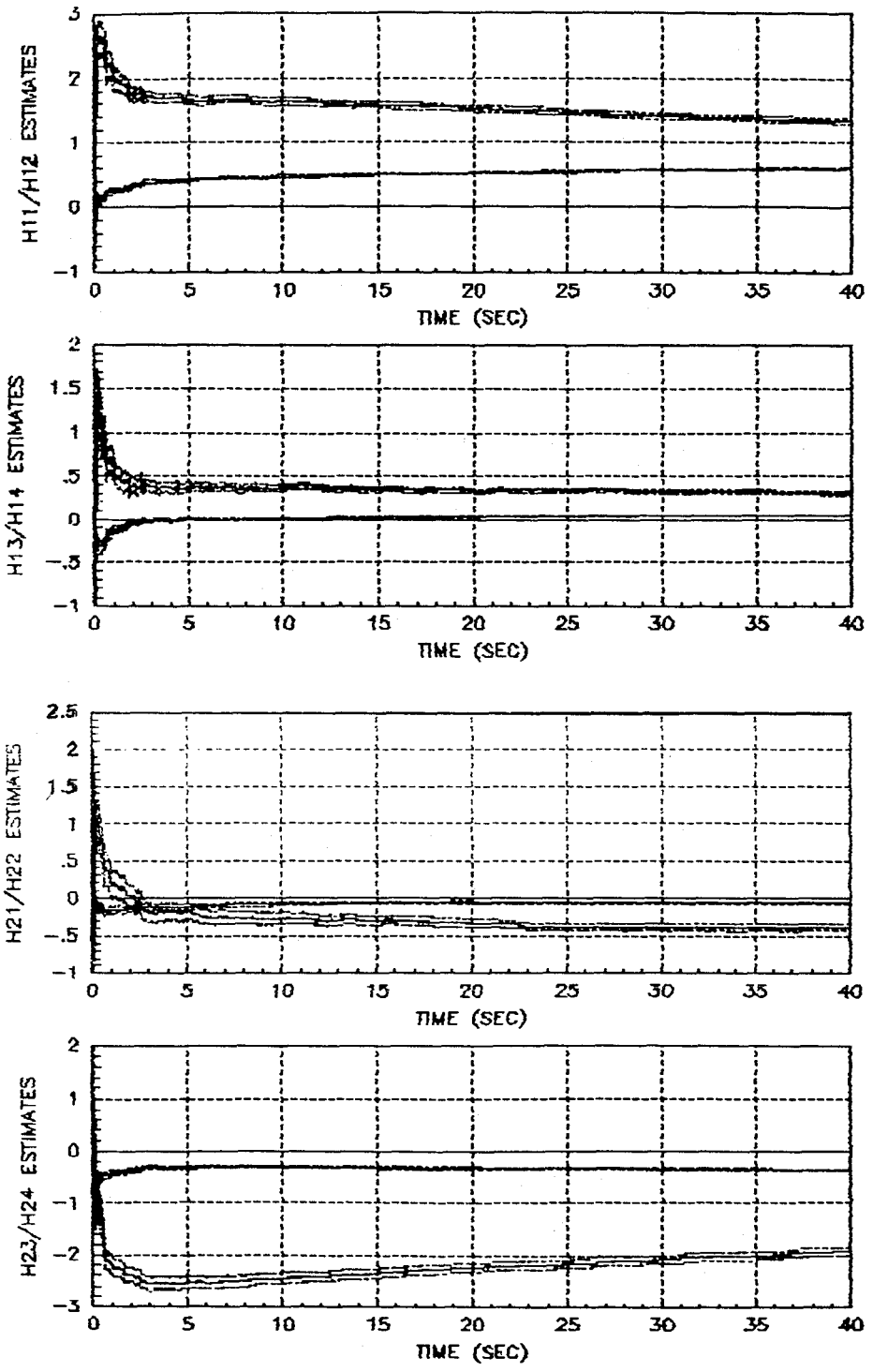


Figure 5-11: Simulated F-16 Selected H Parameter Estimates and Estimated Sigmas

SECTION 6
FLIGHT TEST RESULTS

In the previous section, the results of two simulated data test cases were presented in order to illustrate the EKF algorithm's performance in situations similar to those encountered in the actual flight test data which were processed. This section presents the results of the processing of the actual DAST and F-16 flight test data. Since the actual values of the frequency and damping coefficient estimates are not available, the discussion of the results necessarily involves consistency arguments where comparison is made between 'batch processed' results (i.e. FFT analysis) and the recursive EKF results. Particular attention is paid to the estimates of the parameter error variances, since these are an indication of the amount of information contained in the data available about the parameters of interest.

The first flight test to be discussed is the third DAST flight of the ARW-1 wing. This test is of particular value in ascertaining the performance of the EKF algorithm as a real-time flutter parameter monitor for two reasons. First, throughout the test the ailerons were given sine-wave pulse and swept-sine commands to excite the structural modes. Secondly, due to an FSS implementation error, the vehicle experienced severe flutter which caused the right wing to separate from the fuselage with subsequent ground impact. Since it is of great interest to know how the algorithm performs in such circumstances, the data processing concentrated on the last 100 seconds of this flight.

The second flight test discussed is an F-16 flight in which no exogenous inputs were applied to the control surfaces. The data represent approximately 90 seconds of accelerometer outputs during a period of turbulence induced vibration (flutter). Since very little else was known about the F-16 flight test at the time of analysis, the discussion of the results is limited to comments concerning the general identifiability of flutter parameters without exogenous inputs.

6.1 DAST FLIGHT TEST RESULTS

Background and Data Base

The purpose of NASA's drones for aerodynamic and structural testing (DAST) program is to test aeroelastic research wings (ARW) in an attempt to correlate theoretical predictions and experimental flight test results of aeroelastic effects in the high subsonic to transonic speed range. The first wing to be tested in the program (ARW-1) was a sweptback, supercritical airfoil, transport type wing. The primary research objective of the ARW-1 was to investigate techniques for the active control of flutter utilizing an onboard analog flutter suppression system (FSS). Three test flights of the ARW-1 were conducted. An error in the implementation of the gain in the FSS caused a violent flutter incident at the end of the third flight. The right wing separated and the vehicle impacted the ground.

TABLE 6-1: DAST FLIGHT TEST CHANNEL IDENTIFICATIONS

Channel	Descriptor	Description
1	ALFSO	Left-wing front spar outboard accelerometer
2	ALFSS	Left-wing FSS accelerometer
3	ALRS	Left-wing rear spar accelerometer
4	ARFSS	Right-wing FSS accelerometer
5	ARRS	Right-wing rear spar accelerometer
6	DAL	Left aileron position
7	DAR	Right aileron position
8	FSSEXC	FSS excitation - aileron command

The data available for processing at the time the analysis was performed included eight telemetry channel outputs; five accelerometer outputs, two aileron position outputs, and an aileron command signal (the exogenous input used to excite the structural modes). The channels were labelled 1 through 8 and the Table 6-1 gives the channel identifications. The data sampling rate was 500 Hz and several minutes of data prior to the flutter incident were on the tapes. During this time period the drone was being accelerated from Mach 0.7 to Mach 0.825 and was at an altitude of 15000 feet.

The FSS excitation signal was composed of 'pulses' which contained exactly one cycle of a 20 Hz sinewave, with either 1.7° or 3.4° amplitude, and logarithmic frequency sweeps from 10 Hz to 40 Hz in 7 seconds with 1° or 2° amplitude. The sweeps were tapered at both ends to eliminate transients. The equation for the sweeps was given in Section 5. The FSS excitation input was applied both symmetrically and asymmetrically to the ailerons with the FSS both ON and OFF during various segments of the flight. The objective was to excite the symmetric as well as the asymmetric modes of both the closed-loop and open-loop system. However, as the speed approached Mach 0.8, the FSS OFF tests were terminated as the vehicle was predicted to be open-loop unstable near that altitude and Mach number. Furthermore, after exceeding Mach 0.8, asymmetric excitation was terminated altogether in favor of symmetric excitation since the closed-loop symmetric modes were predicted to be unstable above about Mach 0.85.

Estimation Problem Formulation

As discussed earlier, there was quite a large amount of data present, however, only a limited amount of data was processed in this analysis. Since the primary objective of the EKF algorithm is to identify potential instabilities and provide some indication of pending flutter, the analysis was conducted over the last 100 seconds of the test flight. Less data could have been used, however in order to include at least one symmetric and one asymmetric sweep excitation in the interval, the processing was started at 26035 seconds (07:13:55) as indicated on the data tapes. During the last 100 seconds of flight, the vehicle was initially stabilized at Mach 0.8 to obtain sweep responses, then accelerated to Mach 0.825 near which point the flutter incident occurred. The FSS was ON during this time interval.

Since the expected flutter frequencies were in the 20 Hz region, the sampling rate of 500 Hz was more than sufficient to adequately sample the analog device outputs. Furthermore, this algorithm is expected to run in real-time, and a realistic upper bound on the average throughput rate is about 250 Hz (cf. Section 7). Thus, the analysis was conducted with data sampled at 250 Hz initially. In order to provide some insight into the effects of sampling at lower rates for problems of this nature (which may be

necessary due to computational speed limitations), the data were also processed at a 125 Hz rate. The altering of the sample rates was performed by placing a lower bound (in software) on the minimum sampling interval.

The original estimation problem formulation was based on closed-loop analysis at 250 Hz with two inputs, one output, and two modes (2,1,2). Since both the aileron command and the aileron position were available, there was a choice of performing closed-loop or open-loop analysis. The open-loop analysis basically uses the aileron positions as system inputs and the accelerometers as system outputs and identifies (under certain persistent excitation conditions) the natural frequencies of the open-loop (or uncontrolled) system, i.e. the natural flutter frequencies. The closed-loop analysis uses the command signal as the system input instead, and thereby results in estimates of the modes of the controlled system (assuming the FSS is ON). Since the closed-loop system is the system whose stability is critical in the real-time environment, the closed-loop analysis was performed.

The inputs to the original (2,1,2) estimation problem were the right wing FSS accelerometer (ARFSS - Channel 4) and the FSS excitation signal (FSSEXC - Channel 8). The two inputs were composed of the FSSEXC time history and its appropriately negated counterpart; the excitation signal generated by accounting for the fact that some of the sweeps and pulses were symmetric and others asymmetric. However, there were no asymmetric pulses or sweeps over the last half of the estimation interval, presumably because the symmetric modes were expected to be the least stable as the flutter boundary was being approached. Thus, the two inputs were identical for most of the interval, resulting in a lack of identifiability of the asymmetric component of the input distribution matrix (G). Therefore, an alternate, computationally more attractive approach was used.

Instead of constructing a two-input problem, from one which was inherently a single-input problem, the estimation problem was reformulated as a single-input (1,1,2) problem. During the few instances in the estimation interval when the symmetry of the excitation was switched from symmetric to asymmetric, impulses in the parameter process noise variance densities (q's) were used to allow the parameters to adjust to the 'new conditions'. Basically, this amounted to estimating different models for symmetric and asymmetric excitation periods. Since the actual modes excited during the

symmetric and asymmetric excitation periods were different, and since the level of exogenous input far exceeded that due to random effects, the identification of a new model after each excitation symmetry switch was justified.

The use of impulsive q's has the effect of decoupling the estimation of the modal parameters from one type of excitation to the next occurrence of the same polarity excitation if an excitation of the opposite polarity intervenes. However, due to the dynamics of the changes in the damping coefficients and frequencies over the last 100 seconds of flight, the q's on the frequency and damping coefficient states required to track their time-variations outweighed the impulsive contributions as far as the critical mode was concerned. Very little information regarding the instability was lost. Consequently, the estimates of the means and variances of the performance measures discussed in Section 4, TTI and T-second ahead prediction, were not significantly affected.

Another issue in the estimation problem formulation concerned the estimation of a direct feedthrough term. Such a term was estimated for two reasons. First, the underlying physics of the process relating the aileron deflections to the observed accelerations dictates that a certain amount of the force generated by the deflection is transmitted directly to the observation point virtually without delay (certainly on the time scale of these problems the delay is negligible). Secondly, it was clear from extensive plotting of outputs and excitations that a direct feedthrough term was present. A sample of such output is shown in Figure 6-1. In the plot on the

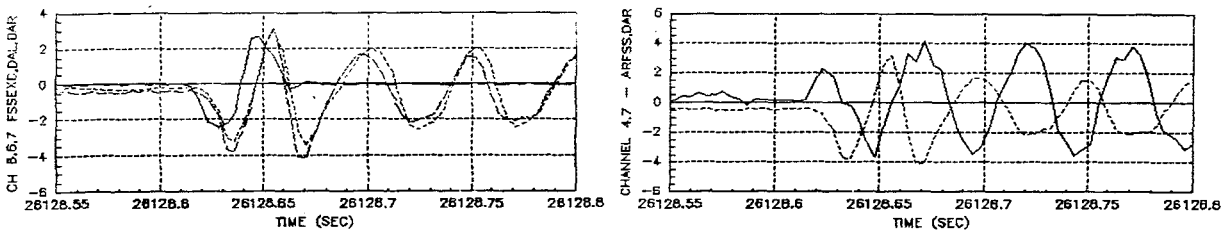


Figure 6-1: Expanded DAST Pulse Excitation and Responses Indicating Presence of a Direct Feedthrough Component

left, a pulse excitation (solid line) is shown along with the aileron deflections (dashed lines). The FSS was ON and the excitation was

symmetric. There is clearly a significant lag in the aileron deflections. The plot on the right shows the right aileron position (dashed line) overlayed on the ARFSS output (solid line). Assuming there were no telemetry sampling problems and no excessive differential delays in the data acquisition system, there is clearly a component of FSSEXC in ARFSS since ARFSS leads significantly DAR (the aileron position). Therefore, a **D**-matrix parameter was estimated in spite of the increased computational burden. The intial value was set to zero and the initial sigma to 0.5 g/deg.

The increased computational burden resulting from the estimation of an element of the **D**-matrix was small in this case due to the fact that for a majority of the estimation interval, the excitation was zero. By taking into account the zero value of the input, the EKF algorithm was optimized to minimize unnecessary multiplies by zero thus realizing significant computational savings. Furthermore, since the excitation over the last 50 seconds consisted solely of short duration pulses, very little estimation accuracy would have been lost if the direct feedthrough term were neglected. However, since a major conclusion of this analysis is that continuous wideband excitation significantly improves estimation accuracy, the direct feedthrough term was retained.

Thus, the estimation problem for the last 100 seconds of the DAST flight was implemented as a (1,1,2) estimation problem with direct feedthrough by taking advantage of the time multiplexed nature of the multi-input aspect of the problem. Both channels, FSSEXC and ARFSS, were filtered with the bias rejection filter with the pole set at 1 Hz. This was required as discussed in Section 4 in order to eliminate the low-frequency center-of-mass motion as well as the static lift force/zero-g accelerometer output level. Inspection of the unfiltered ARFSS output shown in Figure 6-2 manifests the need for such bias rejection (cf. zero-level shift at 26045 seconds). Though highly compressed, the time history of the excitation signal (FSSEXC) also shown in Figure 6-2 indicates the sequence of sweeps and pulses eluded to earlier.

As in the simulated DAST estimation problem, estimation of the parameters in **G** was chosen in favor of those in **H** due to the inherently bilinear nature of **H**-matrix parameter estimation compared to the linear nature of the **G**-matrix parameter estimation. Initial values for the **G**-matrix parameters were set to zero with unit variance. These were appropriate

since autonormalization of the G-matrix elements was used (cf. Section 3.3). The measurement distribution was arbitrarily set to:

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

without loss of generality. Inspection of the data indicated that a measurement noise sigma of 0.1 g's was appropriate, and it was used throughout the analysis.

The initial values for the modal parameters were set based on the a priori knowledge that the frequencies were in the 15 to 30 Hz range and the damping coefficients were on the order of 5%. The initial frequency estimates were set to 20 and 25 Hz with sigma 1 Hz, and the intial damping coefficient estimates were set to 0.05 with sigma 0.02. Since no a priori knowledge concerning the damping coefficient velocities was available, they were initially set to zero with sigma 0.0002.

The tuning of the process noise variance densities was performed as discussed in Section 3.2. The values for the dynamic state q's were set to unity in the associated state units. No impulsive q's were added to the dynamic state q's at input excitation symmetry switching times. The frequency q's were set to 0.02, and at the times of input symmetry switching (26046.5, 26067.9, 26089.9, 26098.0, and 26102.0 seconds) an impulsive q value of 10 was added over a single prediction interval. A q of 0.0002 was used for the damping coefficient velocities, with an impulsive q of 0.02 used. The damping coefficient q's were set to zero (so that the velocities would integrate to the coefficients); however, impulsive q's of 0.4 were used to allow for immediate tracking of new damping coefficients resulting from different modes being excited. The q's for the G-matrix and D-matrix parameters were set to 0.0001 to allow for tracking of slow time-variations, with impulsive q values of 10 used to allow adjustment when the symmetry of the excited modes changed.

As is easily seen from the initial values, there was quite a dynamic range in this problem. In order to avoid potential numerical difficulties, units normalization was performed on the estimated states as discussed in Section 3.3.1. The frequency units were changed from rad/sec to 100 rad/sec. The damping coefficients were estimated in percent (i.e. 0.01 units), and the units for the 'velocity' states in the two oscillators were

set to 100 times there unscaled units. Other than the fact that the 'velocity' states must have the units of the 'position' states per unit time, the units of the dynamic states are completely arbitrary. It should be noted that these unit conversions are internal to the algorithm, and that before outputting any results for plotting, the algorithm converts back to original units. Thus, frequency estimates are plotted in rad/sec, even though internally calculations were performed in 100 rad/sec units.

RESULTS FOR 250 HZ DATA RATE

Figures 6-2 through 6-10 present the results of the DAST data analysis for a sampling rate of 250 Hz. Figure 6-2 shows the algorithm inputs ARFSS and FSSEXC prior to high-pass filtering. Of particular note is the ARFSS response to the symmetric pulse at 26090 seconds indicating near instability, and the 'unforced' ARFSS output at 26132 seconds. This unforced output was probably gust/turbulence induced. Due to the fact that the asymmetric modes were more heavily damped during this transition from Mach 0.80 to 0.825, the asymmetric and symmetric excitations are easily found by noting the duration of the associated pulse responses. The peak response to the symmetric sweep is also significantly larger than the asymmetric sweep peak response. Also shown in this figure are the predicted data residuals along with their theoretical 1-σ values. It should be noted that due to core limitations, only every tenth residual was saved for plotting. The sample sigma of these residuals was 0.25 in good agreement with the steady-state theoretical value (approximately 0.20). The 'spikes' in the sigma estimate are due to the impulsive q's added at input symmetry switching times, and the vertical line at 26110 seconds is the result of the outlier rejection algorithm placing zero values where data were rejected.

Figures 6-3 and 6-4 give the time histories of the parameters being estimated with their plus and minus 1-σ values plotted as well. The modal parameter estimates are given in Figure 6-3. The large jumps in the parameter estimates occur at input excitation symmetry switching times where impulsive q's allow rapid parameter variations, though smaller impulsive adjustments also occur at the times of each of the pulses. These small adjustments are due to the impulsive nature of the increase in information content in the data at these times.

The behavior of the frequency and damping coefficient estimates during the frequency sweep excitations is quite interesting. During the first (asymmetric) sweep, low variance estimates of a pole near 115 rad/sec and 155 rad/sec were achieved for modes 1 and 2 respectively. The estimate of the frequency for mode 1 actually settled for a short while at 120 rad/sec before converging at the end of the sweep interval to 115 rad/sec. These results are consistent with the FFT derived SISO transfer functions shown in Figure 6-5. Plots of the data from all eight channels are shown at the top of the figure, while the transfer functions are shown at the bottom. Of particular note is the dramatic difference in the 'left-wing' transfer function estimates on the left versus those for the 'right-wing' accelerometer outputs shown on the right. This dramatic asymmetry is evidence of either an asymmetry in the wing configuration or a possible control law problem. The ARFSS/FSSEXC power spectrum (the solid line on the bottom right plot) indicates two poles at 20 and 25 Hz (125 and 155 rad/sec respectively). The mode 2 frequency estimate 'locks-on' to the power at 25 Hz, while the mode 1 estimate converges to 20 Hz at 26040 seconds, drifting down about 1Hz as the sweep excitation frequency exceeds 20 Hz. This could be the result of momentary excitation of a 20 Hz symmetric mode between a 15-17 Hz asymmetric mode and a 25 Hz asymmetric mode.

Figure 6-6 presents FFT transfer function estimates for the response to the symmetric sweep. The symmetric transfer function estimates display the expected similarity as seen at the bottom of Figure 6-6. Furthermore, the symmetric mode at 20 Hz is more lightly damped than any of its asymmetric counterparts (note the decreased variance in the FFT transfer function estimates as well), resulting in low variance estimates as indicated in the mode 1 parameter plots. The mode 2 frequency estimate step to a value near 175 rad/sec (28 Hz) at 26055 seconds which is possibly a moderately damped mode (cf. ζ_2 estimate) momentarily excited by the sweep excitation. It might also be the first harmonic of a mode in evidence near 15 Hz (note the 'notch' at 15.5 Hz).

From 26068 to 26090 seconds, there were four successive asymmetric pulse excitations (cf. Figure 6-2). The frequency and damping coefficient estimates (especially those for mode 1) are quite different than their respective values during the symmetric excitation periods as expected. The mode 1 estimate, being initially closer to the 'new' mode and allowed to

move due to the impulsive q's added, did so. The increased variance in the estimates is worthy of comment as well. As noted above, the asymmetric modes were more heavily damped than the symmetric modes. Since the excitation was limited to short duration pulses which resulted in short duration responses, much less information was contained in the data than during the sweep excitations or the extended responses to the symmetric pulses. The reduced information content resulted in increased variances (sigmas). This is quantitatively evidenced in the significantly smaller estimate sigmas at 26045 seconds, the end of the asymmetric sweep interval, and the associated sigmas at any time during the sequence of asymmetric pulses. By continuously exciting the asymmetric modes during the sweep, more information concerning the damping coefficient and frequency was extracted.

The effect of the impulsive q's is most dramatically manifest in the response of the algorithm to the extended system pulse response initiated at 26090 seconds. The estimates rapidly converge to appropriate values for the lightly damped symmetric mode which was excited, and just as rapidly return to values appropriate to the asymmetric modes when the next asymmetric pulse excitation occurs at 26098 seconds (cf. Figure 6-3).

From 26100 seconds to the end of the flight, only symmetric pulse excitation was used. As a result the output power was concentrated near 20 Hz in what $\delta\alpha$s clearly the dominant mode. During this interval, the mode 2 frequency estimate apparently drifts off. However, inspection of the power spectra shown in Figure 6-7 indicate that there was sufficient power at the third harmonic (60 Hz or 380 rad/sec) for the mode 2 estimate to 'lock onto' the third harmonic. The third harmonic became pronounced just prior to the flutter incident (due to the increased amplitude of the response and possible attendant nonlinearities) and the frequency and damping coefficient estimates and their associated variances responded accordingly as seen in Figure 6-3.

Figure 6-8 shows an s-plane plot of the time histories of the estimated pole locations. Figures 6-9 and 6-10 present the most important results in terms of real-time flutter parameter and stability monitoring, the TTI and T-second ahead predictions. At the top of Figure 6-9, the TTI estimate for mode 1 is shown for the entire estimation interval. As discussed earlier, TTI estimates larger than 10 seconds and less than zero were arbitrarily set to zero to avoid plot scaling problems. Thus the estimates

appear to transition abruptly from zero to non-zero values and vice versa as the thresholds are crossed. The estimated time-to-instability for mode 2 was never less than the 10 second upper bound, so it is not shown. During the extended symmetric pulse response from 26090 to 26095 seconds, the TTI estimate indicated potential instability within a few seconds, however the estimated sigma was as large as the estimate itself during the response period, and increased steadily thereafter as seen in the second and third plots in Figure 6-9. In contrast, during the response to the last pulse, the TTI estimate and its sigma continually decrease until just prior to 26140 seconds, instability is indicated. Figure 6-10 shows the time history of the mode 1 damping coefficient estimate, its 5-second ahead prediction, and the TTI estimate over a 10 second interval just prior to the flutter incident. The 5-second ahead ζ_1 estimate plot also displays the ζ_1 estimate retarded by 5 seconds. During the period where the excitation is non-zero (the turbulence response at 26131.5 seconds followed by the pulse at 26132.5 seconds), the estimates are in close agreement. During the 4-second gap in the excitation, the estimates drift apart, clearly indicating the value of continuous excitation during these critical periods of the test flight.

RESULTS FOR 125 HZ DATA RATE

In order to ascertain the effect of reducing the input data (information) rate, the same estimation was performed at 125 Hz. The results of this estimation are given in Figure 6-11 through 6-16. The figures are in the same format as the corresponding ones for the 250 Hz estimation for ease of comparison. The results are quite similar as they are expected to be since the sampling rate is still a factor of 5 above the modal frequencies of interest. However, there is an increase in the variance of the parameter estimates over most of the estimation interval due to the loss of information. Figures 6-15 and 6-16 indicate that in spite of the reduced data rate, the TTI and 5-second ahead prediction estimates are still significant instability indicators. Quantitative differences between the results for the two data rates can be seen in the increase in the indicated time of instability (approximated 0.4 seconds). The key point to be made concerning both sets of results is that higher quality (lower variance) estimates are produced during periods of continuous excitation.

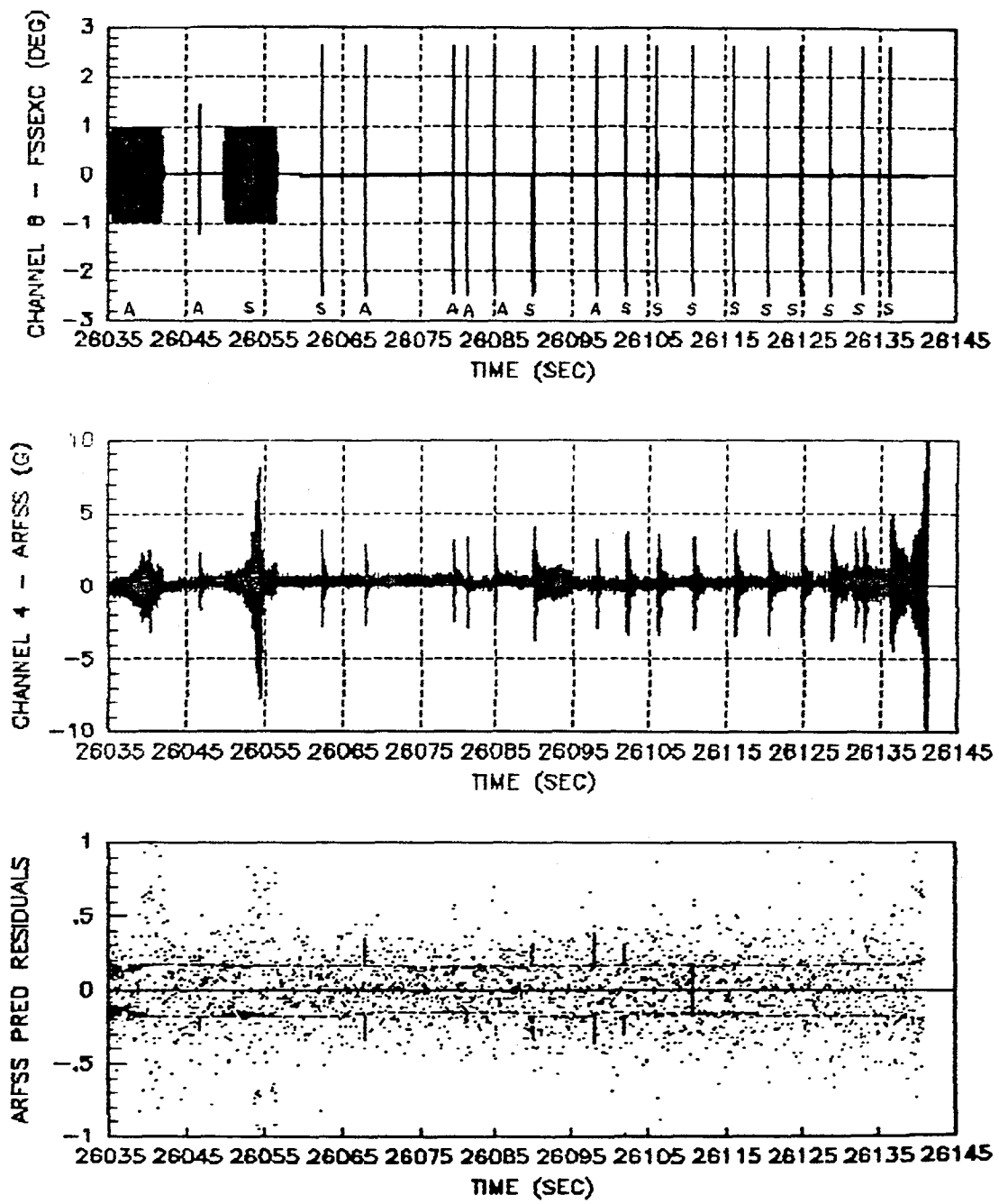


Figure 6-2: DAST 250 Hz Inputs, Outputs, and Predicted Data Residuals

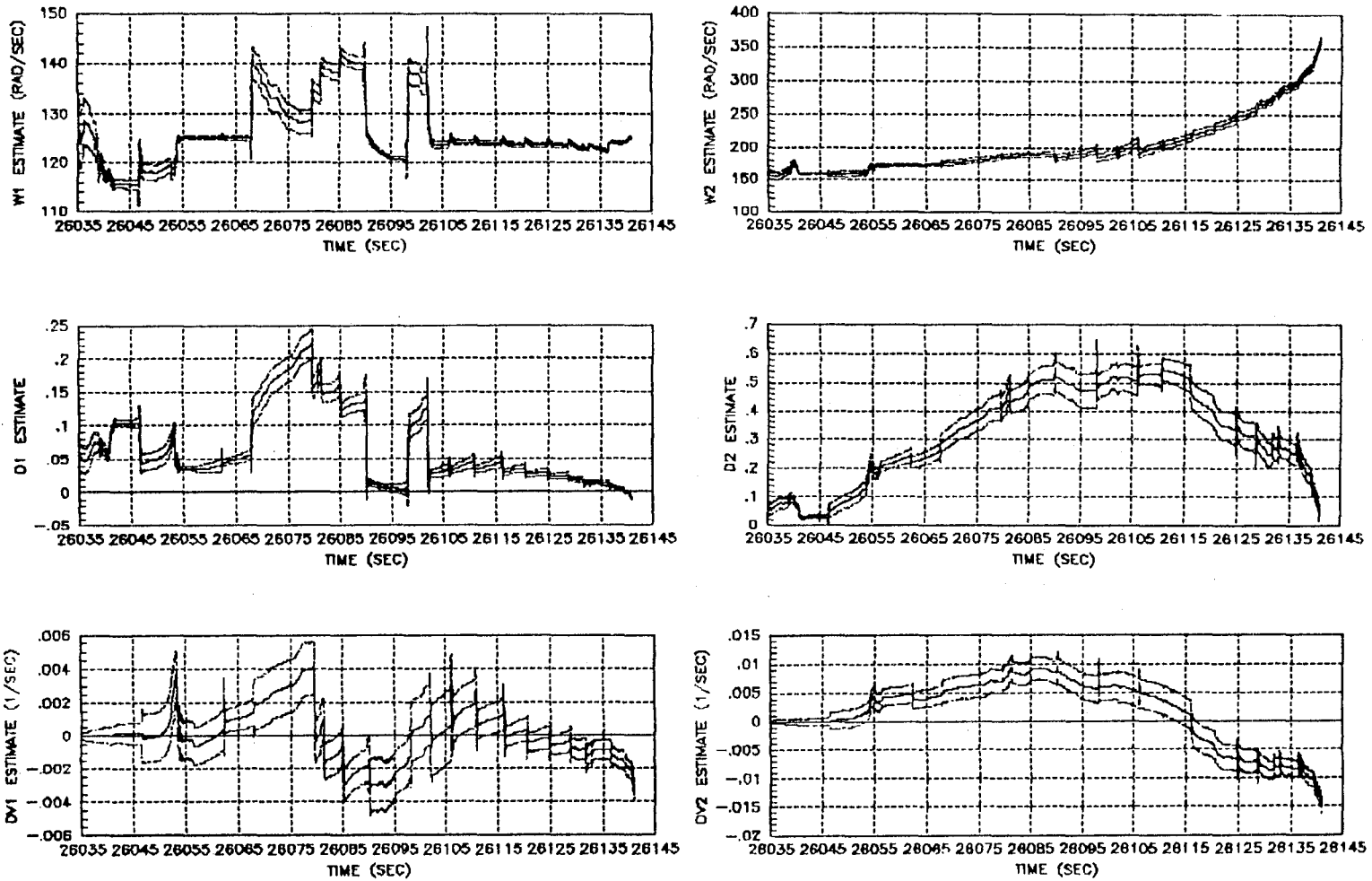


Figure 6-3: DAST 250 Hz Modal Parameter Estimates and Estimated Sigmas

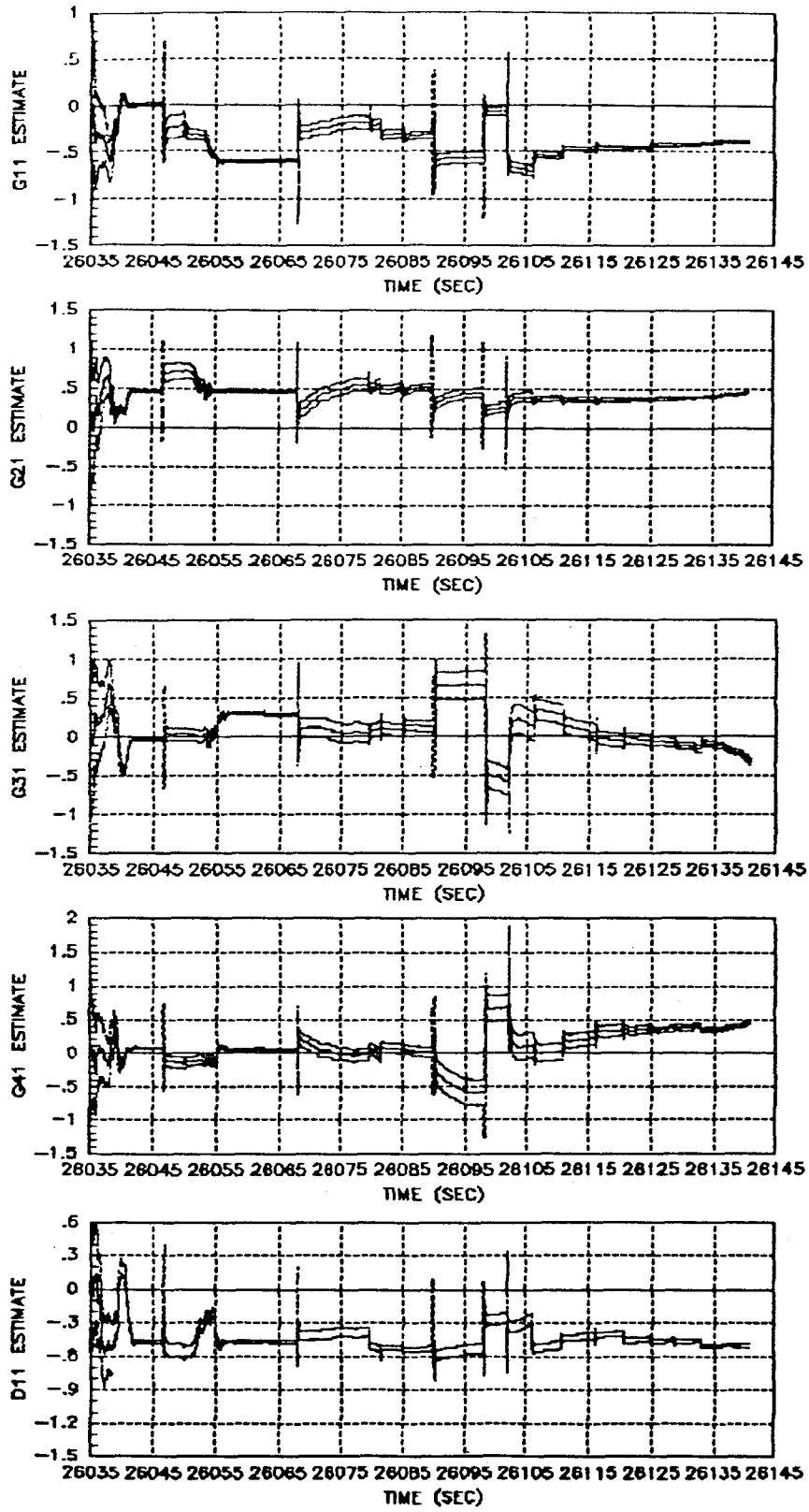


Figure 6-4: DAST 250 Hz G-matrix and H-matrix Parameter Estimates and Estimated Sigmas

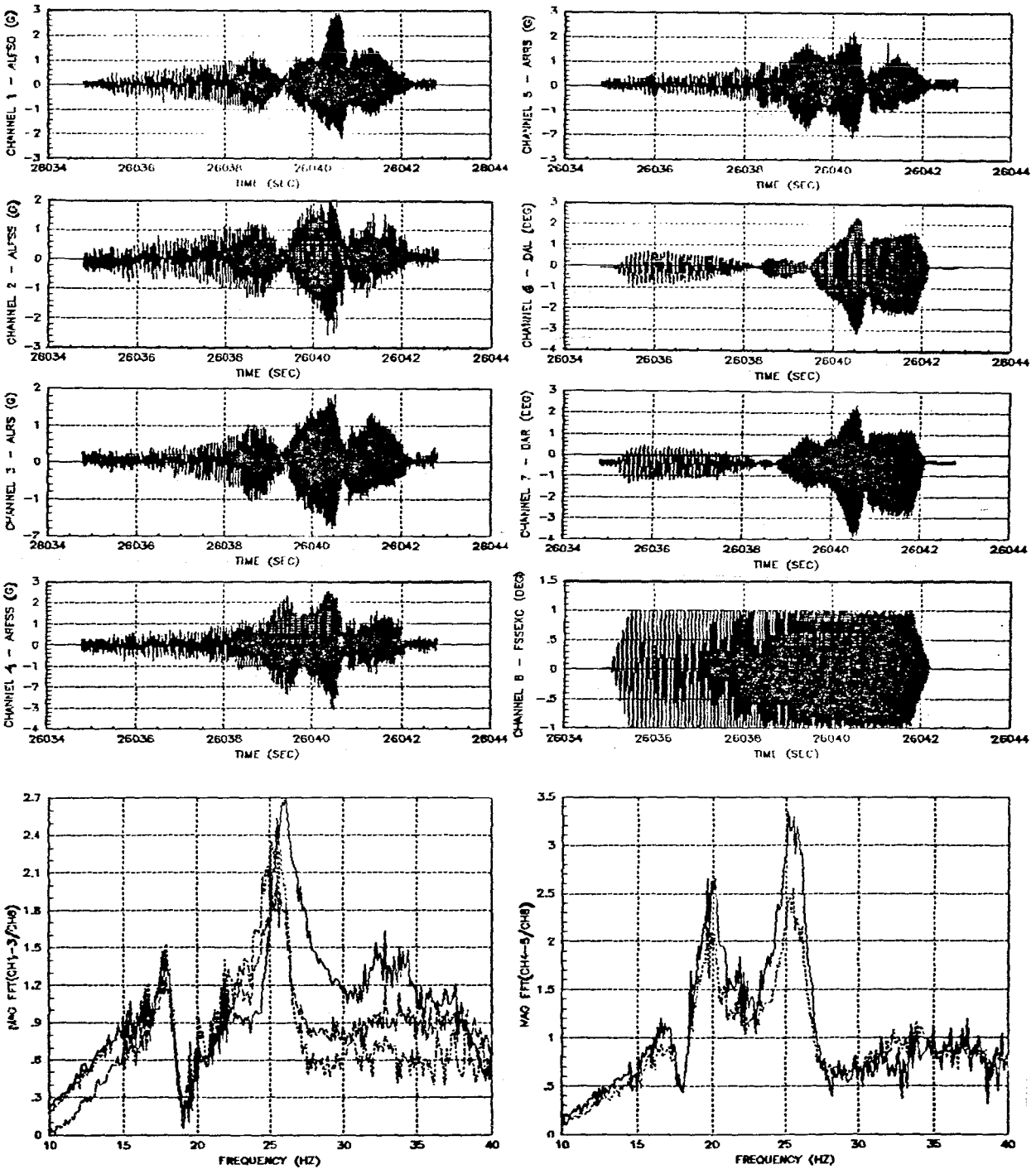


Figure 6-5: DAST Asymmetric Sweep FFT Transfer Function Estimates

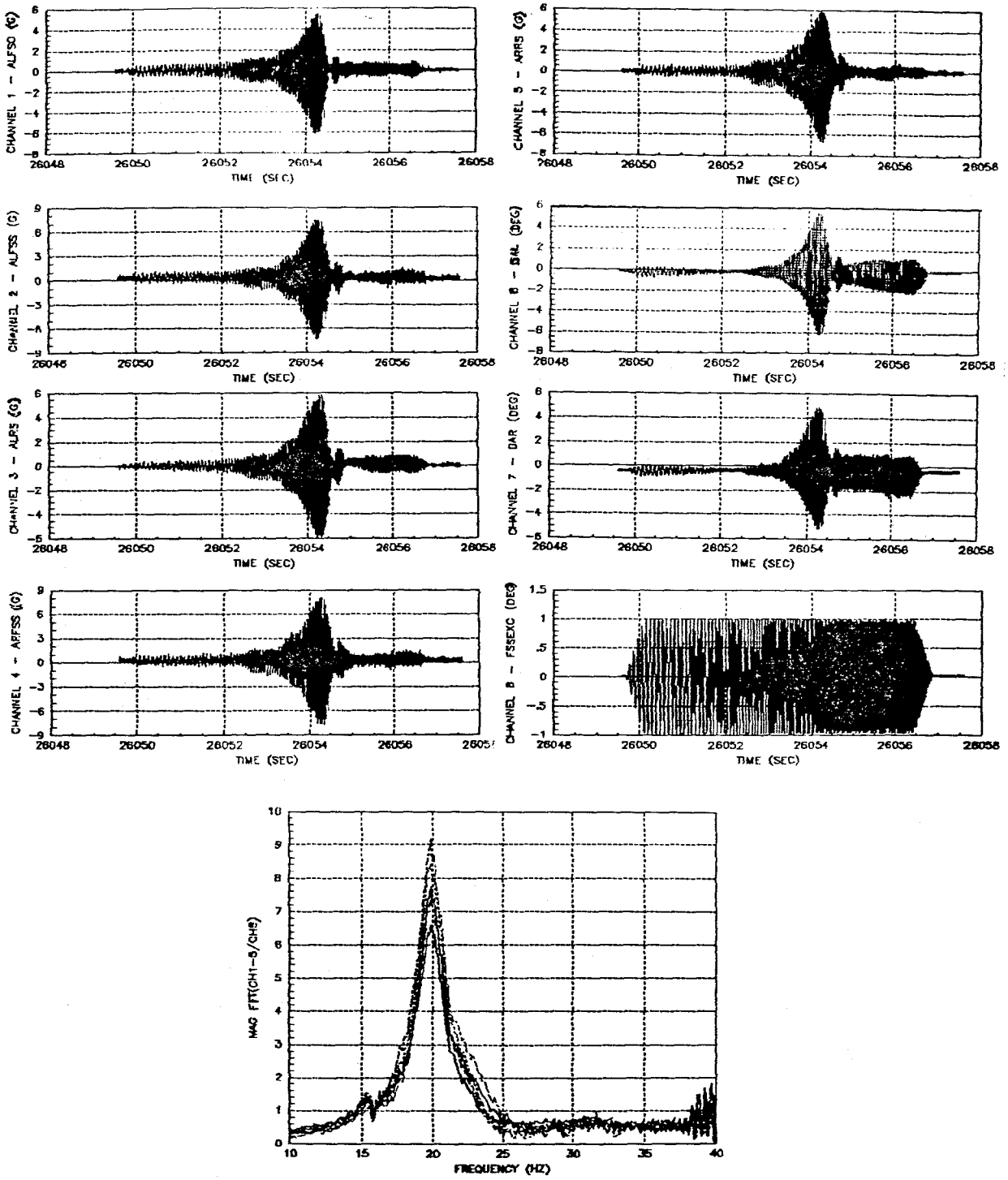


Figure 6-6: DAST Symmetric Sweep FFT Transfer Function Estimates

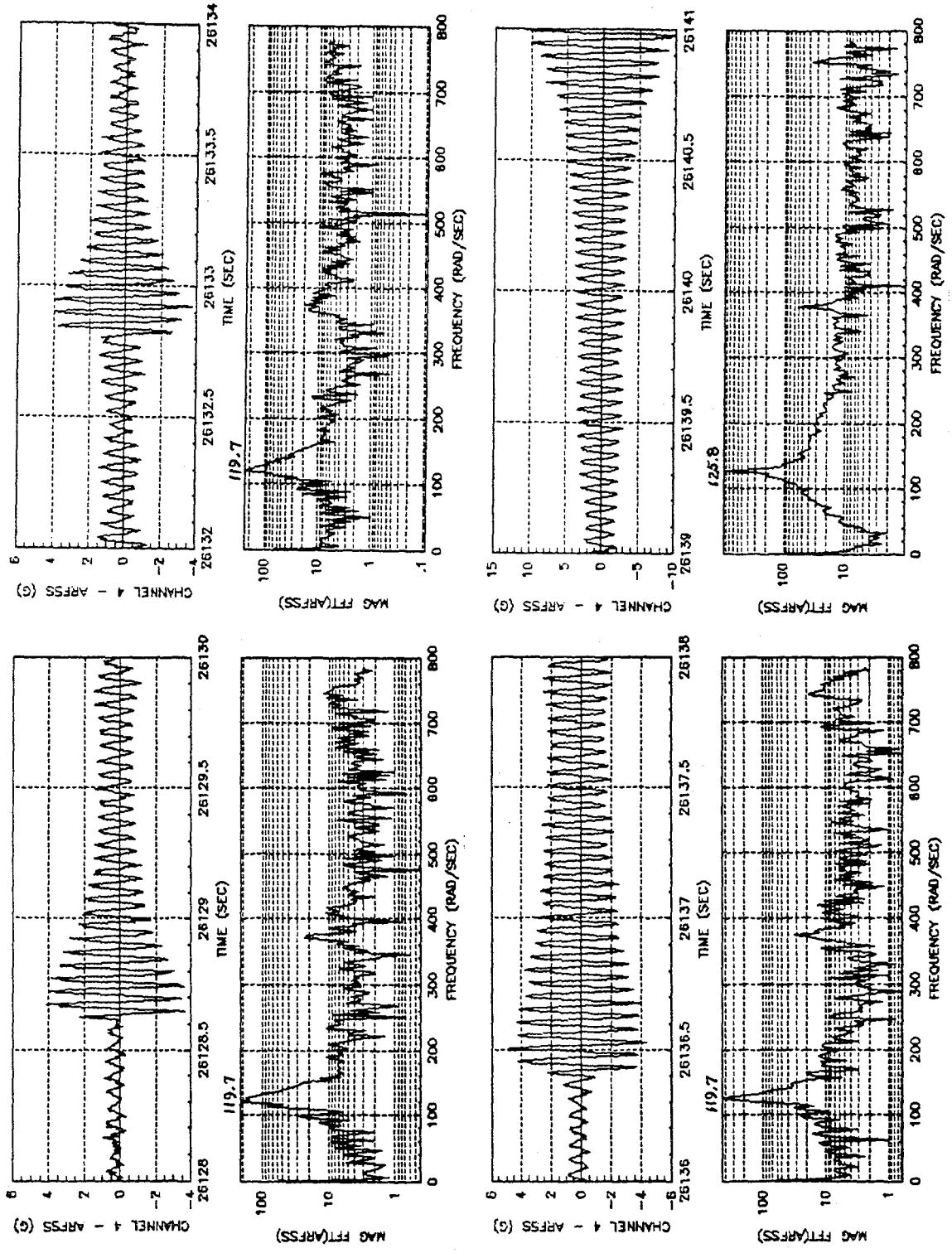


Figure 6-7: DAST 250 Hz Output Power Spectra During Last 15 Seconds of Flight

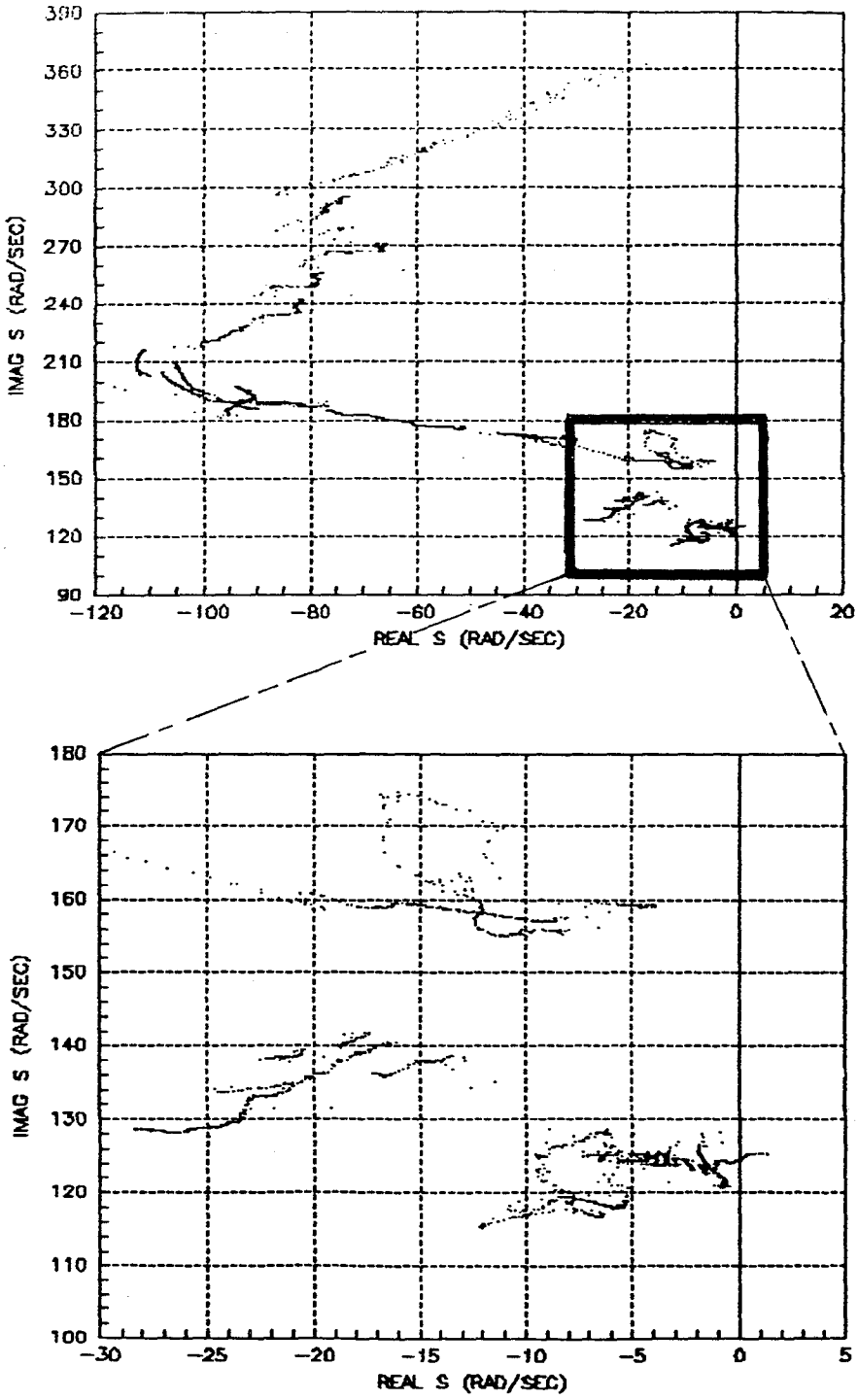


Figure 6-8: DAST 250 Hz Estimated Pole Location Time Histories

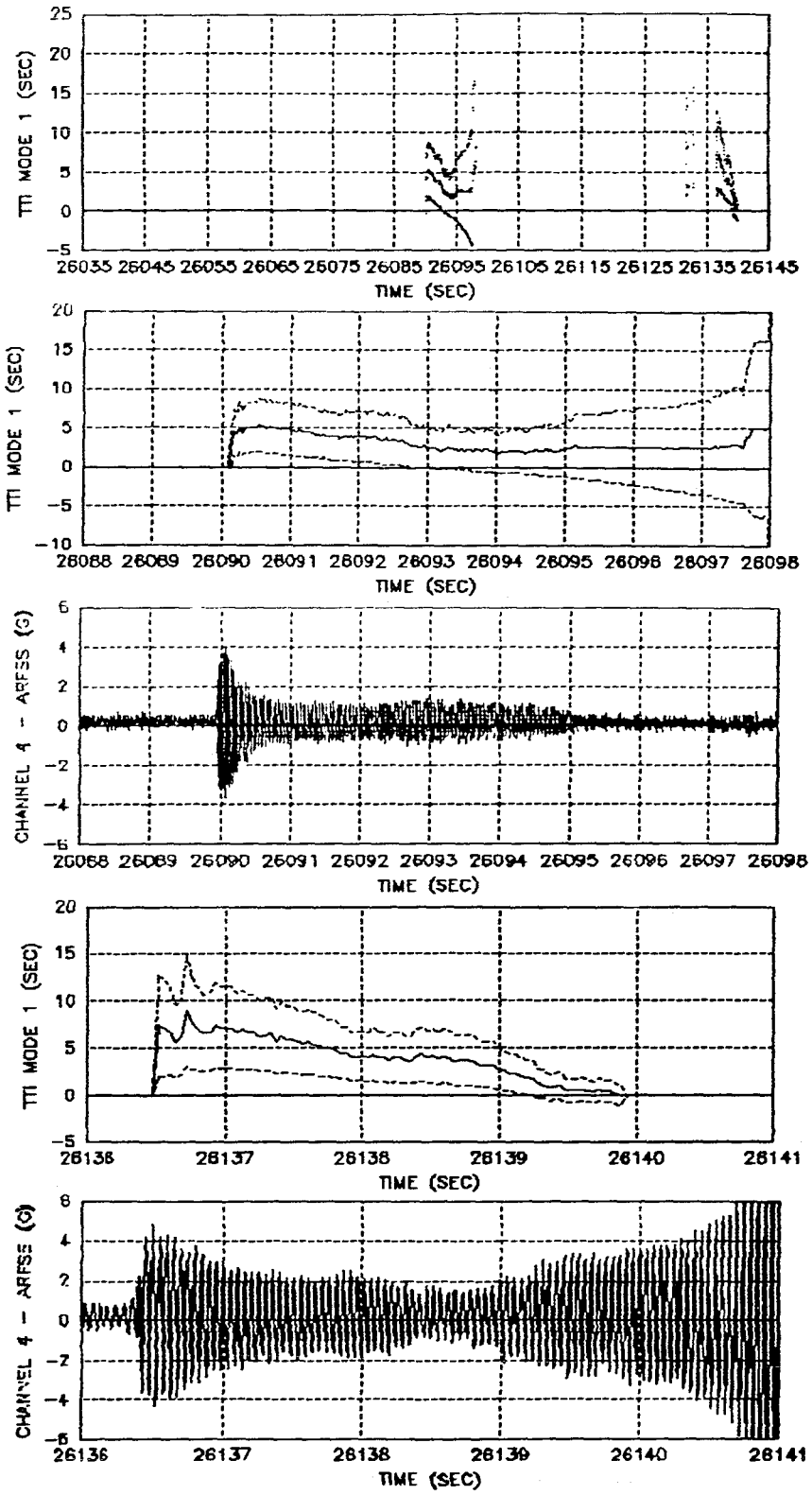


Figure 6-9: DAST 250 Hz TTI Estimates and Estimated Sigmas for Mode 1

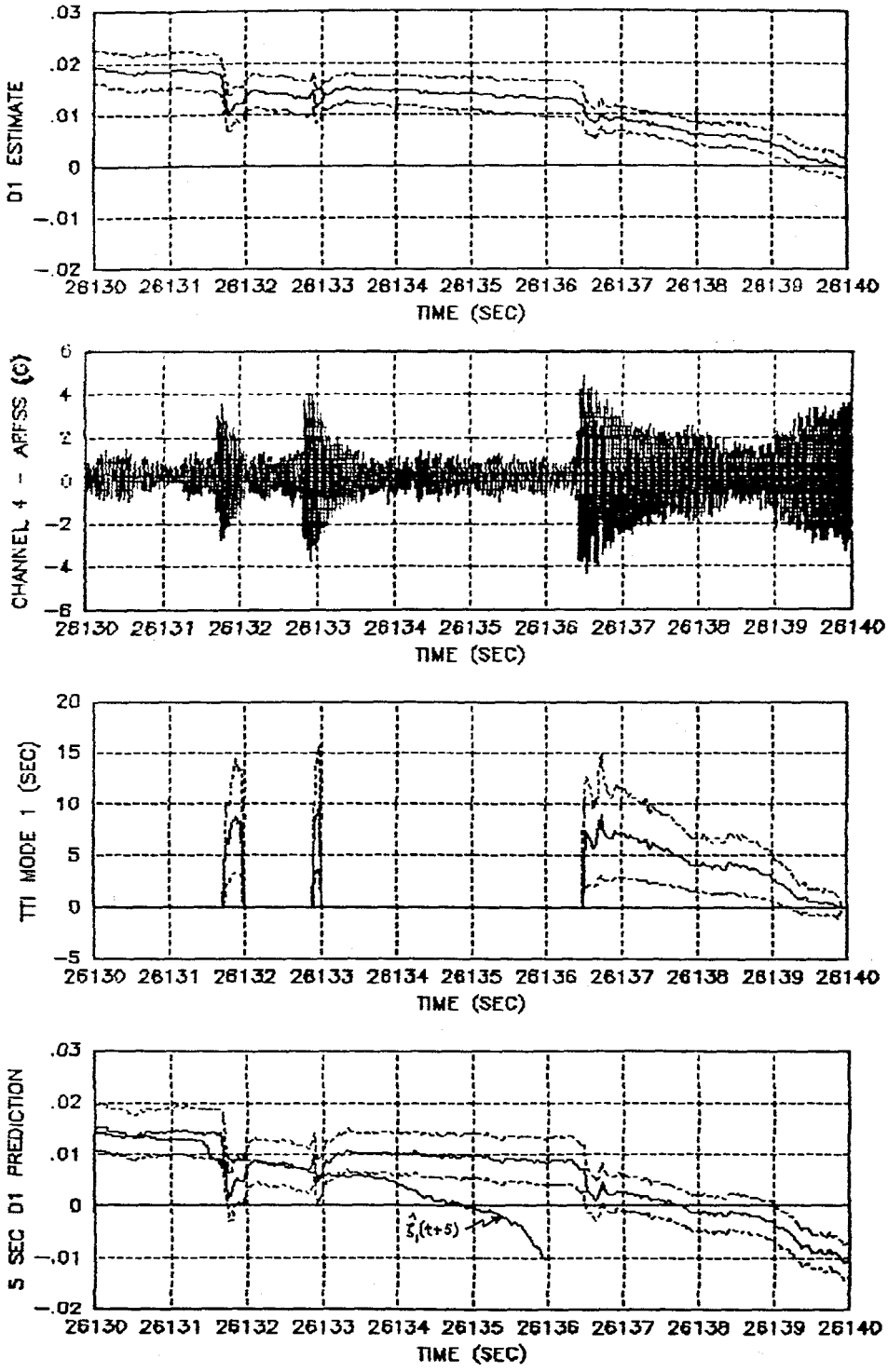


Figure 6-10: DAST 250 Hz TTI Estimates and Estimated Sigmas for Mode 1 Over Last 10 Seconds of Flight

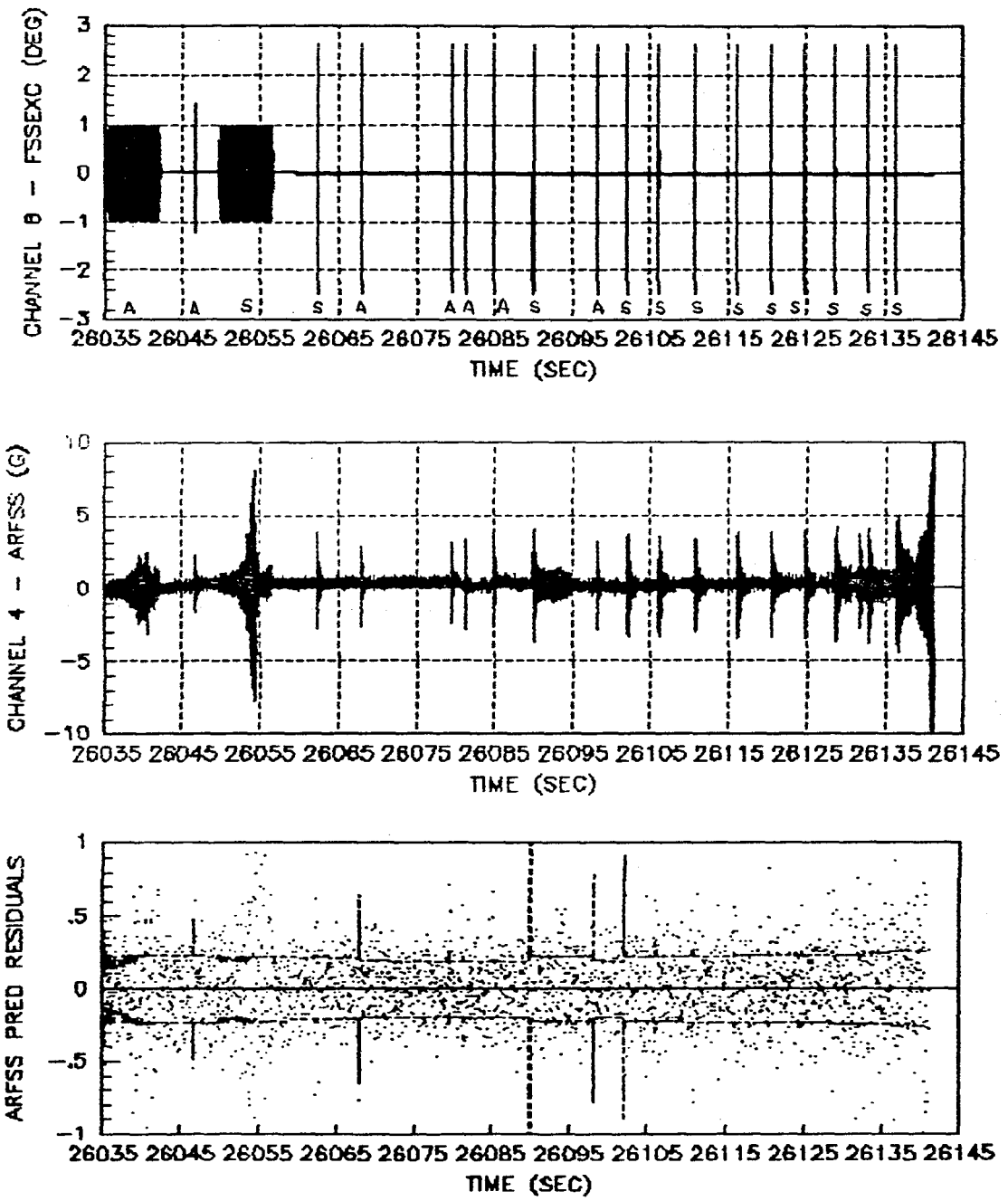


Figure 6-11: DAST 125 Hz Inputs, Outputs, and Predicted Data Residuals

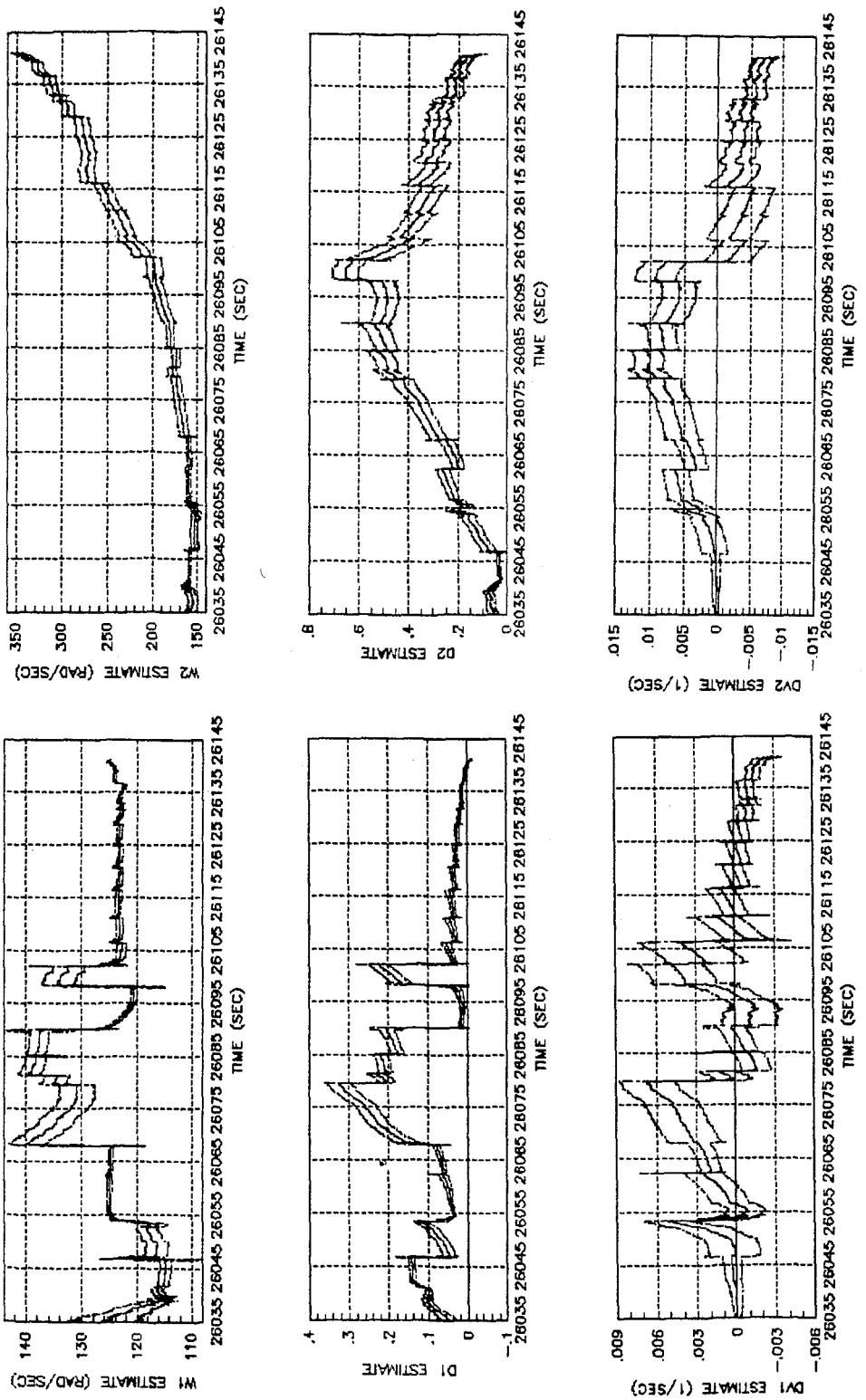


Figure 6-12: DAST 125 Hz Modal Parameter Estimates and Estimated Sigmas

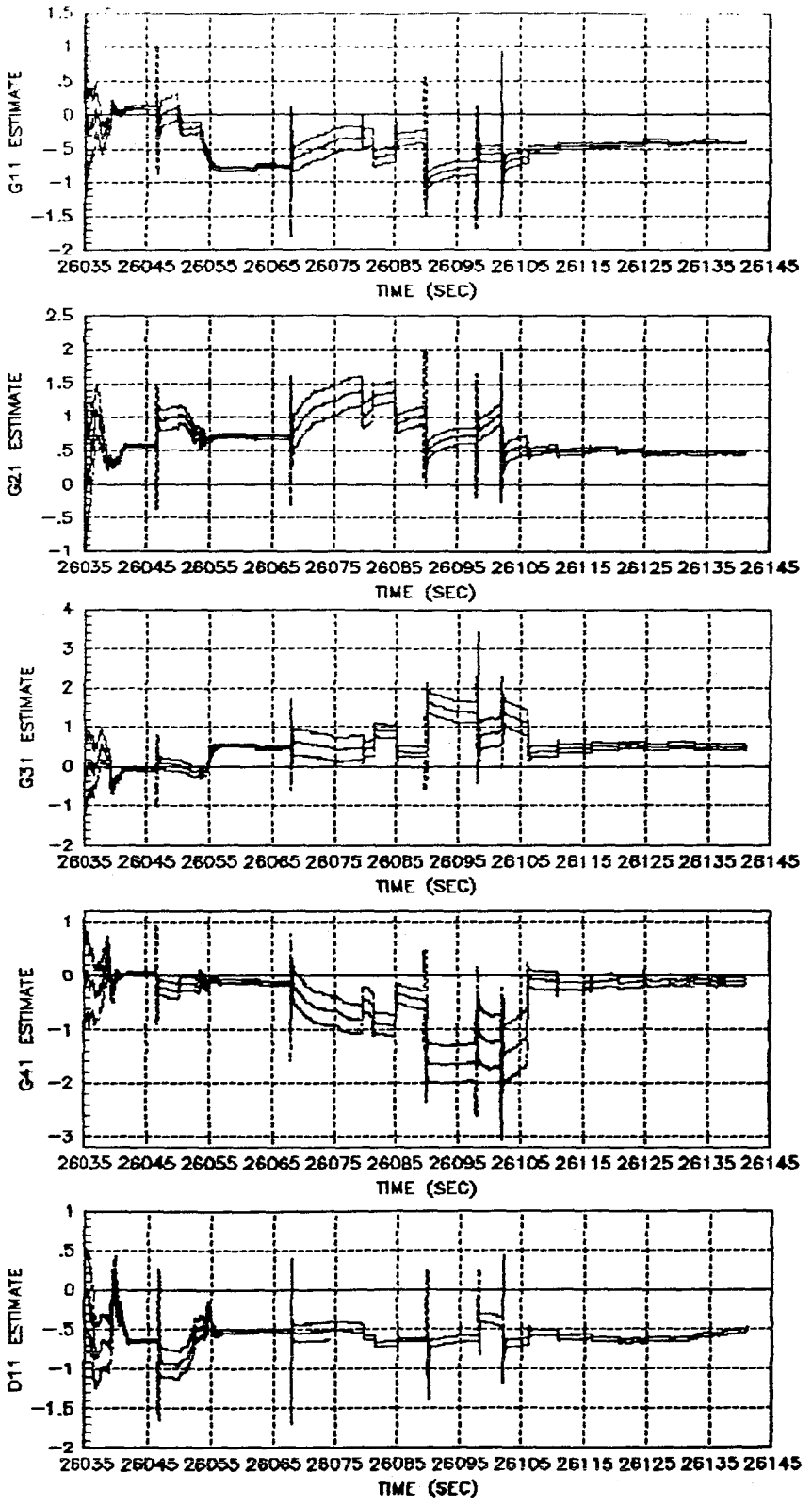


Figure 6-13: DAST 125 Hz G-matrix and H-matrix Parameter Estimates and Estimated Sigmas

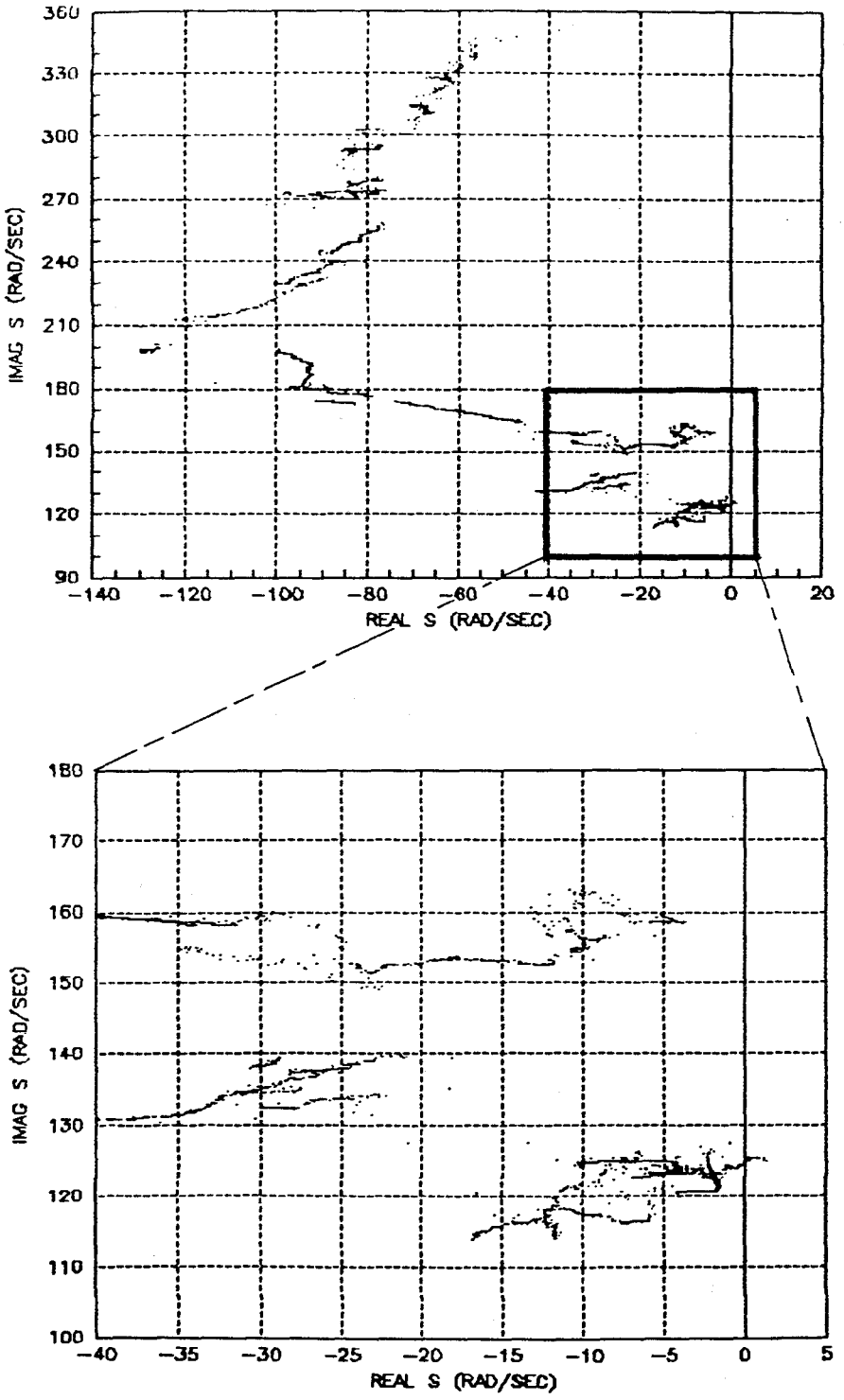


Figure 6-14: DAST 125 Hz Estimated Pole Location Time Histories

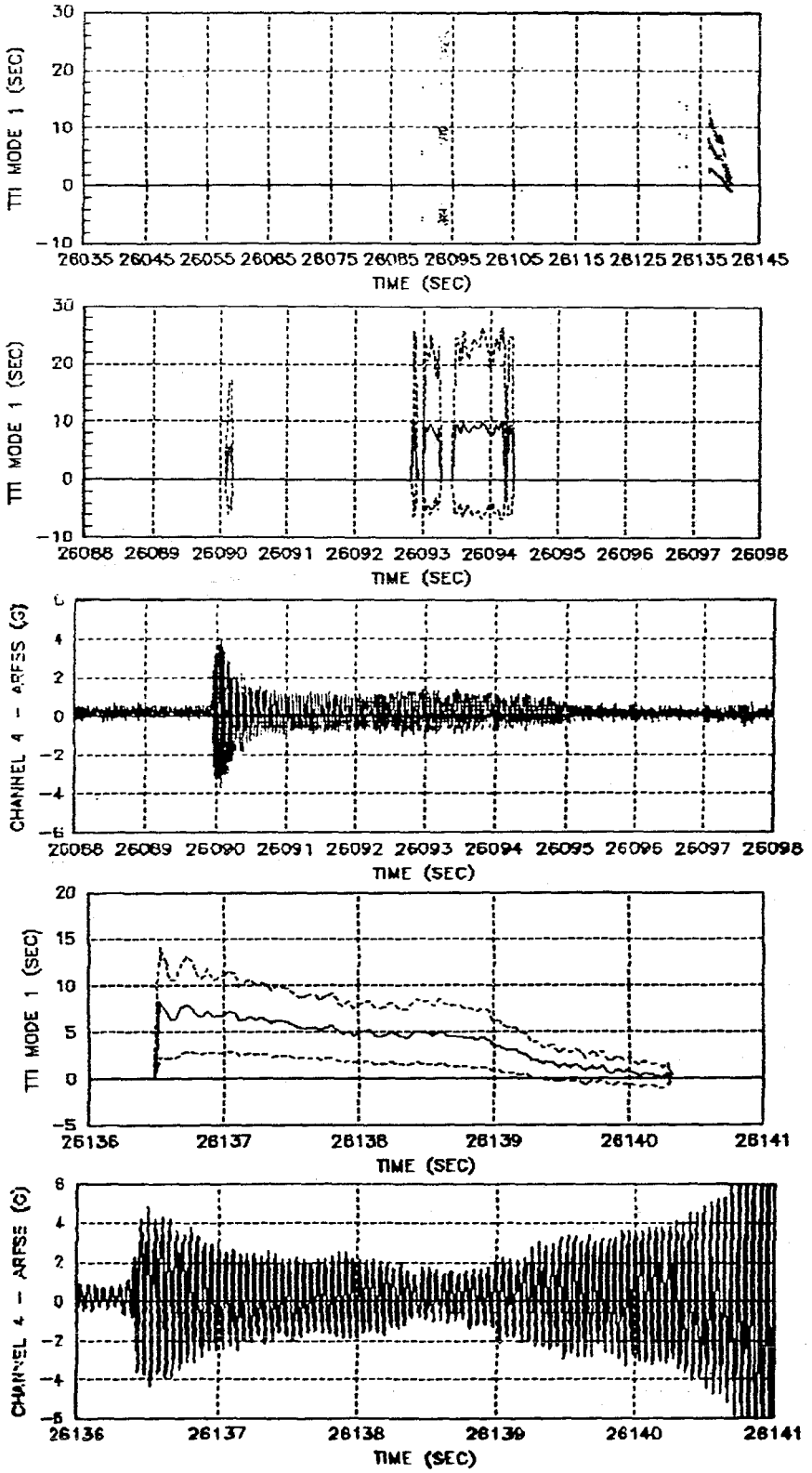


Figure 6-15: DAST 125 Hz TTI Estimates and Estimated Sigmas for Mode 1

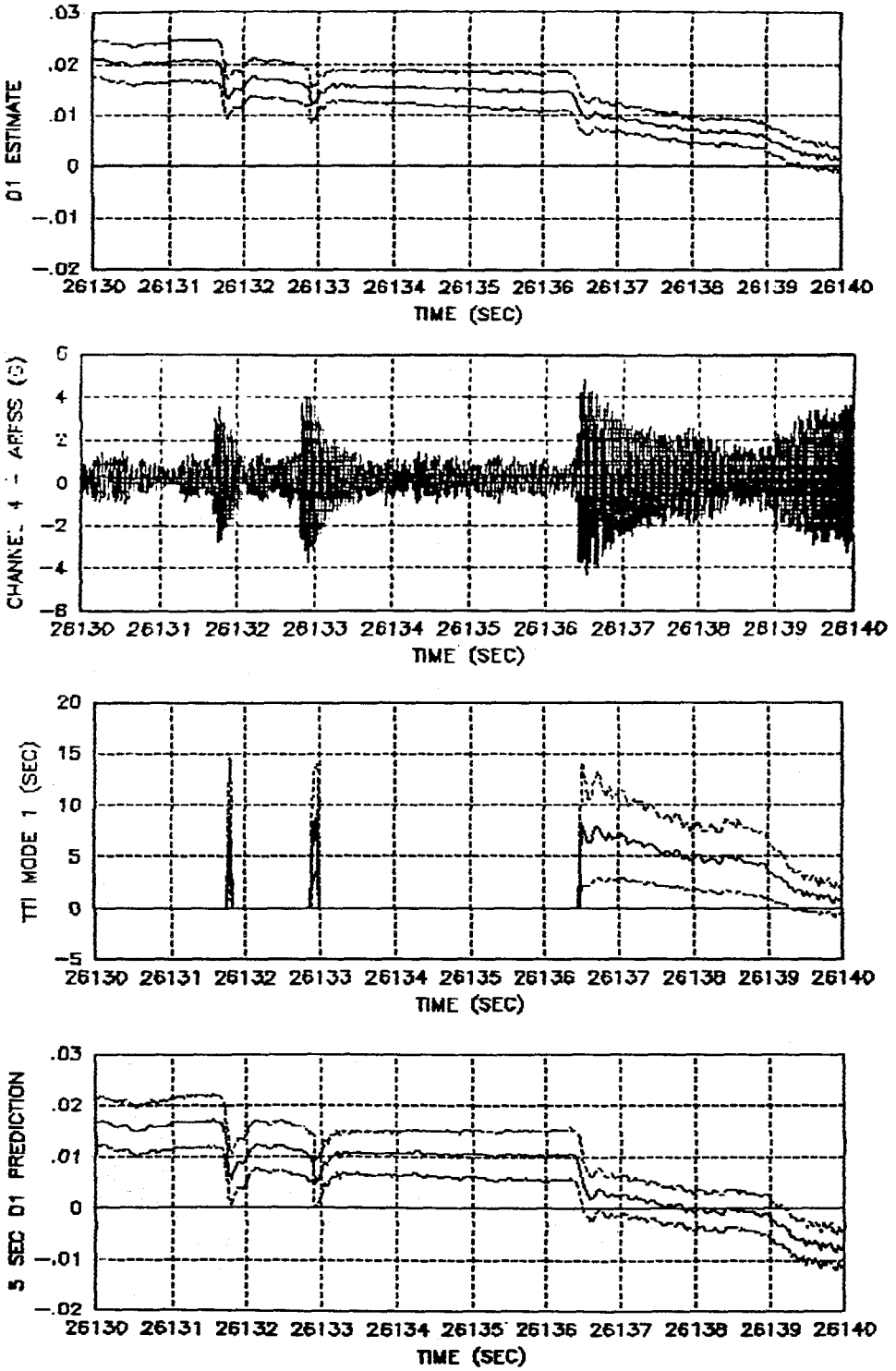


Figure 6-16: DAST 125 Hz TTI Estimates and Estimated Sigmas for Mode 1 Over Last 10 Seconds of Flight

6.2 F-16 FLIGHT TEST RESULTS

Background and Data Base

In addition to processing the DAST data which involved the processing of high quality (large SNR) data from a test designed with exogenous inputs to actively probe the aeroelastic response, turbulence excited F-16 data were also made available for processing. The objective of this analysis was to investigate the behavior of the EKF algorithm for flutter parameter identification in low SNR conditions where no exogenous inputs are present or available. As is discussed, identification of frequencies and damping coefficients under these conditions is quite different than in the presence of probing inputs.

The data available for processing included only three wing-mounted accelerometer outputs recorded during straight and level flight conditions. The channel identifications for these outputs are given in Table 6-2. The

TABLE 6-2: F-16 FLIGHT TEST CHANNEL IDENTIFICATIONS

Channel	Descriptor	Description
1	AS031	Left-wing forward normal accelerometer
2	AS032	Right-wing forward normal accelerometer
3	AS033	Left-wing rear normal accelerometer

data available were sampled at a 400 Hz rate and spanned a contiguous interval of approximately 90 seconds in duration (46486 to 46575 seconds). No control surface position measurements were available, and no other information concerning the flight conditions, aircraft configuration, or accelerometer location was available either. Under these circumstances, the somewhat unrealistic assumption that no control surface deflections were present had to be made.

Estimation Problem Formulation

Figure 6-17 shows plots of the entire available time histories for all three channels at the maximum available data rate (400 Hz). There are several interesting observations which can be made based on the time histories alone. First and most perplexing is the observation that the left-forward normal accelerometer (AS031) indicates a distinct change in 'static' normal specific force at 46498 seconds. Neither AS032 nor AS033 show a similar change. Since center-of-mass motion would be sensed to some degree in all three accelerometers, it is hypothesized that a change in the measurement device output such as a zero-level shift is the cause. Such occurrence,s however, make the high frequency output suspect as well.

A second observation concerning the data can be made by noting the relative amplitudes as a function of time. There is an apparent decrease in vibration level near 46540 seconds which continues for approximately 20 seconds indicative of a decrease in turbulence amplitude. However, the last 5 seconds of data give some indication of significantly larger amplitude vibrations than during the preceding minute. Based on information available a posteriori, significant control surface activity during this interval may have contributed to the increased acceleration, however this was not included in the analysis. Under the given conditions, the system identified was one with only random inputs and accelerometer outputs.

In order to obtain some information regarding the expected range of frequency estimates, some preliminary analysis was performed on the data. This was basically done in lieu of a priori information on the expected flutter frequency locations. Included in Figure 6-17 along with the time histories are FFT power spectra estimates for all three channels. Two frequency ranges are shown in order to provide better resolution at the low frequency range. The power spectra for AS031 and AS032 are quite similar and indicate that there are possibly two closely spaced modes at 4.5 and 5.5 Hz. There is also indication of a mode near 8 Hz, a frequency at which there is a mode clearly visible in the power spectrum of AS033. The very broad peaks in AS031 and AS032 near 20 and 40 Hz are of unknown origin, and are so heavily damped that identification of the parameters of these modes is not possible without exogenous inputs.

In view of the FFT results discussed above, a two-output, two-mode model (0,2,2) was chosen as the model to be identified. Since AS031 and AS032 had such similar spectra, only AS031 was used along with AS033 in the estimation. Two modes were chosen somewhat arbitrarily, and in light of the possibility of the presence of two closely spaced modes, gives some indication of the performance of the algorithm under such conditions (mismodeling). The high-pass (bias) filter was required due the the presence of low-frequency and DC offsets. Since the modal frequencies were expected to be on the order of 5 Hz, the bias filter pole was set to 0.2 Hz.

Since no exogenous inputs were present, the H-matrix elements had to be identified. As discussed earlier, zero initial conditions are not appropriate in these circumstances (with no exogenous inputs, the state estimates would remain zero!); thus the initial H-matrix was set to:

$$\hat{H}(0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Sigmas of unity in the appropriate units were chosen.

The initial frequencies were set to 4 Hz and 8 Hz with sigmas of 0.5 Hz. The initial damping coefficient estimates were set to 10% with sigmas of 2%. The initial damping coefficient velocity estimates were set to zero with sigmas 0.0001. Units normalization was performed in order to prevent potential numerical instabilities. The units of frequency were chosen as 10 rad/sec and the damping coefficient states were measured in 10% or 0.1 units. The oscillator velocity states were in 10 units and the corresponding H-matrix elements were estimated in units of 0.1 g/velocity state unit.

The dynamic state q's were set to 0.05 on all four states. As discussed in Section 3, this was reasonable in light of the auto-normalization capability added to prevent frequency estimates differing by factors larger than 1.2 or so from creating the need to tune the different modes with q's of different orders of magnitude. In the absence of any information concerning the flight conditions under which the data were collected and possible variations therein, moderately large q's were used for the frequency and damping coefficient velocity states, 0.01 and 0.0005 respectively. Though the results indicate that further time-varying tuning might have been reasonable due to some 'step-like' changes in the flight condition, none was performed. Finally, moderately large q's of 0.001 were

used on the H-matrix parameters (in the appropriate units) in order to adapt to possibly rapidly changing conditions.

Results

The results of the F-16 flight data analysis are presented in Figures 6-18 through 6-21. The accelerometer outputs after bias filtering are shown in Figure 6-18 along with the predicted data residuals and their theoretical sigmas. The effect of the bilinear nature of the estimation of H-matrix elements along with the dynamic states is clearly seen in the slow convergence of the sigmas to their steady-state values. The frequency and damping coefficient estimates are given in Figure 6-19. As far as the damping coefficient estimates are concerned, there is clearly a change in flight conditions during the interval 46535 to 46560 seconds. There also appears to be some change in conditions at 46520 seconds where the estimate of ω_2, having converged to about 8 Hz, starts drifting down to 7 Hz and the ζ_1 estimate also steps to a new value and begins drifting off.

Figure 6-20 shows an s-plane plot of the time histories of the estimated pole locations. Figure 6-21 gives the estimated H-matrix parameters and their estimated sigmas. The slow convergence of the velocity state associated parameters is evident and as discussed earlier is a manifestation of the nonlinear nature of the estimation problem coupled with the moderately large q's used to allow for parameter adaptation.

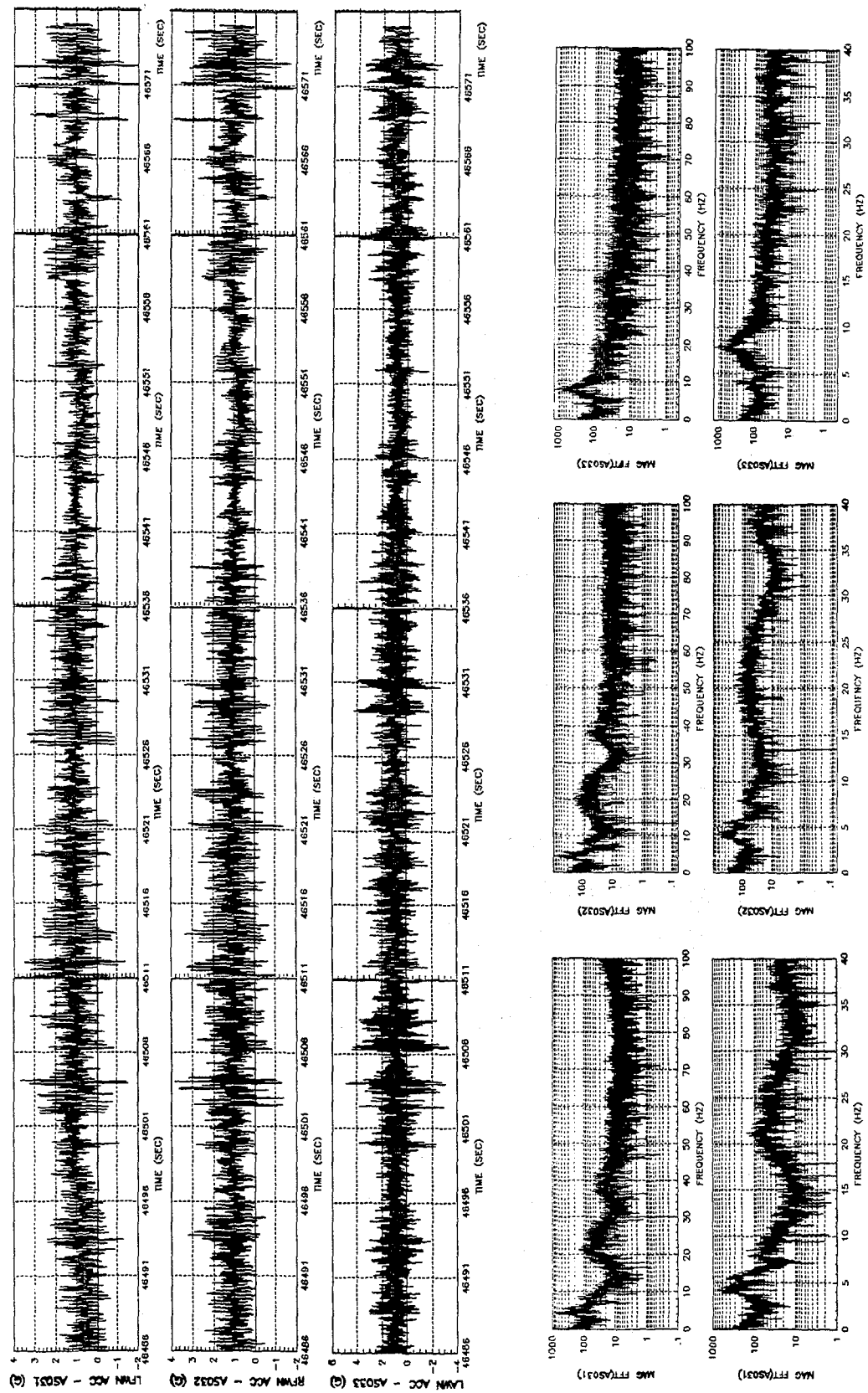


Figure 6-17: F-16 Accelerometer Time Histories and Power Spectra

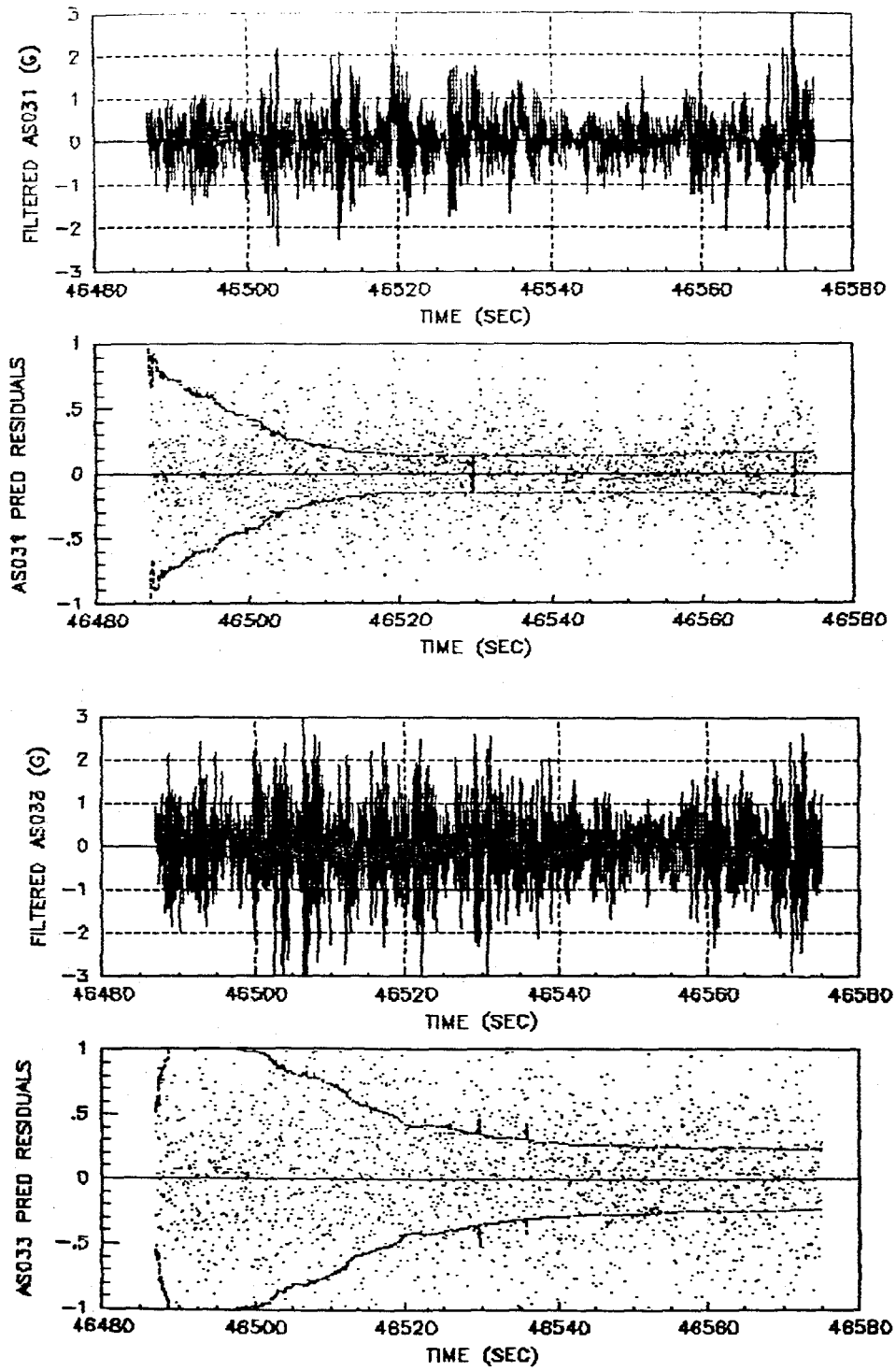


Figure 6-18: F-16 Outputs and Predicted Data Residuals

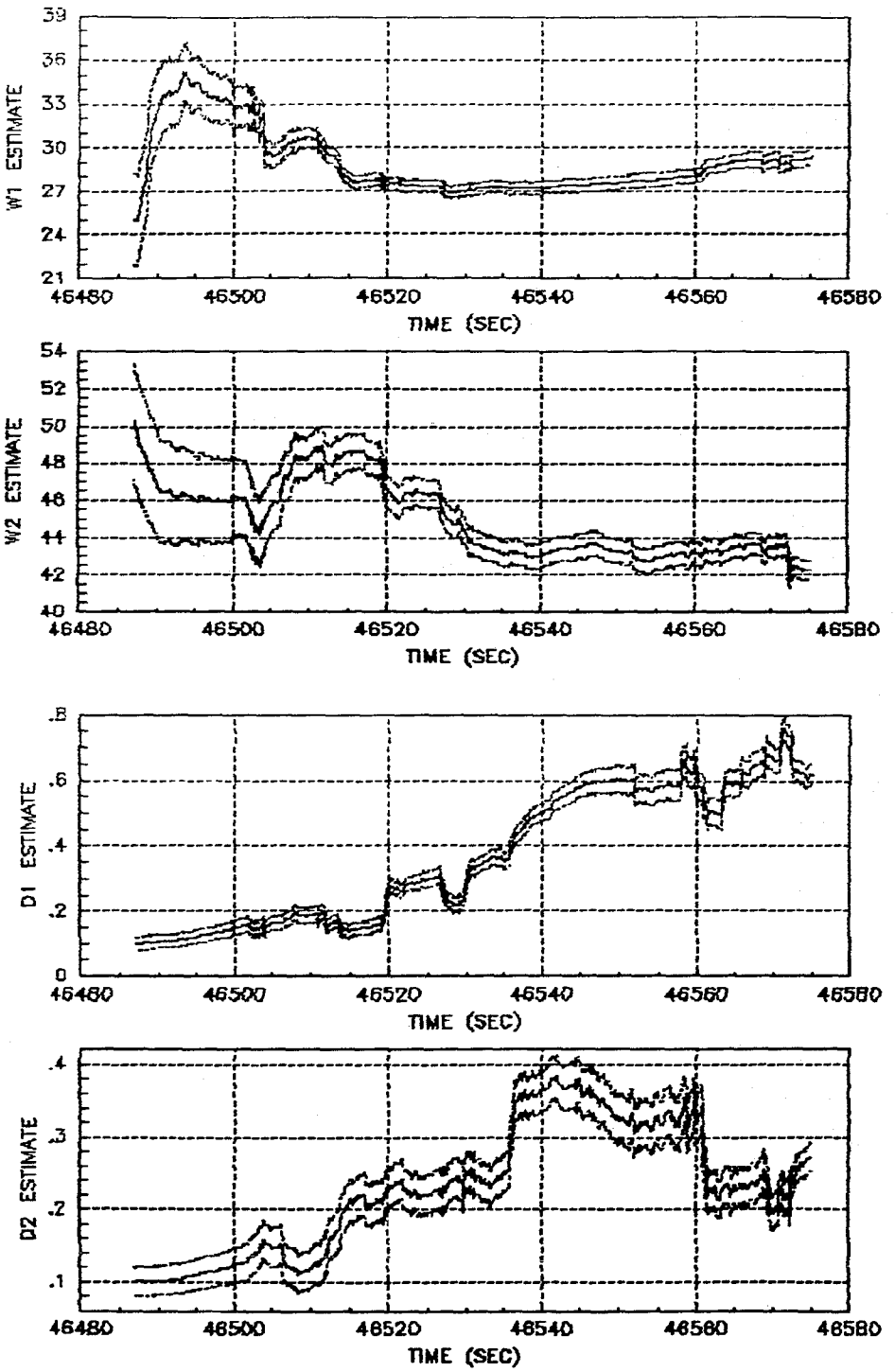


Figure 6-19: F-16 Modal Parameter Estimates and Estimated Sigmas

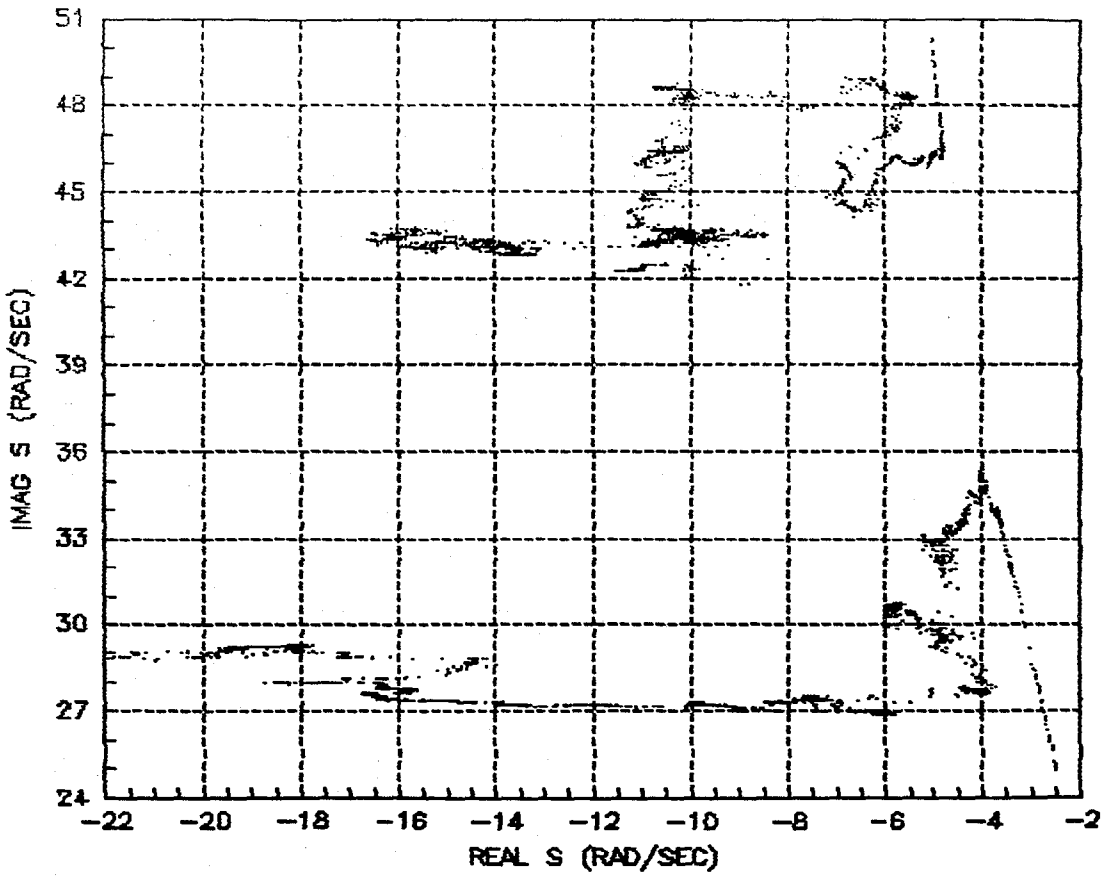


Figure 6-20: F-16 Time Histories of the Estimated Pole Locations

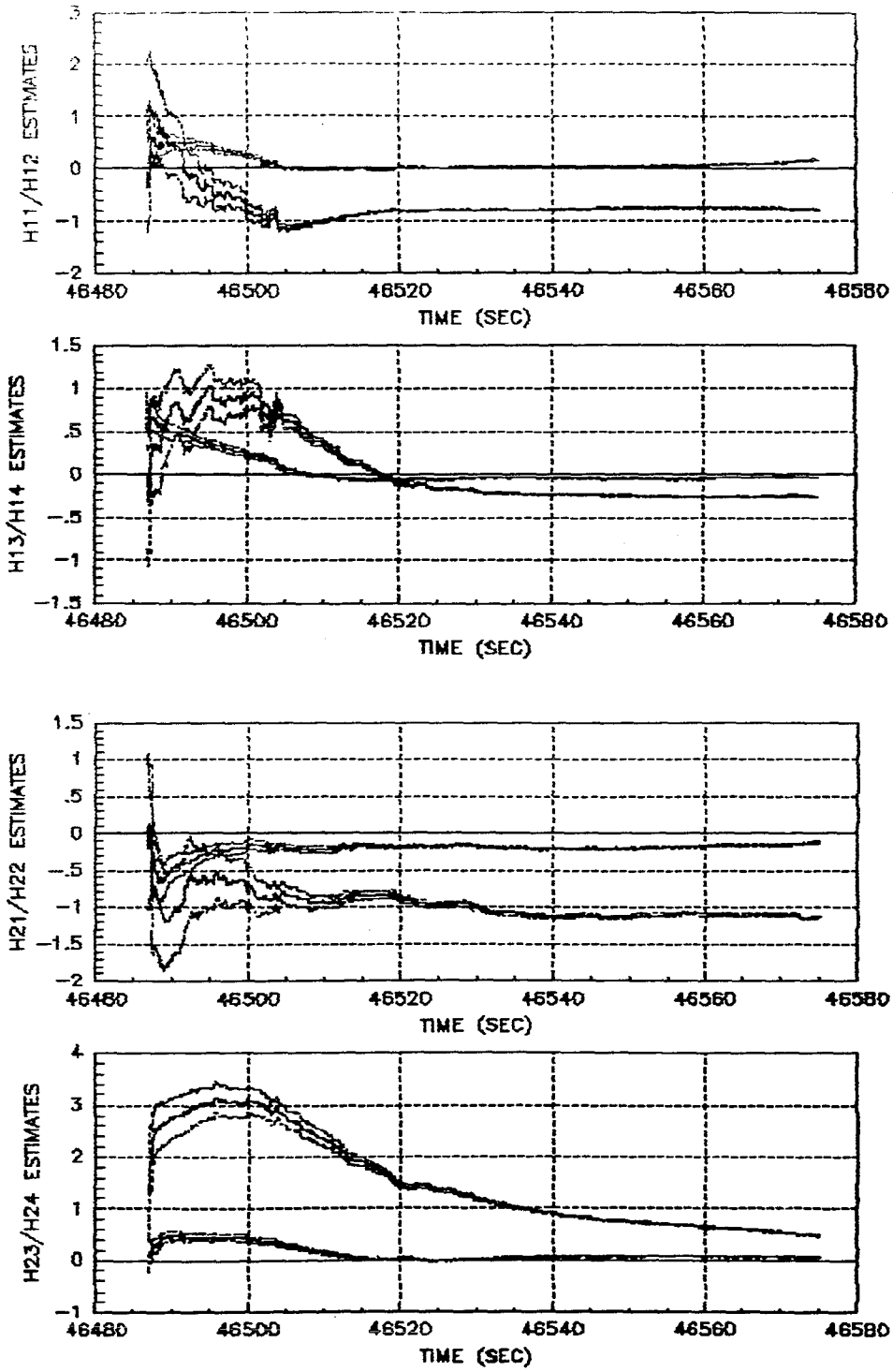


Figure 6-21: F-16 **H**-matrix Parameter Estimates and Estimated Sigmas

SECTION 7
PROGRAM DOCUMENTATION AND COMPUTATIONAL REQUIREMENTS

The purpose of this section is to provide an overview of the MOPID program organization and give some preliminary results on operation counts for various estimation problem configurations. The discussions are of a general nature; this is not intended to be a programmer's guide. Detailed descriptions of the program inputs can be found in Appendix A. The program described is MOPID Version 0.9 dated December 26, 1984.

7.1 MOPID PROGRAM OVERVIEW

The MOPID computer program is a FORTRAN-77 implementation of an extended Kalman filter designed primarily for aeroelastic flutter parameter identification in a real-time environment. The majority of the code is ANSI standard FORTRAN-77, however, in its current state, the inputs to the program are through the DEC supplied NAMELIST extension to FORTRAN-77. The major design consideration was computational efficiency. While substantial efforts were undertaken to minimize the amount of computation, no attempt was made to minimize storage requirements.

7.1.1 Storage Requirements

The EKF algorithm is recursive in nature, and there is currently no requirement to save the past information stored in the state estimates and covariances for further processing. Thus, the estimates and their associated estimated sigmas (square-roots of the diagonal elements of the filtered covariance matrix) as well as the data, residuals (predicted and filtered if calculated), and their sigmas are stored in arrays for plotting only. The calculation of the sigmas (a dot product is required for each sigma since the covariance is propagated in square-root form) and subsequent storing of the estimates occurs at a user selectible interval. The maximum number of time points which can currently be saved is 5000; the program currently detects plotting array overflow and stops when the arrays are full.

Thus, it is necessary that a plot output frequency (cf. IPLFQ) be chosen which guarantees that the arrays will not overflow if the entire amount of data specified (cf. WINDOW) is to be processed.

The maximum number of parameter estimates and sigmas which can be stored each time is 48, and the maximum number of data values (both system inputs and outputs), residuals, and sigmas which can be stored is 34. In the simulation mode, simulated modal parameter trajectories can be saved and their are a maximum of 12 allowed (4 modes and 3 parameters per mode). Finally, up to a maximum of 15 modal parameter correlations can be saved for plotting as well. In each of the arrays for plotting, the first two columns are absolute time and relative time respectively; thus the plotting array storage requirements are:

No. of single-precision words = 5000 x (50 + 36 + 14 + 17) = 585k.

These requirements can be easily altered by any of several techniques. The array limit of 5000 could be simply reduced at the expense of reducing the output data rate. To eliminate the storage requirements altogether while providing a potentially unlimited output data rate (bounded of course by the input data rate), file access in the inner loop could be performed. This approach was not used in the current implementation since file access (reads and writes) are potentially much slower than array access.

The algorithm storage requirements are modest in comparison to those for output plotting. They are governed by the maximum state dimension which is currently 50. With the maximum number of system inputs and outputs set to 4 and 6 respectively, the amount of storage currently allocated to algorithm related quantities is approximately 14000 single-precision words. This requirement could easily be halved, if necessary, with dynamic storage allocation programming methods.

7.1.2 Input Data File Structure

The only file access in the inner loop of the algorithm occurs when reading input data. The data input to the program are assumed to be stored in a binary file (named EKFINP and opened in read-only mode) written with FORTRAN 'unformatted' writes. All records are exactly the same format;

time, data(1),data(2), ... , data(n). See the entries NINPS and NOUTPS in Table A-1 in Appendix A for further details. These records are read into the algorithm sequentially and windowed to determine whether or not to process the data (cf. WINDOW). This file can be as large as the operating system allows.

7.1.3 Output File Structure

Currently, the only files output by MOPID are the plot files. Potentially, four (4) files can be written: EKFDAT, EKFPEST, EKFPCOR, and EKFSIM. These files are written by the subroutine SAVLOD, and are in MATRIX_x format. Currently, the output is only plotted by the MATRIX_x program [16], however SAVLOD could be modified to format the data for input into any available plot program. Note the plot files are limited in size by the size of the plot arrays.

The data are stored in the arrays 'columnwise'. The first two columns in each file are time; the first column being absolute time (i.e. the time associated with the input data vector), and the second column being time relative to the user specified input T0 (cf. T0). The data stored in EKFDAT are in the following order:

$$[T(I),T(I)-T0,u_1(I),\ldots,u_{NU}(I),z_1(I),\nu_1(I),\sigma_{\nu_1}(I),\varepsilon_1(I),\sigma_{\varepsilon_1}(I),z_2(I),\ \ldots\ ]$$

where u_i is the i-th element of the system input vector, z_i is the i-th element of the system output vector whose associated predicted and filtered residuals are ν_i and ε_i respectively. Elements associated with a particular system input datum which has been rejected as an outlier (in RESCHK) are all currently set to zero so as not to create plot scaling problems in MATRIX_x. A similar structure is used for the file EKFPEST in which the parameter estimates, TTI estimates, and their sigmas are stored:

$$[T,T-T0,\hat{\zeta}_1,\hat{\sigma}_{\zeta_1},\hat{\zeta}_{v_1},\hat{\sigma}_{\sigma_{v_1}},\hat{\omega}_1,\hat{\sigma}_{\omega_1},\ldots,\hat{g}_{11},\hat{\sigma}_{g_{11}},\ldots,\hat{h}_{11},\hat{\sigma}_{h_{11}},\ldots,\hat{d}_{11},\hat{\sigma}_{d_{11}},\ldots,$$
$$\hat{TTI}_1,\hat{\sigma}_{TTT_1},\ldots,\hat{TTI}_n,\hat{\sigma}_{TTI_n}].$$

The index I has been dropped for convenience. There are as many sets of frequency and damping coefficient parameters as there are modes being estimated. The g's are stored columnwise, and the h's and d's rowwise. Constrained elements in any of the system matrices are not saved for plotting in order to reduce the storage required (they are constants). The file EKFSIM has a structure similar to the first columns of the EKFPEST file, but no sigmas are present:

$$[T, T-TO, \zeta_1, \zeta_{V_1}, \omega_1, \zeta_2, \zeta_{V_2}, \omega_2, \ldots].$$

The file EKFPCOR contains the correlations of the modal parameters for the first two modes only. Since the correlation matrix is symmetric, only the lower triangular portion is saved. The storing of the elements is done columnwise. Thus, assuming two modes were being identified, the correlations of ζ_1 with all five other parameters would be stored first, followed by the correlations of ζ_{V_1} with the remaining four parameters, etc. This file is written only if requested by the user (cf. IPLCOR).

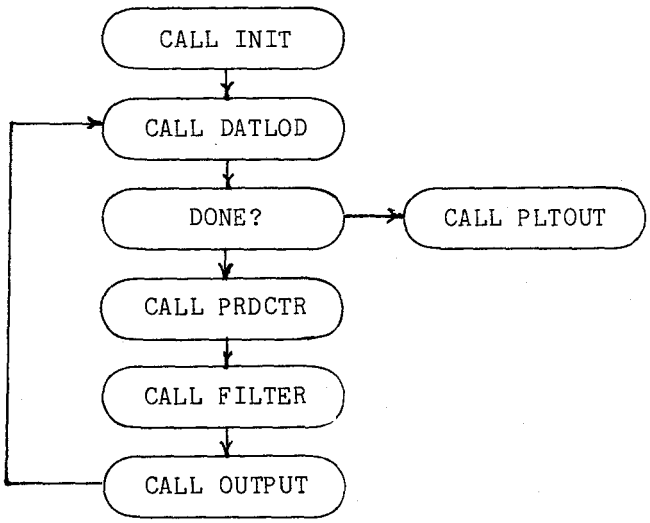
7.1.4 Program Flow

The program flow is dictated by the recursive nature of the EKF algorithm, and the input-output access requirements. NAMELIST input (namelist name &INPUT) is used to initialize the algorithm (cf. Appendix A). The namelist is read in INIT after ZERO is called to initialize various arrays and pointers. INISIM and SUMOUT are routines which printout initialization summary information if requested. Once various consistency checks are performed on the algorithm initialization, the inner loop is entered.

Unless internally generated simulated data are requested, the input data file is recursively read until a data time within the user specified WINDOW is encountered. At this point, data processing begins. The diagram on the following page illustrates the flow in the inner loop. Though the details of how the operations are performed are highly problem specific, the flow indicated in the diagram is that of a very general continuous-discrete recursive filtering problem. Once the time associated with the input data exceeds the maximum specified by WINDOW(2), PLTOUT is called and the program

stops. At the current time, the only operator interaction is through the input namelist. No interactive capability is present. The final step, of course, is to plot the results. This is currently done using the MATRIX$_x$ program.

DATLOD's function is to get the data (from GETSIM or GETDAT) and if requested pass them through the bias rejection filters (BFILTR). PRDCTR calls FLOAD to calculate the appropriate system matrices and partials and predicts the state and estimate error covariance (PHOUSE). PRDCTR then calls PRES to compute the predicted data residuals and their theoretical sigma. RESCHK performs a threshold test on the residuals and deletes the datum from the input data vector if the normalized residual exceeds the threshold (THRSH). FILTER then performs the filter step by calling FHOUSE.



A call to OUTPUT concludes the processing in the inner loop, outputting summary information to the standard output unit (IOUNIT) and writing data into the plot arrays if requested (by various user selectible output frequency and option flags described in detail in Appendix A.)

7.2 MOPID COMPUTATIONAL REQUIREMENTS

In an effort to accurately assess the computational requirements of the MOPID program's EKF implementation, an input flag (IOPCNT) was used to enable counting of floating-point operations in the computation intensive routines. The numbers presented below are the numbers output by the program under the various problem configurations stated. They are not intended to be an exact count, nor are operations such as memory fetches and indexing calculations included and for these reasons could very well be rounded up to the nearest 100 counts. They are included with the resolution indicated solely for comparison with similar results from other installations using the same code.

The majority of the computations performed by the algorithm occur in the two routines PHOUSE and FHOUSE which perform the Householder triangularization of the appropriate matrices corresponding to the prediction and filtering steps of the algorithm respectively (cf. Section 3.3). Since maximum advantage was taken of the structure of the partial matrices and vectors, the operations required to calculate covariance matrix products with the **F**- and **H**-matrices were minimized. In all cases, their total represented less than 10% of the overall operation counts so these counts are not included.

One floating-point operation (FLOP) consisted of a single-precision floating-point multiply and add. The few divisions and square-roots required were counted as a single FLOP. The major factor in determining the number of FLOPS is the estimation problem formulation since that determines the number of elements in the state vector. The number of operations in the Householder routines is roughly proportional to the cube of the state vector dimension! Though there are significant computational savings realized in the case where **G**-matrix elements are being identified and the system inputs are zero over extended periods (several data points), for the purposes of these discussions, the inputs are assumed to be non-zero in all cases (which incidently is a condition for improved parameter identifiability as discussed in Sections 5 and 6).

The state vector being estimated in the EKF algorithm is composed of dynamic states, flutter modal parameters, G-matrix elements being identified, followed by H-matrix and D-matrix elements if any. Thus, the notation used below indicates the number of elements in each of these groups, in order. For example, the notation (4,6,4,0,1) indicates a state vector with 4 dynamic states (i.e. 2 modes), 6 modal parameters (frequencies damping coefficients, and damping coefficient velocity states), 4 G-matrix elements, no H-matrix elements, and 1 D-matrix element.

TABLE 7-1: OPERATION COUNTS FOR VARIOUS ESTIMATION PROBLEM CONFIGURATIONS

Configuration (x,ζω,g,h,d)	Subroutine			TOTAL
	PHOUSE	FHOUSE	FLOAD	
(2,3,2,0,0)	321	371	255	947
(2,3,2,0,1)	424	524	255	1203
(2,3,0,2,0)	414	106	265	785
(2,3,0,2,1)	539	524	265	1328
(4,6,4,0,0)	3022	494	530	4046
(4,6,4,0,1)	3466	2855	530	6851
(4,6,0,4,0)	2268	2359	510	5137
(4,6,0,4,1)	2584	2855	510	5949
(4,6,4,4,0)	5004	4749	754	10507
(4,6,4,4,2)	6221	6390	754	13365
(4,6,8,0,0)	6270	710	774	7754
(4,6,8,0,2)	7735	6390	774	14899

As expected, the computational load increases dramatically with the number of modes being estimated. Assuming a computation speed of 1 μsec/FLOP (approximately the SEL speed), the (4,6,4,0,0) problem can be run at approximately a 200 Hz throughput rate. This corresponds to the 1-input, 1-output, 2-mode case with no direct feedthrough term being estimated.

Interestingly, the (4,6,4,0,1) problem requires approximately 70% more computation than the (4,6,4,0,0) problem. Furthermore, the trend is the same for all configuration pairs (with and without D-matrix elements being identified). There is a significant increase in the number of FLOPS performed (mostly in FHOUSE) with direct feedthrough terms being identified.

The reason for this increase in computational burden with the addition of D-matrix elements has to do with the structure of the matrices on which the Householder triangularizations are performed. The code was originally optimized in the sense of requiring minimal computational effort for various problem configurations, none of which included direct feedthrough identification; so the D-matrix elements were placed at the 'end' of the state vector. Subsequent analysis of the DAST data (cf. Section 6) indicated the need for D-matrix element identification. The computational load can be decreased significantly by optimizing the code for these configurations, however a significant amount of programming effort is required.

SECTION 8
SUMMARY OF RESULTS AND CONCLUSIONS

This analysis and development effort described in this report has shown that it is possible to obtain reliable estimates of flutter parameters in a real-time environment using an approximate model of the system dynamics and an EKF algorithm for dynamic state estimation and model parameter identification. Using linear oscillator models to approximate aeroelastic dynamics over short time intervals, measurements of the system inputs and outputs were used in an EKF algorithm to obtain significant estimates of future system stability in both simulated and actual flight test data.

Processing simulated data for test cases with and without exogenous inputs was successfully performed. The results indicate that:

1) With exogenous inputs, the algorithm is capable of tracking closely spaced, rapidly time-varying modes, providing low-variance estimates of the frequencies and damping coefficients of the modes as functions of time.

2) Without exogenous inputs, the algorithm can still provide low-variance estimates of slowly varying modes in the presence of light turbulence. Increasing the turbulence to measurement noise power ratio will only improve the algorithm's capability to track faster varying modes with smaller variance.

3) Without exogenous inputs, damping coefficient relative error is significantly larger than the associated frequency estimate error due primarily to the lack of relative phase information. Exogenous inputs improve damping coefficient estimates and their variances dramatically!

From these simulated data results, some preliminary conclusions can be drawn. More extensive simulated data analysis is required in order to further quantify the indicated improvement in algorithm performance.

1) The EKF algorithm's ability to track closely spaced, rapidly time-varying modes is improved in the presence of continuous wide-band input excitation of sufficient amplitude.

2) The ability to accurately predict future system stability is also improved substantially when continuous wide-band excitation is present as well. Actively probing the system during the critical

time intervals provides the algorithm with valuable information concerning the rates of change of the system's stability related parameters!

3) In the presence of small disturbances (gusts), rapidly time-varying modes can be accurately tracked only if exogenous inputs are also present. Slowly varying modes can be estimated given sufficient time, the amount of time required being a function of disturbance power and the underlying modal damping coefficients. Heavily damped modes are difficult to estimate with small error variances.

Processing of actual flight test data was also successfully performed. Results from the processing of data from the third ARW-1 flight test in the DAST program indicate that:

1) With proper tuning, the EKF algorithm is capable of providing low-variance estimates of frequencies and damping coefficients of multiple closely spaced time-varying modes.

2) In the presence of multiple modes with significantly different power levels, the high-power modes are estimated with small estimate error variances. Of the smaller power modes, if fewer modes are being estimated than are actually present, the modes which the algorithm 'locks-on to' are functions of the initial conditions and the tuning parameters.

3) Even though limited duration (and power) exogenous excitation was applied during the critical period of the flight test (just prior to the flutter incident), accurate estimates of time-to-instability and a 5-second ahead prediction of the damping coefficient for the mode which eventually became unstable were obtained.

From these DAST flight test results, in conjunction with the simulated DAST test case results, the following conclusions can be drawn.

1) When the data sampling rate (information rate) is sufficient, a simplified linear modal model of the complex aeroelastic dynamics of aircraft flutter can be used in an EKF algorithm to provide accurate estimates of system stability parameters and future predictions of system stability as well!

2) The results stated in 1) can be achieved in the presence of significant mismodeling. Many fewer modes can be estimated than are actually present as long as the number of high power modes does not exceed the number of modes being estimated.

3) Continuous wide-band exogenous input results in significant improvements in the algorithm's performance in terms of smaller estimate error variances and improved parameter tracking capability.

From the F-16 flight test results, in conjunction with the simulated F-16 test case results, the following conclusions can be drawn.

1) In the presence of low-power disturbances with no exogenous inputs, parameter convergence time increases significantly resulting in the capability to track only slowly-varying moderately stable modes.

2) High power disturbances or nearly unstable modes lead to increased measurement (output) information to noise ratios resulting in significant improvements in algorithm performance.

3) With no exogenous inputs, the variances of damping coefficient estimates are strong functions of the associated underlying values of the coefficients themselves. Heavily damped modes result in estimated damping coefficients with large variances.

4) Tuning of the algorithm in cases with no exogenous input becomes a more important factor in the final estimates and variances. This is due to the fact that without known persistent excitation, the a priori information included via q-tuning is a larger percentage of the overall information available to the algorithm.

Finally, preliminary algorithm computation counts indicate that real-time processing with a 2-input, 1-output, 2-mode model without direct feedthrough is currently possible at a throughput rate near 200 Hz on a processor whose FLOP time is 1μsec. Since asynchronous filter operation is possible, this represents an average processing rate. The actual throughput rate can be significantly higher during periods of quiescent input (zero input values), and lower during periods of intensive computation. The only caution is that the minimum processing rate be above approximately five times the Nyquist rate of the signals being processed, or else a sufficiently large buffer may be required to store data for short periods. Program modifications can be performed to make this rate achievable with direct feedthrough as well.

SECTION 9
RECOMMENDATIONS

During the course of this effort, EKF techniques were successfully applied in the development of an algorithm and subsequent program (MOPID) for real-time modal (flutter) parameter identification (estimation). There are several directions in which the current effort can be extended to increase the algorithm's capability to provide better parameter estimates as well as more accurate predictions of future system stability. Practically, there is also a significant level of effort remaining in the interfacing of this algorithm to an on-line data acquisition and real-time signal processing system. Future efforts in this direction include:

1) development of user-friendly real-time operator interface with current algorithm. This includes efficient input and algorithm set-up routines, as well as output interfaces and drivers to permit flexible selection of variables to be plotted and/or printed in real-time.

2) development of interrupt procedures and protocol for allowing operator intervention in cases where problem reinitialization is deemed necessary. This may involve simply altering parameter values on-line or performing a restructuring of the model being identified.

These tasks constitute basic requirements for installation of the MOPID program in a real-time system. Accomplishment of the following tasks will greatly improve the capabilities of the algorithm in several areas.

1) Tuning of the algorithm is a task facing the operator in a real-time environment. Though constant q's can be used, the results contained in this report indicate there is a significant improvement in algorithm performance when the tuning is a function of changes in the time-derivatives of the flight conditions. The process of tuning the algorithm could be automated to some extent by incorporating measurements of the rates of changes of the flight conditions such as dynamic pressure rate, altitude rate, axial acceleration, as well as potential changes in system configuration such as a stores or wing-tip mass release. This would remove some of the burden of tuning from the operator and would yield improved parameter convergence rates and estimates in rapidly changing environments.

2) Though excellent initial conditions are generally available for initialization of the modal parameters, this process could be automated to some extent using efficient batch methods for estimating modal parameters such as signal-subspace eigen-decomposition methods. These methods could also be used to aid in determination of an appropriate number of modes to include in the estimation.

3) Though the objective of the current effort was to provide estimates of current system parameters based on past information, the algorithm could be extended in a straightforward manner to provide optimal 'smoothed' estimates of the parameters over a desired interval by incorporating one of several fixed-interval smoothing algorithms. This would be appropriate in an off-line, batch processing environment since the computational load is significantly increased over the forward filter algorithm requirements.

4) As discussed in Section 7, the code was not optimized for direct feedthrough identification. Subsequent flight data analysis indicated the need for D-matrix element identification. Optimizing the code for D-matrix element identification would result in significant computational savings in these cases.

REFERENCES

[1] Walker, R.A., Gupta, N.K., "Real-Time Flutter Analysis", NASA Contractor Report 170412, March 1984.

[2] Ljung, L., Soderstrom, T., Theory and Practice of Recursive Identification", MIT Press, Cambridge, MA, 1983.

[3] Gilyard, G.B., Edwards, J.W., "Real-Time Flutter Analysis of an Active Flutter-Suppression System on a Remotely Piloted Research Aircraft", AGARD Flight Mechanics Panel Symposium, Cesme, Turkey, Oct. 11-14, 1982.

[4] Russo, M. L., Richards, P.T., Perangelo, H.J., "Identification of Linear Flutter Models", AIAA/AHS/IES/SFTE/DGLR 2nd Flight Testing Conf., Las Vegas, Nevada, Nov. 16-18, 1983.

[5] Wendler, B.H., "Near-Real-Time Flutter Boundary Prediction from Turbulence Excited Response", AIAA/ASME/ASCE/AHS 24th Structures, Structural Dynamics, and Materials Conference, May 2-4, 1983, Lake Tahoe, Nevada, pp 58-67, Part 2.

[6] Molusis, J.A., Kleinman, D.L., "On-Line Methods for Rotorcraft Aeroelastic Mode Identification", 21st IEEE Conf. on Decision and Control, Orlando, FL., Dec. 8-10, 1982.

[7] Walker, R.A., Gupta, N.K., Gilyard, G.H., "Algorithms for Real-Time Flutter Identification", Proc. of AIAA Guid. and Control Conf., Gatlinburg, TN, August 1983, pp. 432-437

[8] Roy, R.H., Saberi, H.A., Walker, R.A., "Fixed-Gain and Adaptive Techniques for Rotorcraft Vibration Control", ISI Final Report 45, under NASA Contract NAS2-11548, August 1984.

[9] Walker, R.A., Gregory, C.Z., Shah, S.C., "Matrix$_x$: A Data Analysis Identification, Control Design and Simulation Program", IEEE Control Systems Magazine, Dec. 1982.

[10] Bryson, A. E., Ho, Y. C., Applied Optimal Control, Hemisphere Publishing Co., New York, 1975

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

[12] Kailath, T., Linear Systems, Prentice Hall, Englewood Cliffs, N.J., 1980

[13] Jazwinski, A.H., Stochastic Processes and Filtering Theory, Academic Press, New York, 1970

[14] Ljung, L., "Asymptotic Behavior of the Extended Kalman Filter as a Parameter Estimator for Linear Systems", IEEE Trans. on Automatic Control, vol. AC-24, pp. 36-50, 1979

[15] Dobbins, J., Friedlander, B., Morf, M., Kailath, T., "Square-root Algorithms for Parallel Processing in Optimal Estimation", Automatica, vol. 15, pp. 299-306, 1979

[16] "MATRIXx User's Guide", Integrated Systems Inc., Palo Alto, Calif., September 1984

APPENDIX A
MOPID PROGRAM USER'S GUIDE

This appendix gives a description of the inputs to the MOPID program. Currently, NAMELIST (a DEC VAX/VMS extension to ANSI standard FORTRAN-77) input is used to initialize the filter parameters. The namelist is assumed to be located in a file whose name is EKFNML. On the VAX/VMS operating system, a logical ASSIGNment can be used to associate this logical file name with the actual file name, eg.

$ASSIGN real_file_name EKFNML

Directory extensions are required in the real_file_name specification if the file is not in the current working (or default) directory. The namelist name is &INPUT. The "&" must be in column two and the namelist terminates with &END, the "&" in column two as before.

Since there are essentially two categories of input variables, one for the estimation problem set-up and one for the simulation set-up, they are described separately for ease of use of this appendix. The descriptions are intended to be brief, leaving discussions of the subtleties to the body of the report.

The Table A-1 describes the use of the namelist inputs in specifying the estimation problem set-up. References to 'system inputs' and 'system outputs' (or simply 'inputs' and 'outputs') are used to describe those 'channels' in the 'input sampled data vector' which are the inputs to and outputs of the system being identified. These system inputs and outputs are outputs of the data collection system which comprise the 'input data vector' input to the MOPID program.

Table A-2 gives a description of the variables appropriate for the generation of simulated data using the capabilities internal to the MOPID program. In general, when performing simulations, entries from both tables will be required. The estimation problem is largely separate from the simulation set-up with a few exceptions, so both categories of inputs are required. This may seem somewhat redundant, but the intent is to facilitate studies of mismodeling where n modes are simulated and only $m < n$ are used in the estimation model.

TABLE A-1: ESTIMATION PROBLEM RELATED MOPID NAMELIST INPUTS

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
BFPOLE	1.0	s-plane location of the bias rejection filter pole in Hz.
DNAMES(10)	10*' '	Array of character*6 variables containing the names of the input data channels, eg. ' ARFSS'
DSIG(6)	6*0.0	Measurement noise sigmas (system outputs only).
DTMIN	0.001	Minimum allowable time between measurements in seconds.
DO(4,6)	24*0	Array containing initial values for the D-matrix elements. Used only if IDFT is non-zero. DO(I,J) is the (J,I)th element of the initial D-matrix estimate used to calculate the direct feedthrough component of the controls to the system outputs. The index reversal is intentionally designed so that specifying DO in the input namelist is easily accomplished by entering it as it would appear written in standard matrix form. The switch results from the FORTRAN standard column-wise matrix element storage versus the row-wise entry resulting from reading matrices entered in standard matrix form.
GO(4,8)	32*0	Array containing initial values for the G-matrix elements. Used only if IUTYPE(I) > 0 for some I < number of inputs used! See DO for indexing convention discussion.
G1UN	1.0	Internal units conversion factor for G-matrix element corresponding to 'position' oscillator states. Applies to all modes estimated. A value of G1UN = 100, for example, will result in internal position state associated G-matrix element estimates which are a factor of 100 **smaller** than the default of 1.0 would yield. Used for state normalization to prevent numerical instabilities.
G2UN	1.0	Internal units conversion factor for G-matrix element corresponding to 'velocity' oscillator states. See G1UN entry for an example of its use.
HEADER	Blank	Character*40 variable containing information about run conditions normally. Printed out

TABLE A-1: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
		each time PFQ directs summary information to be output.
H0(8,6)	48*0	Array containing initial values for the H-matrix elements. Used only if IYTYPE(I) > 0 for some I < number of outputs used! See D0 for indexing convention discussion.
H1UN	1.0	Internal units conversion factor for the **H**-matrix element corresponding to the 'position' oscillator state. See G1UN for an example of its use.
H2UN	1.0	Internal units conversion factor for the **H**-matrix element corresponding to the 'velocity' oscillator state. See G1UN for an example of its use.
IDEBUG	0	Debug printout flag. For values from 1 to 4, increasing amounts of debug output are sent to the standard output unit (usually the terminal).
IDFT	0	Flag to turn on estimation of a direct feedthrough (**D**-matrix) term in the measurement equations. IDFT = 1 enables **D**-matrix element identification.
IDUNIT	4	Input data unit number. If IDUNIT = 0, then simulation mode is assumed. A non-zero IDUNIT will cause a FORTRAN OPEN statement to be executed requiring EKFINP to be a valid file name (eg. under VAX/VMS, the following command will have to have been issued: "$ASSIGN input_data_file EKFINP"). IDUNIT can assume any integer value but should not conflict with standard input or output unit numbers (eg. 5 or 6 in standard F77 implementations)
IDUSW	1	Index of the system input whose value is to be calculated using USIGNO, TUSWCH, and the first system input. Basically this is used in the case where two elements of IUCHAN are the same (i.e. point to the same sampled data input channel), presumably an exogenous excitation, except for a possible time-varying sign change.
IGUMDL	1	Flag controlling $\mathbf{G}_u$-matrix element definition. Default is frequency normalized, i.e. for each

TABLE A-1: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
		mode and each input, the input distribution vector has the form [ωg_1 , $\omega^2 g_2$]'. Setting IGUMDL = 0 estimates [g_1 , g_2]' instead.
IGWMDL	1	Flag controlling G_w-matrix element definition. Default is to scale input q's by the frequency estimates as discussed in IGUMDL description.
IOPCNT	0	FLOP counter flag. Default is OFF. IOPCNT = 1 enables FLOP counting in computation intensive routines. A FLOP is considered as a single-precision floating point multiply and add. The op-count adds themselves are not counted and be forewarned that copious amounts of output will be generated!
IPFQFP	10	Modal parameter estimate print-out frequency. Default is to print modal parameter summary information every **tenth** time a point is saved for plotting in the plot array (See IPLFQ).
IPLCOR	0	Modal parameter correlation print-out flag. Default is OFF. IPLCOR = 1 causes selected elements of the modal parameter correlation matrix to be saved every IPLFQth time point for plotting in a file given the default name RTFACOR.DAT (the ".DAT" extension is supplied by the VAX/VMS operating system).
IPLFQ	1	Output plot frequency. Every IPLFQth data vector and estimated parameter vector (and sigmas) are stored in arrays for plotting. The inputs, outputs, and predicted data residuals and sigmas are written to file RTFADAT.DAT. The parameter estimates and sigmas are written to file RTFAPEST.DAT, and the selected correlation matrix elements are written to file RTFACOR.DAT. Note that the ".DAT" extension is supplied by the VAX/VMS operating system.
IPRCOR	0	State correlation matrix printout flag. Default is OFF. IPRCOR = 1 causes predicted and filtered correlation matrices to be printed out each time print-out is requested via PFQ.
IPRCOV	0	State covariance matrix printout flag. Default is OFF. IPRCOV = 1 causes predicted and filtered covariance matrices to be printed out each

TABLE A-1: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
		time print-out is requested via PFQ.
IPREST	1	Estimated state vector print-out flag. Default is ON. IPREST = 0 suppresses print-out of the predicted and filtered state estimates each time print-out is requested via PFQ.
IPRQ	0	Q-matrix print-out flag. Default is OFF. IPRQ = 1 causes the Q-matrix to be printed out each time print-out is requested via PFQ.
IPRSIM	1	Simulated data parameter summary information print-out flag. Default is ON. IPRSIM = 0 suppresses print-out of the simulated parameter summary information. If requested, information is printed out at initialization time only.
IPRSUM	1	Estimation algorithm parameter specification summary print-out flag. Default is ON. IPRSUM = 0 suppresses algorithm initialization summary information print-out. If requested, information is printed out at initialization time only.
IPUNIT	10	Plot output unit number. Unless the default conflicts with a system definition, there is no need to alter this value.
IQAPPX	0	Q-matrix approximate calculation flag. Default is to use 'exact' expression for Q. IQAPPX = 1 saves a bit of computation by approximating Q. The savings are not great however!
IUBF	0	System input bias rejection filter flag. Default is OFF. IUBF = 1 causes all system inputs to be passed through a bias rejection filter prior to inclusion in the identification algorithm.
IYBF	0	System output bias rejection filter flag. See IUBF description for further details.
IUCHAN(4)	4*0	System input channel pointer array. Input to the algorithm is assumed to be in the form of a vector whose first element is the time of the measurement, and whose remaining elements are the measurements themselves. The first element (time) is given a zero index. Both system inputs and outputs are included in this vector and can occur in any order. IUCHAN(I) = J (I,J integer)

TABLE A-1: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
		specifies that the Ith element of the system input vector is the Jth element in the sampled data vector. The maximum size of the sampled data vector is 10 which is therefore the maximum J-value allowed. Non-zero values for J must occur sequentially starting from index I = 1. The number of system inputs used is the number of consecutive non-zero elements of IUCHAN starting from index 1 (in the real data case).
IUMDL	0	System input approximation order. Default is zero-order hold approximation. IUMDL = 1 causes trapezoidal input integration to be performed. Though the approximation is clearly superior to ZOH, more computation is required each iteration due to partial calculations primarily, so there are data rate versus model accuracy trade-offs here.
IUTYPE(4)	4*0	Input type specification array. IUTYPE(I) = 1 results in G-matrix elements being estimated for system input number I. A value of -1 results in the appropriate G-matrix elements being constrained to their initial values specified in GO. A value of 1 or -1 is required for each system input specified in IUCHAN.
IYCHAN(6)	6*0	System output channel pointer array. See IUCHAN for a description of the use of this input.
IYTYPE(6)	6*0	Output type specification array. IYTYPE(I) = 1 results in H-matrix elements being estimated for system output number I. A value of -1 results in the appropriate H-matrix elements being constrained to their initial values specified in HO. A value of 1 or -1 is required for each system output specified in IYCHAN.
NINPS	0	For real data, NINPS is the total number of system inputs in the sampled data vector which is input to the program. NINPS + NOUTPS + 1 must equal the number of elements in each sampled data vector (which in the off-line version of the program are stored in a data file written with unformatted FORTRAN writes!).
NMODES	1	Number of modes which are to be included in the system model being identified. The current maximum is **four** (4).

TABLE A-1: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
NOUTPS	0	For real data, NOUTPS is the total number of system outputs in the sampled data vector. See NINPS for further details.
OMEGA0(4)	4*0	Initial frequency estimates.
PFQ(2,10)	1E6,1,18*0	Print-out frequency array. PFQ(1,J) = T(J), PFQ(2,J) = DTJ specifies that from relative time T(J-1) to T(J) that estimate and covariance summary information is to be printed-out every DTJth second. Modulus arithmetic is used so the actual output times depend on the actual relative time values. Relative time is the time specified in the current sampled data vector minus the time specified in T0 (which is usually the first sample time!). For example, for T0 = WINDOW(1), PFQ=1,.1,10,1,1E6,100, requests print-out each 1/10 of a second (on the even 1/10th of a second relative time) until relative time 1 second, every one second thereafter until 10 seconds, and every 100 seconds from then on. This input is most useful in 'post-flight' or simulated data analysis where increased print-out frequency is desired during critical periods of the system activity!
QD(4,6)	24*0	D-matrix element associated q's. The QD(I,J) is the square-root of the process noise variance density to be associated with the (J,I)th element of the D-matrix. The index reversal is the result of the column-wise storage convention employed by FORTRAN and the natural row-wise input when specifying arrays in the input namelist. This is done so that input namelists have matrices specified as they would be written down on paper for ease of input debugging!
QDIMP(4,6)	24*0	Impulsive q's associated with D-matrix elements. These q's are RSS'ed with associated QD specified q's at the times TUSWCH(I). See QD for further details on the indexing.
QG(8,4)	32*0	G-matrix element associated q's. See QD for further details.
QGIMP(8,4)	32*0	Impulsive q's associated with G-matrix elements. See QDIMP for further details.

TABLE A-1: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
QH(8,6)	48*0	H-matrix element associated q's. See QD for further details.
QHIMP(8,6)	48*0	Impulsive q's associated with H-matrix elements. See QDIMP for further details.
QOMEGA(4)	4*0	Modal frequency parameter associated q's.
QWIMP(4)	4*0	Impulsive q's associated with frequency states.
QX(8)	8*0	Dynamic state vector associated q's.
QXIMP(8)	8*0	Impulsive q's associated with dynamic states.
QZETA(4)	4*0	Damping coefficient parameter q's.
QZETAV(4)	4*0	Damping coefficient velocity parameter q's.
QZIMP(4)	4*0	Impulsive q's associated with damping coefficient parameters.
QZVIMP(4)	4*0	Impulsive q's associated with damping coefficient velocity parameters.
SIGDO(4,6)	24*0	Initial D-matrix element sigmas. See QD for further details on indexing.
SIGGO(4,8)	32*0	Initial G-matrix element sigmas. See QD for further details on indexing.
SIGHO(8,6)	48*0	Initial H-matrix element sigmas. See QD for further details on indexing.
SIGOMO(4)	4*0	Initial frequency parameter sigmas.
SIGXO(8)	8*0	Initial dynamic state sigmas.
SIGZO(4)	4*0	Initial damping coefficient sigmas.
SIGZVO(4)	4*0	Initial damping coefficient velocity sigmas.
THRSH	10.0	Data outlier rejection threshold. Predicted data residuals exceeding THRSH*theoretical sigma result in the measurement being disregarded.
TSS	0.0	Steady-state time interval in seconds. Starting from WINDOW(1), inputs and outputs are high-pass filtered (bias rejection) for TSS seconds before estimation proceeds. The intent is to

TABLE A-1: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
		allow the filter outputs to reach steady state in circumstances where the biases are significant and the time constants are small. If IYBF and IUBF are both zero, this is a real waste of time!
TUSWCH(50)	1E6,49*0	Array of times at which input symmetry switching is performed. Used basically for generating a second input excitation signal from a given excitation signal assuming that the sign of the excitation changed at time TUSWCH(I) from plus to minus or vice-versa. This is required since in some cases (eg. DAST data analysis) only one excitation signal is monitored, and is applied two system inputs differently at different times. The times are specified in absolute, not TO relative time! See USIGNO and IDUSW.
TO	WINDOW(1)	Relative time reference for printout frequency calculation.
UMIN	5E-3	Threshold on absolute value of inputs below which they are set to 0.0 to save computation.
USIGNO	1	Initial sign to be applied to the input excitation signal when computing the value of the IDUSWth input.
WINDOW(2)	-1D10,1D10	Algorithm start and stop times. Only data whose associated time falls within WINDOW will be processed. Used primarily for off-line analysis of real and simulated data.
WUN	1.0	Frequency units conversion factor. See G1UN for further details.
XO(8)	8*0	Initial dynamic state vector estimates.
X1UN	1.0	Initial position dynamic state units conversion factor. See G1UN.
X2UN	1.0	Initial velocity dynamic state units conversion factor. See G1UN.
WMAX	0.5/DTMIN	Upper bound on estimated frequencies.
WMIN	BFPOLE	Lower bound on estimated frequencies.
ZETAO(4)	4*0	Initial damping coefficient estimates.

TABLE A-1: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
ZETAVO(4)	4*0	Initial damping coefficient velocity estimates.
ZMAX	0.9999	Upper bound on damping coefficient estimate.
ZMIN	-0.9999	Lower bound on damping coefficient estimate.
ZUN	1.0	Damping coefficient and damping coefficient velocity parameter units conversion factor. See G1UN.

The preceding table discussed the namelist variable inputs with application to the estimation problem formulation. The following table describes the namelist input variables associated with the setting-up of simulated data cases. Some of the namelist variables appear in both tables. This is due to the fact that the interpretation of the namelist variable depends on the mode, simulated or real data.

Simulated data refers to data generated by the simulated data generation capability of the MOPID program. Simulated data from other sources can certainly be accessed just as actual flight test (or real) data are accessed (through input_data_file input), but are treated as real data as far as the algorithm is concerned. The advantage of internal simulated data generation is that the 'true' parameter trajectories are saved for parameter estimate error calculations! These are potentially valuable for Monte Carlo analysis should it be desired.

TABLE A-2: SIMULATED DATA RELATED MOPID NAMELIST INPUTS

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
DSIM(4,6)	24*0	Simulated **D**-matrix initialization. The (I,J)th element of DSIM is the (J,I)th element of the **D**-matrix used to calculate the direct feedthrough component of the controls to the system outputs. The index reversal is intentionally designed so that specifying DSIM in the input namelist is easily accomplished by entering it as it would appear written in standard matrix form. The switch results from the FORTRAN standard column-wise matrix element storage versus the row-wise entry resulting from reading matrices entered in standard matrix form.
DTSIM	0.0	Simulated data integration step size. A value greater than or equal to 1D-4 must be entered when simulating data.
DW	30.0	Swept sine-wave input delta frequency value in Hz. See INTYPE, WSTART, SWEEPDT, SWPAMP for further details.
GUSIM(4,8)	32*0	G_u-matrix values used in simulating data. Their interpretation depends on ISGUML. See DSIM for an indexing convention discussion.
GWSIM(2,8)	16*0	G_w-matrix values used in simulating data. This is the process noise distribution matrix and the interpretation of the elements depends on ISGWML. See DSIM for an indexing convention discussion.
HSIM(8,6)	48*0	**H**-matrix values used in simulating data. See DSIM for an indexing convention discussion.
IDUNIT	4	Input data unit number. Setting IDUNIT = 0 turns on the simulated data generation capability!
INTYPE(10)	10*1	Input type specification flag. INTYPE(I) = 1 specifies that the Ith input interval is to contain an input whose function form is one cycle of a sine wave whose period is the average of the simulated modal frequencies at the time of the onset of the 'pulse'. INTYPE(I) = 2 specifies the tapered swept sine-wave or chirp input waveform. See PLSINT, PLSAMP, TPULS1, WSTART, DW, SWEEPDT, SWPAMP.

TABLE A-2: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
IPRSIM	1	Simulated data parameter summary information print-out flag. Default is ON. IPRSIM = 0 suppresses print-out of the simulated parameter summary information. If requested, information is printed out at initialization time only.
ISGUML	1	Flag to determine interpretation of G_u-matrix elements in generating simulated data. Default is frequency normalized, i.e. for each mode and each input, the input distribution vector has the form $[\ \omega g_1\ ,\ \omega^2 g_2\ ]'$. Setting ISGUML = 0 simulates $[\ g_1\ ,\ g_2\ ]'$ instead.
ISGWML	1	Flag to determine interpretation of G_w-matrix elements in generating simulated data. Default is frequency normalized, i.e. for each mode and each input, the input distribution vector has the form $[\ \omega g_1\ ,\ \omega^2 g_2\ ]'$. Setting ISGWML = 0 simulates $[\ g_1\ ,\ g_2\ ]'$ instead.
ISUMDL	1	Input interpolation order flag. Default is to perform trapezoidal integration of 'continuous' simulated inputs (first-order hold). ISUMDL = 0 specifies zero-order hold integration.
IUCHAN(4)	4*0	System input channel specification array. For simulated data, these elements must take on values I, 0 <= I <= NINPS, since the inputs are arbitrarily given a channel number their cardinal number in the input ordering.
IWRTDT	0	Simulated data file control flag. The default is no file access whatsoever. IWRTDT > 0 specifies that the un-noised outputs, inputs, and an entire simulated data summary are to be written out to file RTFASIM.DAT opened as unit number IWRTDT. IWRTDT < 0 specifies that instead of actually simulating data on-line, unit number -IWRTDT is to be opened (file name RTFASIM.DAT) and read for appropriate specifications and data. It is assumed that the file was written on a previous pass with IWRTDT > 0.
IYCHAN(6)	6*0	System output channel specification array. See IUCHAN for a discussion of valid entries. Note IUCHAN and IYCHAN can specify fewer inputs and/or outputs than were actually used in simulating the data to investigate mismodeling.

TABLE A-2: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
NINPS	0	Number of inputs to simulate. Must equal number written to simulated data file if such a file is being read.
NMSIM	0	Number of modes to be used in simulation. A positive value less than 5 must be entered when requesting simulated data generation.
NOUTPS	0	Number of outputs to simulate. Meaning is distinctly different in real data case.
PLSAMP	1.7	Single cycle sine-wave ('pulse') amplitude.
PLSINT(10)	10*2.0	Time interval containing input type INTYPE(I). Time is in seconds and is the length of the interval. For example, PLSINT(3) = 5 specifies that input type INTYPE(3) will occur during the third input interval and the interval will be 5 seconds in duration. See INTYPE, TPULS1, and SWEEPDT for further timing information.
QW(2)	2*0	Process noise sigmas for disturbance simulation. Non-zero values are sigmas of WGN to be passed through a low-pass filter (if WPOLE > 0.0) and used as disturbance inputs to the simulation.
SDSIG(6)	6*0	Simulated data measurement noise sigma. WGN of sigma SDSIG(I) is added to output I after writing output I to the simulated data file if requested (see IWRTDT).
SIMTSS	0.0	Time interval in seconds during which the simulation integration is carried out, but no inputs or outputs are generated for estimation purpose. The idea is that during this time period the simulation is reaching a statistical steady-state. Note the relevant time constants in the problem are the bias filter pole and the largest $1/\zeta\omega$ product of the modes being simulated.
SWEEPDT	1.0	Duration of the frequency sweep in seconds. Logarithmic frequency sweep starts at WSTART and in SWEEPDT second increases to WSTART + DW Hz. SWEEPDT should be < PLSINT(I) if INTYPE(I) = 2 !
SWPAMP	1.0	Amplitude of the swept sine-wave (chirp) input waveform.

TABLE A-2: (CONTINUED)

VARIABLE NAME	DEFAULT VALUE	DESCRIPTION
TPULS1	0.0	Time relative to T0 in seconds of the start of the first PLSINT interval for input function generation.
T0	WINDOW(1)	Time reference for output determination as well as simulated modal parameter calculation. See WSIM and ZSIM.
WINDOW(2)	-1D10,1D10	Start and stop time for simulated data generation. WINDOW=0,10 is a typical entry and specifies the simulation and estimation interval is to be 0 to 10 seconds. See DTMIN (in prior table) and DTSIM for further timing information.
WPOLE	0.0	s-plane pole of the low-pass process noise 'gust' filter for simulated data generation. The value is in Hz and 0.0 specifies no low-pass filtering is to be performed. Both process noise inputs, if specified, are filtered identically.
WSIM(4)	4*0	Initial values of the frequencies of the NMSIM modes to be simulated. See WVSIM.
WSTART	10.0	Initial value of the frequency in Hz of the simulated chirp or swept sine-wave input.
WVSIM(4)	4*0	Values for the time derivative of the frequencies to be used in the simulations, cf. W(t) = WSIM + (t-T0) * WVSIM for each mode.
ZASIM(4)	4*0	Damping coefficient acceleration specifications for simulated data generation. See ZSIM.
ZSIM(4)	4*0	Initial values of the damping coefficients for the modes to be simulated. The coefficients can be time-varying and are given by: Z(t) = ZSIM + ZVSIM*(t-T0) + ZASIM*(t-T0)**2
ZVSIM(4)	4*0	Damping coefficient velocity specifications for simulated data generation. See ZSIM.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-3933		

4. Title and Subtitle	5. Report Date
REAL-TIME FLUTTER IDENTIFICATION	October 1985
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Richard Roy and Robert Walker	ISI Report 48
	10. Work Unit No. RTOP 505-33-41
9. Performing Organization Name and Address Integrated Systems, Inc. 101 University Avenue Palo Alto, CA 94302-1696	11. Contract or Grant No. NAS2-11451
	13. Type of Report and Period Covered Contractor Report — Final
12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

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

16. Abstract

This report documents the techniques and a FORTRAN 77 MOdal Parameter IDentification (MOPID) computer program developed for identification of the frequencies and damping ratios of multiple flutter modes in real time. Physically meaningful model parameterization has been combined with state-of-the-art recursive identification techniques and applied to the problem of real-time flutter mode monitoring. The performance of the algorithm in terms of convergence speed and parameter estimation error is demonstrated for several simulated data cases, and the results of actual flight data analysis from two different vehicles are presented. The results indicate that the algorithm is capable of real-time monitoring of aircraft flutter characteristics with a high degree of reliability.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Flutter analysis Flutter testing Parameter identification Recursive identification Extended Kalman filter Real-time flutter analysis	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	146	A07

For sale by the National Technical Information Service, Springfield, Virginia 22161
NASA-Langley, 1985

End of Document