

83 W 1 15 JK

EIGENSPACE TECHNIQUES FOR ACTIVE FLUTTER SUPPRESSION
FINAL REPORT COVERING THE PERIOD OCT. 1981 TO DEC. 1982
NASA RESEARCH GRANT NAG-1-217*

William L. Garrard (Principal Investigator)
Bradley S. Liebst

Department of Aerospace Engineering and Mechanics
University of Minnesota
Minneapolis MN 55455



*NASA Technical Officer for this grant was Mr. William M. Adams, Jr.
of NASA Langley Research Center, Hampton VA 23665

Abstract

Eigenspace (ES) techniques are used to design an active flutter suppression system for the DAST ARW-2 flight test vehicle. The ES controller meets control surface activity specifications and at the flutter test condition provides reduced wing root torsion at the gust test condition, and results in improved flutter boundaries. The ES controller is compared with a controller designed using Linear Quadratic (LQ) techniques. The LQ controller exhibits better phase margins at the flutter condition than does the ES controller but the LQ design requires large feedback gains on actuator states while the ES does not. This results in reduced overall actuator gain for the LQ design.

Nomenclature

Vectors

u = control input

v_i = attainable closed loop eigenvector associated with λ_i eigenvalue

v_{d_i} = desired closed loop eigenvector associated with λ_i eigenvalue

w_i = vector used in calculation of ES gain matrix, see Eqn. 17

x = system state

ξ_F = flexure modal displacements

ξ_C = control surface displacements

Γ = disturbance input vector

Matrices

- $[A_m]$ = aerodynamic coefficient matrix
A = dynamics matrix
B = control distribution matrix
 $[C_s]$ = structural damping matrix
E = aerodynamic coefficient error matrix
K = control gain matrix
 $[K_s]$ = structural stiffness matrix
 $[M_s]$ = structural mass matrix
 P_i = eigenvector weighting matrix
Q = state weighting matrix
 $[Q_c]$ = calculated unsteady aerodynamic influence coefficient matrix
 $[Q_A]$ = s-plane representation of unsteady aerodynamic influence coefficient matrix
R = control weighting matrix
V = matrix whose columns are v_i
W = matrix whose columns are w_i
A = diagonal matrix composed of λ_i

Scalars

- c = reference chord, 0.75m
 $H(s)G(s)$ = loop transfer function
j = $\sqrt{-1}$
k = reduced frequency
L = reference length in gust model, 762m
M = Mach number
q = dynamic pressure
s = Laplace operator
V = forward velocity
 β_m = aerodynamic lag frequencies
 η = zero mean white noise input to gust model with intensity,
$$\frac{L}{V} \overline{\xi_G^2}$$

 λ_i = ith eigenvector
 $\overline{\sigma}(E)$ = maximum singular value of E
 $\underline{\sigma}(E)$ = minimum singular value of E
 ω = circular frequency
 ξ_G = normal wind gust velocity
 $\overline{\xi_G}$ = rms normal wind gust velocity

Subscripts

i = inboard aileron
o = outboard aileron
ES = eigenspace
LQ = linear quadratic
s = structural

Superscripts

* = complex transpose
T = transpose
-1 = inverse

Table of Contents

I.	Introduction.	6
II.	Mathematical Models and Performance Specifications	8
III.	Eigenspace Design	19
IV.	Results	29
V.	Conclusions	43
VI.	References.	47
VII.	Publications Issued During the Course of this Research.	49

Appendices

A. Flexure Mode Shapes

B. Program Descriptions and Listings

List of Tables

Table 1	Maximum Singular Value of Error Matrix vs β
Table 2	Eigenvalues of DAST ARW-2 Flight Test Vehicle at $M=0.86$, $h=4572m$, Based on 2 Rigid Body and 7 Flexure Modes
Table 3	Eigenvalues of DAST ARW-2 Flight Test Vehicle at $M=0.86$, $h=4572m$, Based on 3 Flexure Modes
Table 4	Comparison of RMS Control Surface Activity for Various Controller Designs and Flight Conditions
Table 5	RMS Response for Various Weights on Actu- ator Rates
Table 6	RMS Loads at Gust Test Condition

List of Figures

- Fig. 1 Root Locus of Seven Mode Model of Uncontrolled Wing as Velocity is Varied
- Fig. 2 Root Locus of Three Mode Model of Uncontrolled Wing as Velocity is Varied
- Fig. 3 Variation in Actuator Roots with Weighting on Inboard Aileron Deflection Rate
- Fig. 4 Inboard Actuator Transfer Function Frequency Response
- Fig. 5 Root Locus of LQ Controlled Wing as Velocity is Varied
- Fig. 6 Root Locus of ES Controlled Wing as Velocity is Varied
- Fig. 7 Flutter Boundaries for Uncontrolled, LQ, and ES Controlled Wing
- Fig. 8 Frequency Response of Magnitude of $H(s)G(s)$ for LQ and ES Designs
- Fig. 9 Frequency Response of Phase of $H(s)G(s)$ for LQ and ES Designs

Eigenspace Techniques for Active Flutter Suppression

I. Introduction

The objective of the research described in this report is the application of Eigenspace (ES) design techniques to the synthesis of active flutter suppression systems. ES techniques allow the designer to use feedback control to place closed loop eigenvalues and shape closed loop eigenvectors to satisfy performance specifications. The basic theory behind ES design has been given by Moore [1] and others. Moore has shown that it is possible not only to place controllable eigenvalues and shape controllable eigenvectors but also to shape uncontrollable eigenvectors. Since the dynamic response characteristics of a system are determined by its eigenvectors as well as its eigenvalues, the ability to shape eigenvectors provides the designer with an important tool.

If performance specifications are given or can be interpreted in terms of desired closed loop eigenvalues and eigenvectors, ES techniques provide a natural design procedure where the desired eigenstructure (if obtainable) can be calculated directly without iteration. Cunningham [2] has shown how ES techniques can be used to improve aircraft flying qualities by placing rigid body poles in desired locations and decoupling rigid body modes. If performance specifications cannot be stated directly in terms of closed loop eigenvalues and eigenvectors, for

example specifications on rms responses or stability margins, the application of ES techniques is not so straightforward.

This is the case in aeroelastic control problems such as flutter suppression. One of the principal contributions of this report is the description of a methodology for application of ES techniques to active flutter suppression and other aeroelastic control problems.

To the authors' knowledge, the full power of ES techniques to shape eigenvectors as well as place eigenvalues has not been applied to active flutter suppression. Ostroff and Pines [3] used eigenvalue placement to design a flutter controller, but did not attempt to shape closed loop eigenvectors. In this report ES design techniques are applied to the design of a flutter suppression system for the DAST ARW-2 flight test vehicle. Only full-state feedback is considered. Use of ES techniques in the design of robust state observers will be the subject of future investigations.

It is shown that ES techniques can easily be used to design a flutter controller which satisfies performance specifications on rms control surface activity at the design condition. This controller also provides some gust load alleviation capability at off-nominal conditions. The ES controller requires no feedback on control surface actuator states and thus the open loop frequency response characteristics of the actuators are retained in the closed loop system.

The remainder of the report is divided into four major sections. First mathematical models of the aircraft, actuators, and normal wind gust are presented and performance requirements are discussed. Next the theory of ES design is described. Then the results obtained by applying this theory to the design of a flutter suppression system for the DAST ARW-2 aircraft are given and compared with results obtained from a controller designed using Linear Quadratic (LQ) optimal control theory. Finally conclusions and suggestions for future research are presented. A brief description and listing of a computer program used to perform ES design is given in Appendix B.

II. Mathematical Models and Performance Requirements

The DAST ARW-2 flight test vehicle is a Firebee II Drone which has been modified by replacing the conventional wing with a high aspect ratio, supercritical wing designed to flutter within the flight envelope. Two control surfaces, an inboard and an outboard aileron, are available on the wing and a stabilizer is available on the horizontal tail. The outboard aileron is to be used for flutter suppression, gust load alleviation, and maneuver load alleviation. The inboard aileron is to be used for maneuver load alleviation, and the stabilizer is to be used to compensate for reduced static stability, for automatic flight control, gust load alleviation, and maneuver load alleviation. In practice only the outboard aileron

is to be used for flutter control. However, both the inboard and outboard surfaces were utilized in the controller designs since ES design methodologies are most useful in the design of controllers for systems with multiple control inputs.

The design flight condition for the flutter control system was a Mach Number of 0.86 and an altitude of 4572 m (15000 ft). At this flight condition the flutter control system was required to stabilize the wing. A normal gust with rms velocity of 3.66 m/s (12 ft/s) was assumed at this condition. The rms deflection of the inboard aileron was limited to 10° and the deflection rate to $130^\circ/\text{s}$. The rms deflection of the outboard aileron was limited to 15° and the deflection rate to $740^\circ/\text{sec}$. The control system was also to be evaluated on its ability to reduce bending and torsional stresses and shear forces in the wing at a gust test condition of Mach 0.7 and 4572 m. The vertical gust velocity at this condition was 18 m/s (59 ft/s). In the actual DAST vehicle, the flutter suppression system would not be used at this condition because a separate gust load alleviation system is available. However, the gust test condition did provide a convenient condition for evaluating the performance of the flutter controllers at an off-nominal condition; therefore, in this study it was assumed that the flutter controller was also to be used for gust load alleviation.

Actuator/Control Surfaces

Stabilizer. The stabilizer transfer function was a simple first-order lag

$$\xi_{c_e}/u_e = \frac{20}{s+20} \quad (1)$$

The allowable rms control surface activity levels for the stabilizer were 7 degrees and 80 degrees/s.

Inboard Aileron. An eleventh-order transfer function with third order numerator dynamics was given for the inboard aileron. Up to about 300 rad/s, a fourth order model gave an acceptable approximation of the eleventh order inboard aileron transfer function. This fourth order approximation was

$$\xi_{c_i}/u_i = 1.614 \times 10^{11} / (s^2 + 671s + (477)^2) (s^2 + 322s + (878)^2) \quad (2)$$

Outboard Aileron. A seventh-order transfer function with second-order numerator dynamics was given for the inboard aileron. Up to about 300 rad/s, a third order transfer function gave an acceptable approximation of the seventh order outboard aileron transfer function. This third order approximation was

$$\xi_{c_o}/u_o = 1.774 \times 10^7 / (s+180) (s^2 + 251s + (314)^2) \quad (3)$$

Wind Gust. The wind gust was modeled by the second-order model given below

$$\xi_G/\eta(s) = (1 + (\sqrt{3}L/V)s) / (1 + (L/V)s)^2 \quad (4)$$

Aircraft. The modes of the aircraft considered were rigid body plunge and pitch modes and seven symmetric aeroelastic modes of the wing. The flexible aircraft model was

$$([M_s]s^2 + [C_s]s + [K_s]) [\xi_F] + q[Q_c(s)] \begin{bmatrix} \xi_F \\ \xi_C \\ \xi_G \\ \frac{V}{V} \end{bmatrix} = 0 \quad (5)$$

The aerodynamic influence matrix $[Q_c(s)]$ was calculated over a range of reduced frequencies ranging from 0.0 to 1.2 by a doublet lattice procedure. A rational s-plane approximation of $Q_c(s)$ was given by the approximation

$$[Q_A(s)] = [A_0] + [A_1] \frac{cs}{2V} + [A_2] \left[\frac{cs}{2V} \right]^2 + \sum_{m=1}^L \frac{[A_{m+2}]s}{s + \frac{2V}{c}\beta_m}$$

This approximation has been widely used in design of active flutter suppression systems [4-10].

The first column of the matrix $[A_0]$ was set equal to the first column of $[Q_c(0)]$, which is zero since the aerodynamic forces due to plunge displacement are zero. The second column of $[A_0]$ gives the force due to a change in angle of attack resulting from a change in pitch angle and must be set equal to the second column of $[Q_c(0)]$. The first column of $[A_1]$ multiplied by $c/2$ gives the force due to a change in plunge velocity. Since the force due to a change in angle of attack must be the same regardless of whether this change results from a change in plunge velocity or a change in pitch angle, the first column of

$[A_1]$ must equal the second column of $[Q_c(0)]$ divided by $c/2$ (there is also a scaling factor of 0.0062486 which must be divided into $[Q_c(0)]$).

Usual procedure is to select the β'_m 's so as to bracket the reduced flutter frequencies [6]. Once the β'_m 's are specified, the remaining elements of the $[A_m]$ matrices are determined to give the best least squares fit to Q_c over the range of reduced frequencies for which this matrix has been calculated.

In this study a new method was used to select the β'_m 's. An error matrix was defined as

$$E(j\omega) = [Q_c(j\omega)] - [Q_A(j\omega)]$$

The norm of this matrix can be bounded above and below by its maximum and minimum singular values [11]

$$\underline{\sigma}(E) \leq \frac{\|Ex\|}{\|x\|} \leq \bar{\sigma}(E)$$

The singular values of E are defined as the positive square roots of the eigenvalues of E^*E . If the β'_m 's are chosen to minimize $\bar{\sigma}(E)$, the norm of the error matrix will be small. In this study a single β was used in $[Q_A]$. Since the reduced flutter frequency was known to be small (0.15), β was selected to minimize $\bar{\sigma}(E)$ at zero frequency. The procedure for determining the $[A_m]$'s and β was as follows (1) a small initial value for β was arbitrarily selected, (2) the first column of $[A_0]$ was set equal to zero, the second column of $[A_0]$ was set equal to the second column of $[Q_c(0)]$ and the first column of $[A_1]$ was set equal to the second column of $[Q_c(0)]$ divided by $c/2$ times a scale factor, (3) the remaining values of $[A_0]$, $[A_1]$, $[A_2]$ and $[A_3]$ were determined to give the

best least squares fit to $[Q_c(j2kV/c)]$ over the range of reduced frequencies 0.0, 0.05, 0.2, 0.3, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 1.0 and 1.2 (4) the maximum singular value of the error matrix $E(j\omega=0)$ was calculated (5) β was increased by a small amount and the process was repeated until $\bar{\sigma}(E)$ reached a minimum. The results are summarized below

β	$\bar{\sigma}(E(0))$
0.100	214.4
0.125	138.2
0.130	137.0
0.135	147.0
0.150	162.8

Table 1 Singular Value of Error Matrix versus β
 The value of β equal to 0.13 which minimizes $\bar{\sigma}(E(0))$ is very close to the reduced flutter frequency of 0.15. The eigenvalues resulting from this model for a Mach number of 0.86 and an altitude of 4572 m are given below

Mode	Eigenvalues
plunge	0, -1.17
pitch	-1.74±j6.29
1st flexure	41.6±j118.1
3rd flexure	-0.7±j136.6
4th flexure	-97.1±j108.4
6th flexure	-16.9±j218.3
8th flexure	-4.1±j397.4
9th flexure	-11.0±j425.0
10th flexure	-24.8±j452.5

Table 2 Eigenvalues of DAST ARW-2 Flight Test Vehicle at M=0.86, h=4572m based on 2-Rigid Body and 7 Flexure Modes

The values of the flexure mode eigenvalues are close to those calculated for the ARW-2 using a model containing four lag states [12]. The pitch eigenvalues were not too different from the actual values of $-1.5 \pm j5.6$; however, one of the plunge eigenvalues is considerably different from the actual values of $0.0 \pm j1.1 \times 10^{-3}$. The flutter characteristics of the aircraft are not affected by the rigid body modes and these modes are not included in the model used in the control system design. The poorest correlation between elements of the $[Q_C]$ and $[Q_A]$ matrices occurred for the coefficients associated with the gust velocity, even then rms structural and control surface gust responses calculated using the model with a single β were close to those calculated from higher-order models using several values of β_m .

It is felt that selection of the numerical values of the β'_m 's based on the maximum and minimum singular values of an error matrix such as given by Eq. 7 has considerable potential in generating low-order approximate models of unsteady aerodynamics. Even the simple approach of minimizing the maximum singular value of the error matrix at a fixed frequency yielded acceptable results. Other approaches such as choosing the β'_m 's to minimize functions of both the maximum and minimum singular values over a range of frequencies should be investigated.

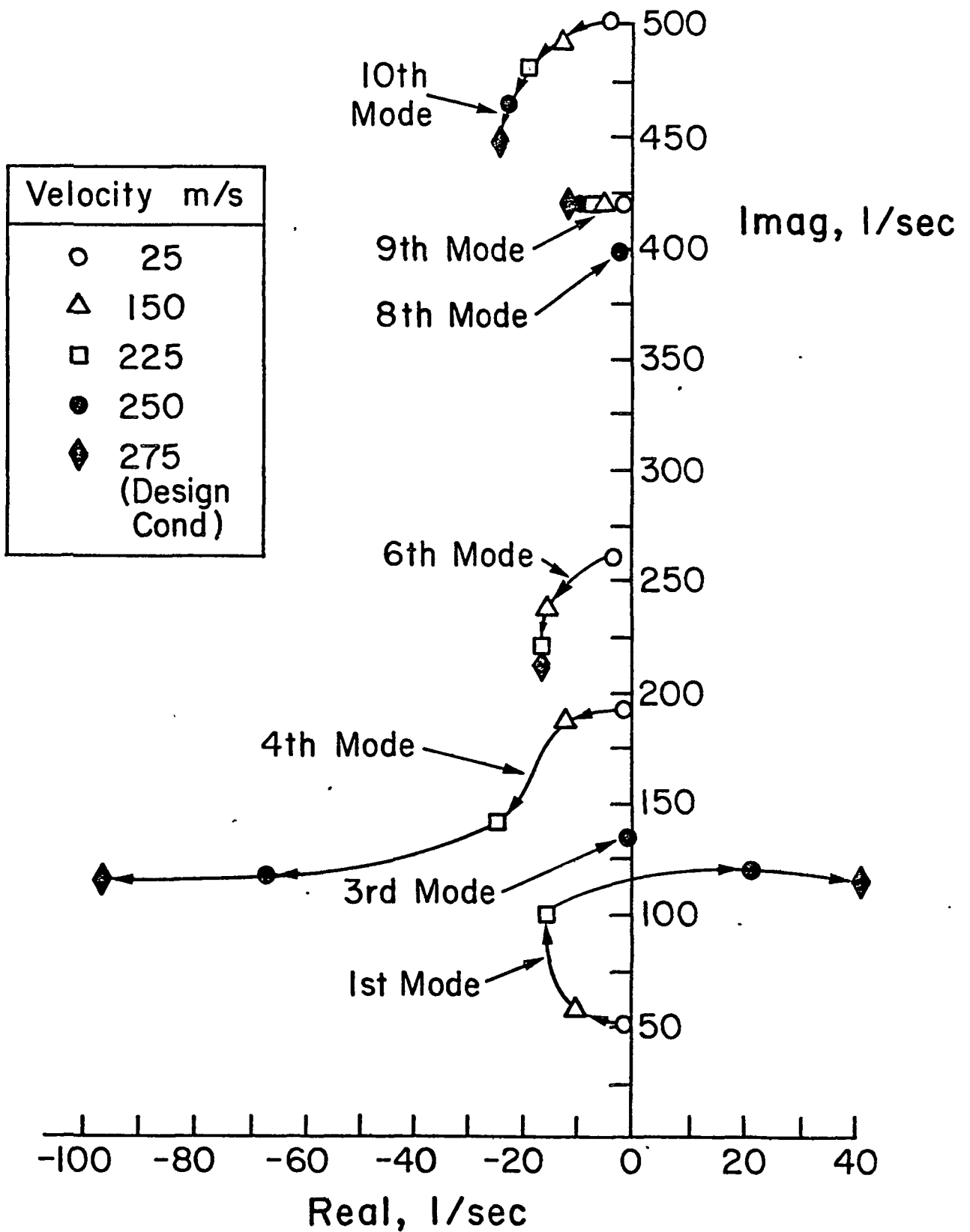


Fig. 1 Root Locus of Seven Mode Model of Uncontrolled Wing as Velocity is Varied

Some possible indices are as follows:

1. Minimize average $\bar{\sigma}$ and $\underline{\sigma}$ over a range of frequencies, i.e.

$$\min\left\{\frac{1}{\omega_2 - \omega_1} \int_{\omega_1}^{\omega_2} \sigma(E)(j\omega) d\omega\right\}$$

2. Minimize maximum value of $\bar{\sigma}$ and $\underline{\sigma}$ over a range of frequencies, i.e.

$$\min\{\max(\sigma(E(j\omega)))\} \quad \omega_1 \leq \omega \leq \omega_2$$

3. Minimize weighted sum of $\bar{\sigma}$ and $\underline{\sigma}$ over a range of frequencies, i.e.

$$\min\left\{\sum_{i=1}^N a_i \sigma(E(j\omega_i))\right\}$$

An examination of the 10 flexure modes (see Appendix A) indicated that modes 2 and 5 were primarily fuselage bending modes and mode 7 was exclusively a tail mode. Therefore these three modes were not considered further in the analysis. Mode 1 was primarily the first wing bending mode, mode 2 was the second wing bending mode, and mode 6 was the first wing torsion mode. These modes were obviously important in modeling flutter and were retained. Modes 3 and 8 included wing tip bending and it was felt that these modes should also be analyzed further. Mode 9 was primarily wing bending and mode 10 was primarily wing torsion and these modes were also retained. Thus seven flexure modes, the 1st, 3rd, 4th, 6th, 8th, 9th and 10th were included in the initial flutter analysis. The loci of the eigenvalues of these modes as velocity was varied are shown in Fig. 1. The first mode flutters at a velocity of about

241 m/s (M=0.75) at a frequency of 120 rad/s. Modes 3 and 8 are insensitive to changes in velocity indicating that they are primarily vibrational modes; however, modes 9 and 10 do vary somewhat with velocity. Eigenvalues were calculated using models in which various modes were deleted. Deletion of the 3rd, 8th, 9th and 10th modes has very little affect on the lower modes.

3 Mode Model	
Mode No	Eigenvalue
6	-15.7+j216.1
4	-94.3+j106.4
1	+39.9+j118.0

Table 3 Eigenvalues of DAST ARW-2 Flight Test Vehicle at M=0.86 h=4572m Based on 3 Flexure Modes.

Basic flutter characteristics could be accurately modeled with only three modes corresponding to the first and second bending (modes 1 and 4) and first torsion (mode 6), and computational expense could be reduced significantly by reducing the order of the system model; therefore, the design studies were based on a structural model containing only these three flexure modes. The loci of the eigenvalues of the three mode model as velocity is varied are shown in Fig. 2.

Equations 2-6 can be combined to give the equations which describe the wing, control surfaces and actuators, and wind gust in vector-matrix form as

$$\dot{\mathbf{x}} = \mathbf{Ax} + \mathbf{Bu} + \Gamma\eta \quad (8)$$

The 18th order state vector, \mathbf{x} , consists of (1) the dis-

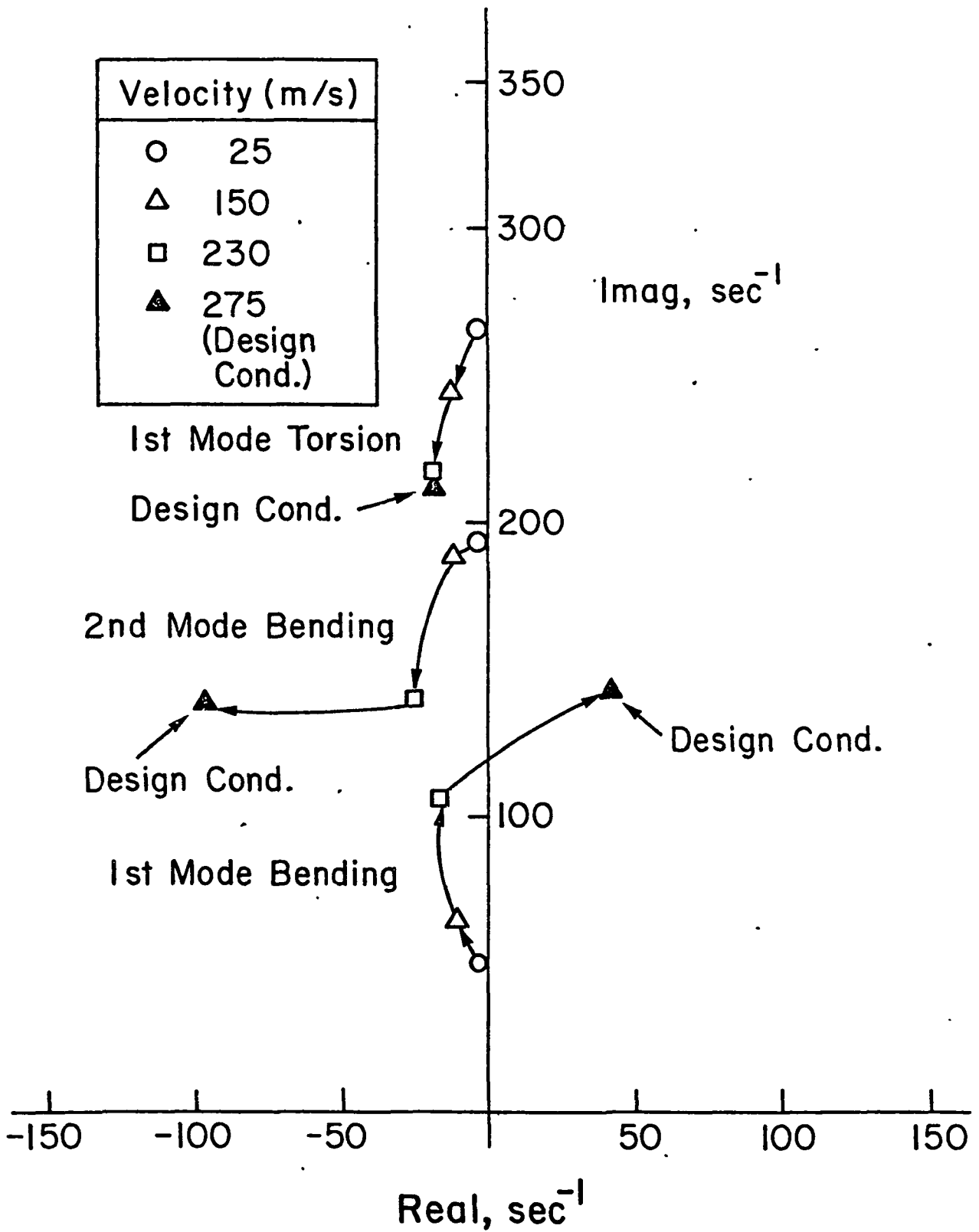


Fig. 2 Root Locus of Three Mode Model of Uncontrolled Wing as Velocity is Varied

placements and velocities associated with the three flexure modes, (2) the unsteady aerodynamic lag states, (3) the states associated with the inboard and outboard control surfaces, and (4) the states associated with the wind gust model. The 2nd order control vector, u , consists of the inputs to the inboard and outboard aileron/actuators and the zero mean white noise input, η , drives the wind gust model. (See Appendix B for a more detailed discussion of Eq. 8)

III. Eigenspace Design

Theory

Moore [1] and others have shown how feedback can be used to directly place eigenvalues and also shape eigenvectors. If performance specifications are given or can be interpreted in terms of desired closed loop eigenvalues and eigenvectors, then ES techniques can provide a natural design procedure. If performance specifications cannot be clearly stated in terms of closed loop eigenvalues and eigenvectors, for example specifications on rms responses, it may be necessary to iterate eigenvalues and eigenvectors until performance specifications are met.

Before discussing the details of the ES flutter controller, the theoretical basis of ES design methodology will be discussed. Consider a system modeled in state variable form as

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}u + \mathbf{\Gamma}\eta$$

with measurements

$$y = \mathbf{C}\mathbf{x}$$

where the state $x \in \mathbb{R}^n$, the feedback control $u \in \mathbb{R}^m$, and the output $y \in \mathbb{R}^p$. If we express u in output feedback form

$$u = Ky = KCx$$

the the closed loop system response is

$$\dot{x} = (A + BKC)x + \Gamma\eta \quad (9)$$

The ES design procedure consists, of determining a gain matrix, K , such that for all desired closed loop eigenvalue (λ_i) and eigenvector (v_i) pairs

$$(A + BKC)v_i = \lambda_i v_i \quad (10)$$

Introduce $w \in \mathbb{R}^m$ where

$$w_i = KCv_i \quad (11)$$

The problem becomes one of determining w_i such that

$$(\lambda_i I - A)v_i = Bw_i$$

where K is determined from Eq. 11.

Case 1: Full State Feedback

If all the states of the system are available, the feedback control law becomes (assuming $p=n$ and letting $C=I$ without loss in generality)

$$u = Kx$$

The design problem is to find w_i and K such that

$$(\lambda_i I - A)v_i = Bw_i \quad (12)$$

and

$$w_i = Kv_i \quad (13)$$

For the entire collection of closed loop eigenvalues and eigenvectors Eq. (12) becomes

$$V\Lambda - AV = BW \quad (14)$$

Likewise from Eq. (13)

$$W = KV \quad (15)$$

therefore

$$K = WV^{-1} \quad (16)$$

Case 2: Output Feedback ($p < n$)

$$y = Cx$$

In this case Eq. (15) becomes

$$W = KCV$$

Since C is of rank $p < n$ a unique solution for K is impossible.

The alternative is to place only p eigenvalue-eigenvector pairs, i.e. choose p λ_i 's and the corresponding p columns of V and solve for K such that

$$K = W(CV)^{-1}$$

The key design issue is to find W which satisfies Eq. (14).

In general one cannot completely satisfy both exact eigenvalue and eigenvector placement.

Case 3: Single Input Systems

For single inputs Eq. (12) reduces to

$$(\lambda_i I - A)v_i = bw_i$$

where w_i is a scalar. This single variable cannot be adjusted to place n parameters on the left hand side, therefore, the single input case only involves pole placement with arbitrary eigenvector position. The gain K is unique in this case.

Case 4: Multiple Input Systems

The real benefit of eigenspace techniques is realized when more than one control is available. If the rank of B

is n , i.e., one independent control for each state then from Eq. (12)

$$w_i = B^{-1}(\lambda_i I - A)v_i$$

for each desired λ_i and v_i of the closed loop system.

Practically speaking, however, one has fewer controls than states and exact placement of λ_i and v_i for all $i=1,2,\dots,n$ is impossible. The design procedure then becomes one of choosing portions of v_i to eliminate certain state responses from a mode while emphasizing others and letting other responses react arbitrarily. Assuming the system is controllable and $\text{rank}(B) = m < n$, only m free parameters can be specified. If it is desired to change an eigenvalue associated with a controllable mode, then for the new eigenvalue, λ_i , the matrix $(\lambda_i I - A)$ is nonsingular. Thus

$$v_i = (\lambda_i I - A)^{-1} B w_i \quad (17)$$

or

$$v_i = L_i w_i \quad (18)$$

where

$$L_i = (\lambda_i I - A)^{-1} B$$

Since there are not enough independent controls available to arbitrarily place all λ_i 's and v_i 's, the w_i 's are selected to minimize the following least square performance index

$$J_i = (v_{d_i} - v_i)^* P_i (v_{d_i} - v_i)$$

where P_i is used to emphasize certain components of v_{d_i} .

Solving for w_i which minimizes J_i produces an optimal psuedo-

inverse

$$w_i = (L_i^* P_i L_i)^{-1} L_i^* P_i v_{d_i}$$

and v_i is given by Eq. (18).

If certain eigensolutions are not to be altered (e.g. actuator poles) or if λ_i is uncontrollable then

$$w_i = 0$$

and Eq. (12) is satisfied. The general design procedure used is to vary P_i , λ_i , and v_{d_i} until performance specifications on rms values of the state and control are met.

The above analysis is valid for complex or real eigensolutions. If an entirely real K matrix is desired then the the problem must be decoupled into its real and imaginary parts. If

$$\lambda = \lambda_R + j\lambda_I$$

$$v = v_R + jv_I$$

$$w = w_R + jw_I$$

then for real A

$$(\lambda_i I - A)v_i = Bw_i$$

becomes in entirely real terms

$$\begin{bmatrix} (\lambda_R I - A) & -\lambda_I I \\ \lambda_I I & (\lambda_R I - A) \end{bmatrix} \begin{bmatrix} v_R \\ v_I \end{bmatrix} = \begin{bmatrix} Bw_R \\ Bw_I \end{bmatrix} \quad (19)$$

and

$$\begin{bmatrix} v_R \\ v_I \end{bmatrix} = \begin{bmatrix} (\lambda_R I - A) & -\lambda_I I \\ \lambda_I I & (\lambda_R I - A) \end{bmatrix}^{-1} \begin{bmatrix} Bw_R \\ Bw_I \end{bmatrix} \quad (20)$$

or

$$\begin{bmatrix} v_R \\ v_I \end{bmatrix} = L \begin{bmatrix} w_R \\ w_I \end{bmatrix}$$

where

$$L = \begin{bmatrix} (\lambda_R I - A) & -\lambda_I I \\ \lambda_I I & (\lambda_R I - A) \end{bmatrix}^{-1} \begin{bmatrix} B & 0 \\ 0 & B \end{bmatrix} \quad (21)$$

Then the optimal psuedo-inverse solution in entirely real terms is

$$w_i = (L_i^T P_i L_i)^{-1} L_i^T P_i v_{d_i} \quad (22)$$

where

$$w_i = \begin{bmatrix} w_{R_i} \\ w_{I_i} \end{bmatrix}$$

$$v_{d_i} = \begin{bmatrix} v_{d_{R_i}} \\ v_{d_{I_i}} \end{bmatrix}$$

$$P_i = \begin{bmatrix} P_{R_i} & 0 \\ 0 & P_{I_i} \end{bmatrix}$$

To determine K with λ_1 and λ_2 complex conjugates, we must solve

$$K[v_{R_1} + jv_{I_1}, v_{R_1} - jv_{I_1}, V] = [w_{R_1} + jw_{I_1}, w_{R_1} - jw_{I_1}, W] \quad (23)$$

where V and W are the remaining v_i 's and w_i 's. Post multiplying both sides of this equation by

$$R = \begin{bmatrix} \frac{1}{2} & -j\frac{1}{2} & 0 \\ \frac{1}{2} & +j\frac{1}{2} & 0 \\ 0 & 0 & I \end{bmatrix} \quad (24)$$

yields

$$K[v_{R_1}, v_{I_1}, V] = [w_{R_1}, w_{I_1}, W] \quad (25)$$

Now the left hand eigenvector matrix is nonsingular so

$$K = [w_{R_1}, w_{I_1}, W][v_{R_1}, v_{I_1}, V]^{-1} \quad (26)$$

This procedure can be applied to any complex conjugate pair. It should be noted, however, that the transformation matrix R is not unique so neither is K.

Application to Active Flutter Suppression

Using the above procedure a flutter control system was designed. The initial ES controller was designed by rotating the unstable eigenvalues around the imaginary axis and leaving all other eigensolutions in their open loop configuration. This resulted in acceptable rms control surface activity at the flutter condition and a stable response at the gust test condition. Although the ES design stabilized the wing at the gust test condition, the maximum allowable values for the rms inboard deflection rate and outboard deflection and deflection rate were exceeded. It was felt that the performance at the gust test case might be enhanced by redesign of the control system. Since the aircraft exhibits satisfactory response at velocities somewhat less than the flutter speed, it was decided to use ES design techniques to force the closed loop eigenvalues at the design velocity (275 m/s) to be the same as the open loop eigenvalues at the velocity of 200 m/s (this is 20% less than the flutter speed) and to force the closed

loop eigenvectors at 275 m/s to approach the open loop eigenvectors at 200 m/s. The control was designed to retain the eigenvalues and eigenvectors associated with the actuators, unsteady aerodynamics, and gust model at their open loop values so that energy would not be transferred to these modes. (The gust states are uncontrollable and the gust eigenvalues cannot, in any case, be moved.) The ability to keep poles in desired locations is one of the key advantages of the ES design.

In the open loop condition, the actuator dynamics are decoupled from the aeroelastic modes; therefore, the components of the aeroelastic eigenvectors in the direction of the actuator states are zero. Thus driving the closed loop aeroelastic eigenvectors to exactly their open loop values would decouple the actuators from the aeroelastic response of the wing. This is obviously not desirable since the wing would be uncontrolled; however, it is possible to reduce control surface activity and still stabilize the wing by using the weighting matrices P_i to penalize large values of the components of the closed loop aeroelastic eigenvectors in the actuator directions. Initially all closed loop aeroelastic eigenvalues at 275 m/s were placed at the locations of the open loop aeroelastic eigenvalues at 200 m/s and the weighting matrices were chosen to be identity matrices. This resulted in a reduction of control surface activity of less than 7% at the gust test condition. The rms inboard aileron deflection rate was still over three

times its allowable value while the outboard aileron deflection and rates were about twice their allowable values. A gust of 5.7 m/s would saturate the inboard control surface while the outboard surface remained unsaturated since its deflection and rate were about two-thirds of the allowable values.

It was decided to shift some of the control effort from the inboard to the outboard aileron in an attempt to increase the gust velocity at which the system would saturate. This was accomplished by increasing the weights on the components of the aeroelastic eigenvectors in the inboard actuator directions while retaining all other weights at one. Values of these weights were increased to 2.5×10^3 yielding the results in Table 4. The inboard rate was reduced substantially without excessively increasing outboard activity. Specifications on inboard rate and outboard deflection and rate were still not met at the gust test condition but all of the quantities were about twice their maximum values. Thus both control surfaces would saturate at about the same gust velocity (8.3 m/s). Since it proved impossible to further reduce outboard surface activity by adjusting the weighting matrices P_i , the ES controller resulting from eigenvector shaping to reduce inboard rate was chosen for further evaluation.

Table 4

Comparison of RMS Control Surface Activity
for Various Controller Designs and Flight Conditions

Controller Design	Inbd Defl (Deg)		Inbd Rate (Deg/s)		Outbd Defl (Deg)		Outbd Rate (Deg/s)	
	Flutter	Gust Test	Flutter	Gust Test	Flutter	Gust Test	Flutter	Gust Test
<i>Unstable Roots Rotated About Imag. Axis</i>								
ES	0.5	5.0	108.0	412.0	3.3	31.4	509.0	1464.0
<i>Unstable Roots Rotated About Imag. Axis</i>								
LQ	1.8	*	271.0	*	2.8	*	407.0	*
<i>Eigenvector shaping to Reduce Inbd Rate (Final ES)</i>								
	0.9	8.1	86.0	279.7	4.7	31.2	612.0	1464.0
<i>Weighting Inbd Rate in Perf Index (Final LQ)</i>								
	0.7	4.6	85.0	284.1	3.1	30.2	486.0	1335.1
<i>Final ES Inbd Failed</i>								
	-	-	-	-	4.4	26.5	572.0	1427.0
<i>Final LQ Inbd Failed</i>								
	-	-	-	-	3.5	30.0	519.0	1410.0
<i>Max Allowable</i>								
	10.0	10.0	130.0	130.0	15.0	15.0	740.0	740.0

*Unstable

Results

In addition to the ES design, a flutter controller was designed using Linear Quadratic(LQ) optimal control theory. LQ control theory has been discussed extensively in numerous texts, for example Ref. 13. The basic LQ design procedure consists of selecting quadratic weighting matrices Q and R in a scalar performance index

$$J = \frac{1}{2} \int_0^{\infty} [x^T Q x + u^T R u] dt \quad (27)$$

The control which minimizes this performance index is given as

$$u = Kx$$

where K is the solution of a matrix Ricatti equation.

It is well known that if Q is set equal to zero and R equal to the identity matrix in the performance index given by Eq. (27), then all stable eigenvalues will remain unchanged and all unstable eigenvalues will be rotated about the imaginary axis [13]. Initial LQ design was performed using this approach; however, as shown in Table 4, the rms deflection rate for the inboard actuator was approximately double its allowed maximum at the flutter test condition and the controller resulted in an unstable wing at the gust test condition (note the uncontrolled wing is stable at this condition). Since all rms responses except the inboard actuator deflection rate were acceptable at the flutter test condition, only this state was weighted in the performance index. The results of varying the weight on inboard actuator deflection rate is shown in Table

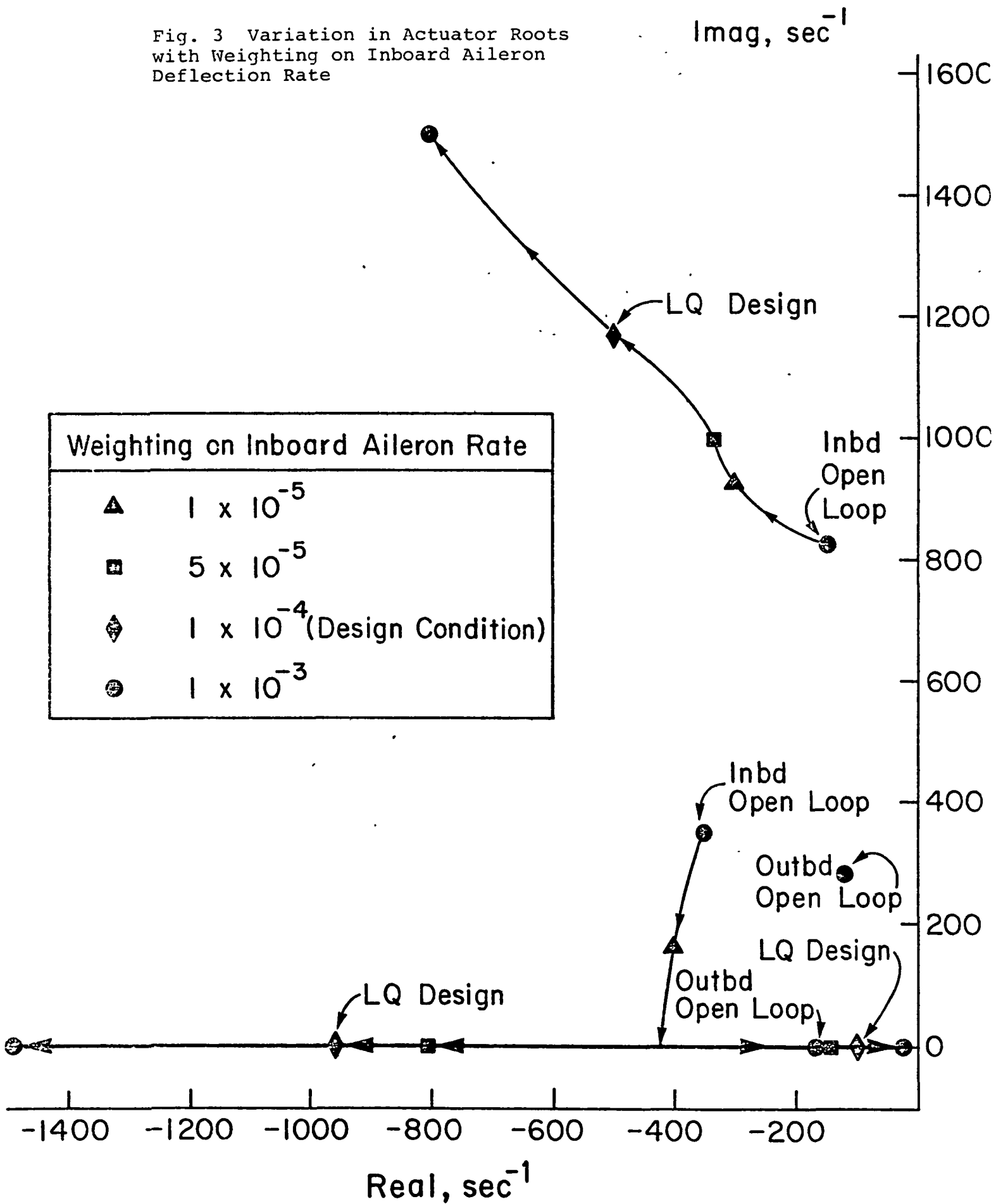
Table 5

RMS Response for Various Weights
on Inboard Actuator Rate

Weight on Inbd Deflection Rate	RMS Response (Degrees and Degrees/s)			
	Inbd Def.	Inbd Rate	Outbd Def.	Outbd Rate
0	1.60	217.6	2.73	427
1×10^{-5}	1.58	256.0	2.74	428
5×10^{-5}	0.92	122.0	3.02	470
1×10^{-4}	0.66	84.8	3.14	486
5×10^{-4}	0.22	26.0	3.32	514
1×10^{-3}	0.12	14.0	3.36	518
Max Allowable	10.0	130.0	15.0	740

5. The rms value of the inboard actuator deflection rate decreases fairly rapidly as its weight is increased and the rms activity level of the outboard control surface does not increase substantially. The eigenvalues associated with the flexure modes and the outboard aileron do not change as the weight on the inboard actuator rate is changed; however, as shown in Fig. 3, the eigenvalues associated with the inboard actuator change radically. The moduli of three of the roots become very large while the fourth root approaches zero. Since any large change in actuator frequency response characteristics would be difficult to obtain without substantially redesigning the actuator, a value for the weight on actuator deflection rate of 1×10^{-4} was selected. This gave acceptable rms responses at the flutter test condition and also stabilized the wing at the gust test condition and did not affect the frequency response of the inboard actuator too much. The open and closed loop inboard actuator frequency response curves for the final design are shown in Fig. 4. It can be seen that the overall gain for the closed loop system is reduced compared with the open loop response. At zero frequency the open loop gain is unity and the closed loop gain is 0.79. Thus the forward loop gain would have to be increased by 21% in order to restore the steady state gain to its open loop value. This might be difficult without modifications to existing actuator hardware.

Fig. 3 Variation in Actuator Roots with Weighting on Inboard Aileron Deflection Rate



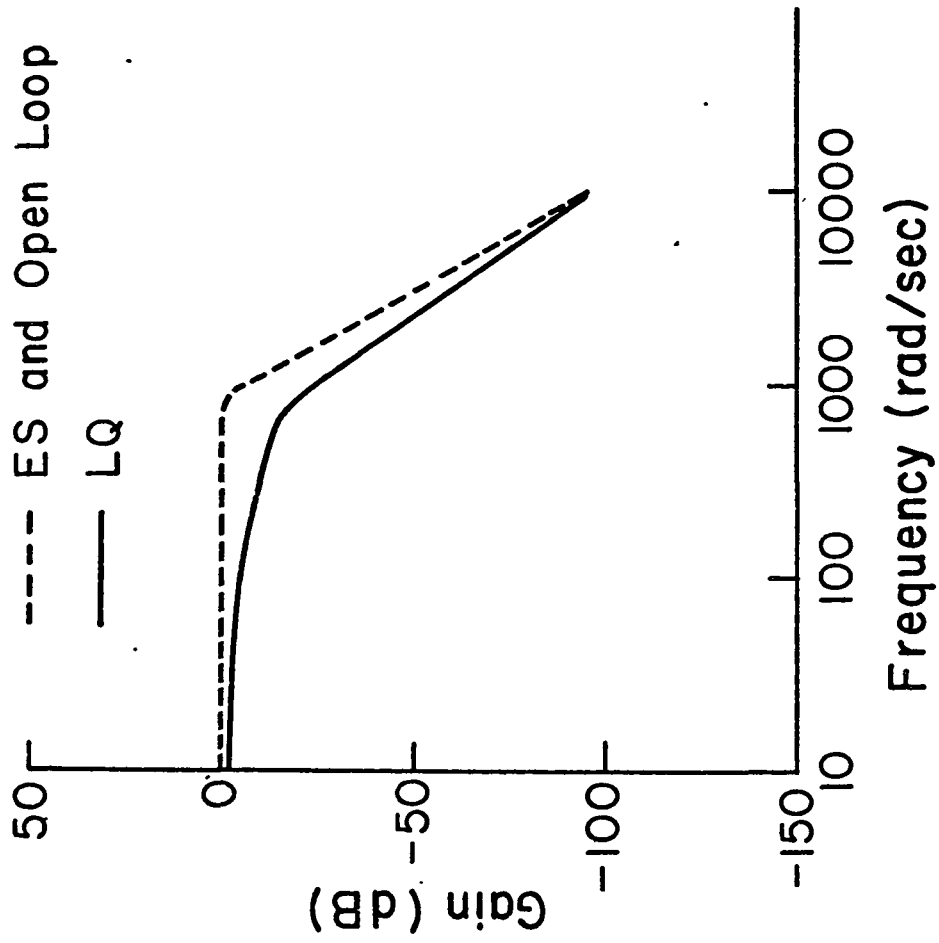


Fig. 4 Inboard Actuator Transfer Function Frequency Response

It appears to be impossible to design either an ES or LQ flutter controller which also meets the specification on rms control surface activity at the gust test condition. In the actual DAST ARW-2 vehicle, the inboard aileron is used for maneuver load alleviation but not for flutter suppression or gust load alleviation. Furthermore a different controller design is used for gust load alleviation than for flutter suppression. Thus it is not too surprising that the ES and LQ flutter controllers do not meet specifications on rms surface activity at the gust test condition. However, if the rms gust velocity at the gust condition is reduced to 8.3 m/s (about 50% of its nominal value) both the ES and LQ controllers meet all specifications on rms surface activity. Table 6 shows that both controllers provide some torsional load alleviation at this flight condition. The bending and torsional stresses and shear force were calculated at nine stations on the wing (station 1 is at the root and station 9 is near the tip). The LQ design results in the largest rms bending moments at all stations. The ES design results in bending moments which are lower than those resulting from the LQ design but higher than the uncontrolled values. The differences in bending moment in all three cases are not large. Both the LQ and ES designs result in substantial reductions (over 50%) in torsional stresses at all stations compared with the uncontrolled values. The LQ reduces torsional stresses slightly more than the ES design. The ES design results in lowest values of shear near the wing

Table 6

RMS Loads at Modified Gust Test Condition
(M=.7, h=4572m, $V_G=8.3$ m/s)

		Root		Station						Tip
		1	2	3	4	5	6	7	8	9
FINAL LQ	<i>Bending Moment (N-M)</i>	5740	4530	3426	2444	1371	889	377	83	5
	<i>Shear (N)</i>	92	82	74	63	42	36	24	6	1
	<i>Torque (N-M)</i>	36	54	73	82	73	63	52	6	2
FINAL ES	<i>Bending Moment (N-M)</i>	5558	4412	3359	2418	1374	900	389	91	5
	<i>Shear (N)</i>	87	79	71	61	42	36	24	6	1
	<i>Torque (N-M)</i>	46	63	79	85	76	65	53	7	1
OPEN LOOP	<i>Bending Moment (N-M)</i>	5545	4262	3116	2125	1115	671	248	47	3
	<i>Shear (N)</i>	98	85	75	61	38	31	18	3	1
	<i>Torque (N-M)</i>	106	130	152	156	139	121	101	24	3

root but the uncontrolled values of shear are less near the tip. The differences in shear between the LQ, ES, and uncontrolled cases are not large.

The root locus of the eigenvalues of the wing with the LQ controller is shown in Fig. 5 as velocity is varied. The wing goes unstable at a velocity of about 295 m/s (the design condition was 275 m/s and the uncontrolled flutter speed was 241 m/s). It is interesting to note that for the LQ controller, one of the roots associated with the unsteady aerodynamic lag states goes unstable resulting in zero frequency flutter. In the uncontrolled case, the first bending model goes unstable in classical coupled mode flutter (Fig. 1). The root locus for the ES design is shown in Fig. 6. For the ES controller the wing goes unstable at a velocity of about 310 m/s. As in the uncontrolled case the first bending mode goes unstable, but in the controlled case the eigenvalues associated with this mode move to the real axis where one real root goes unstable resulting in zero frequency flutter.

The results of varying altitude while maintaining Mach number constant are shown in Fig. 7. At $M=0.86$, the uncontrolled wing is unstable until an altitude of 6700 m is reached. At the same Mach number, the LQ controller results in a stable wing at all altitudes above 3200 m and the ES controller stabilizes the wing above altitudes of 2900 m. At $M=0.7$ the uncontrolled wing is stable for altitudes above 1800 m whereas the ES controller stabilizes the

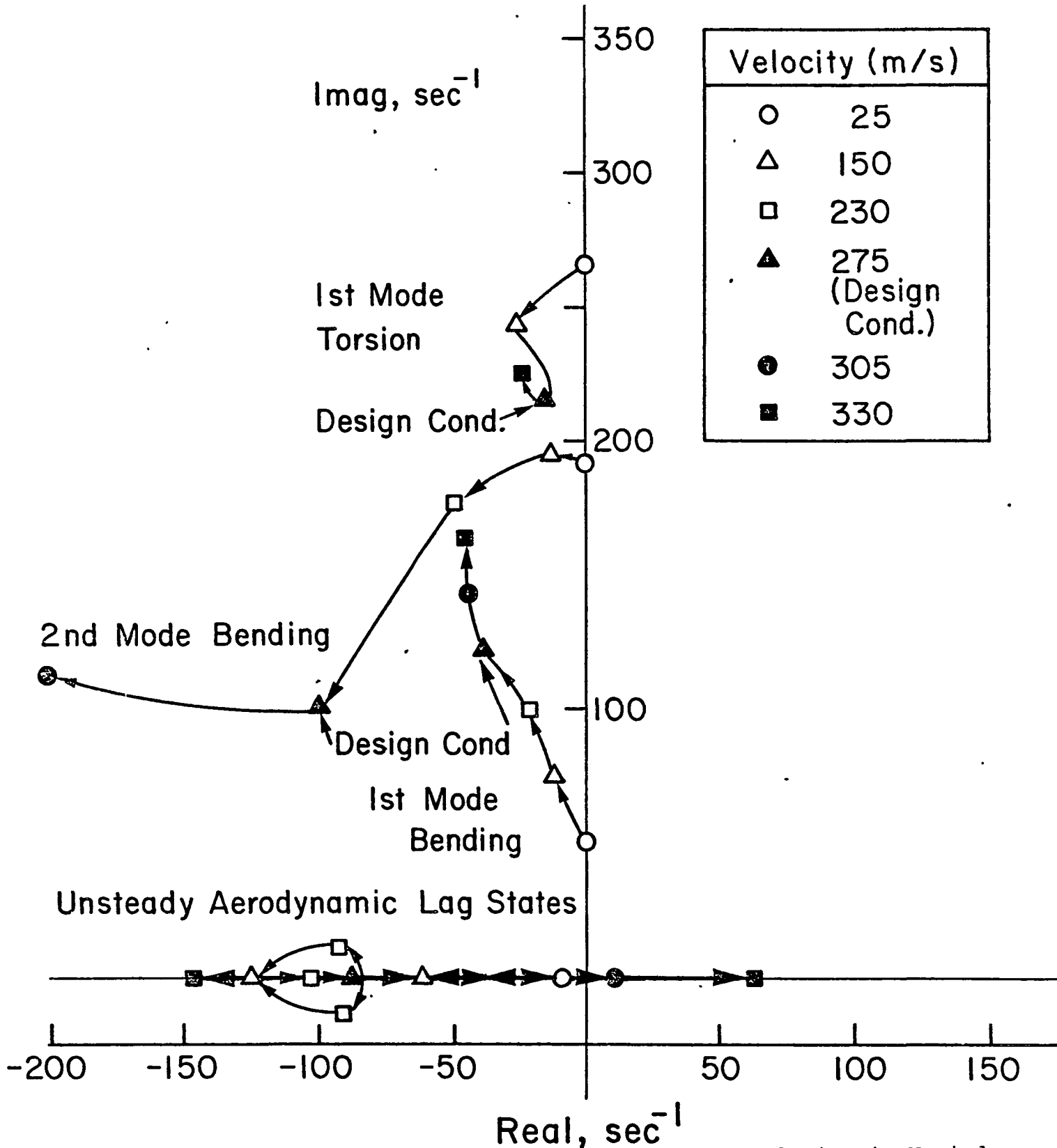


Fig. 5 Root Locus of LQ Controlled Wing as Velocity is Varied

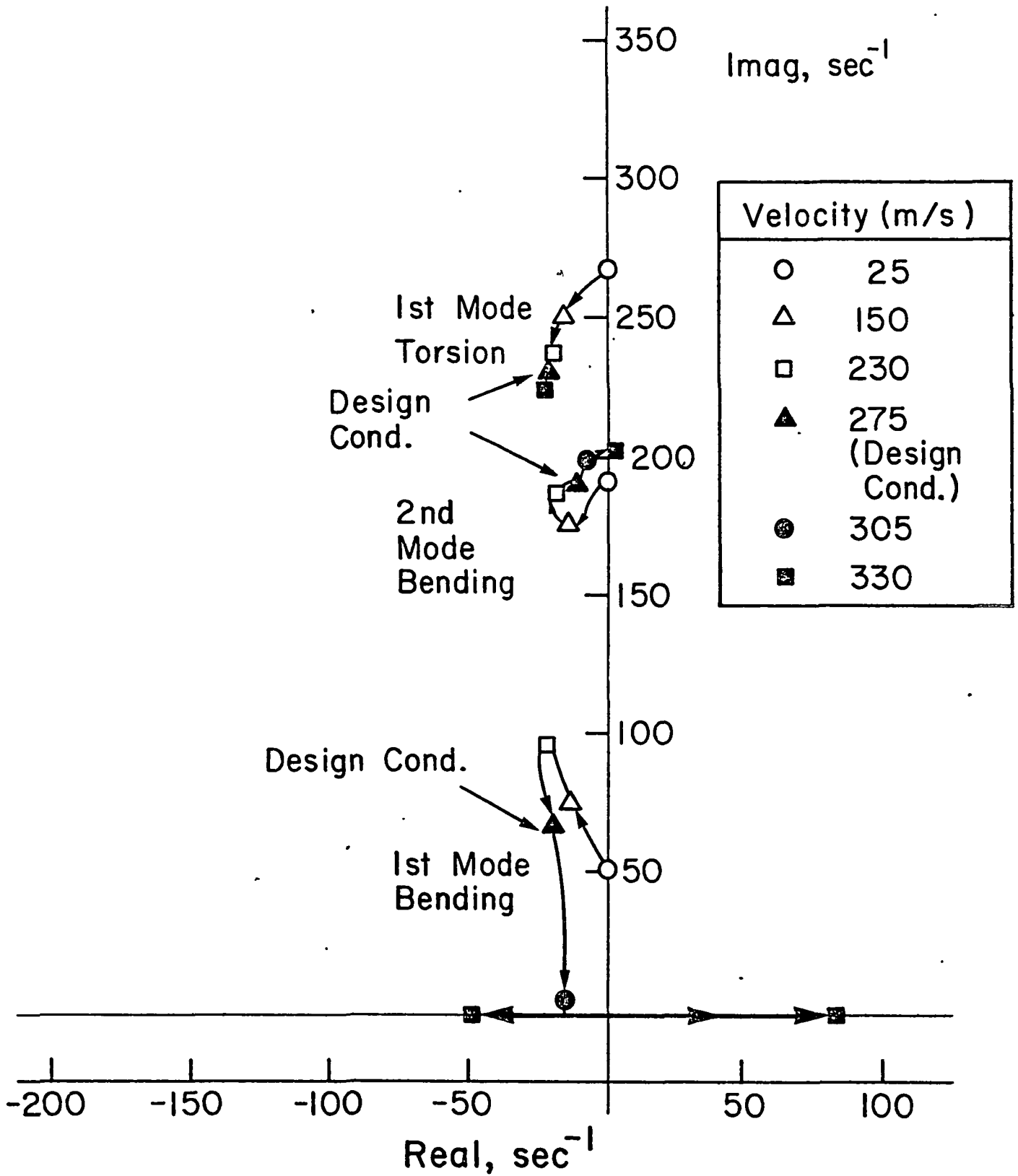


Fig. 6 Root Locus of ES Controlled Wing as Velocity is Varied

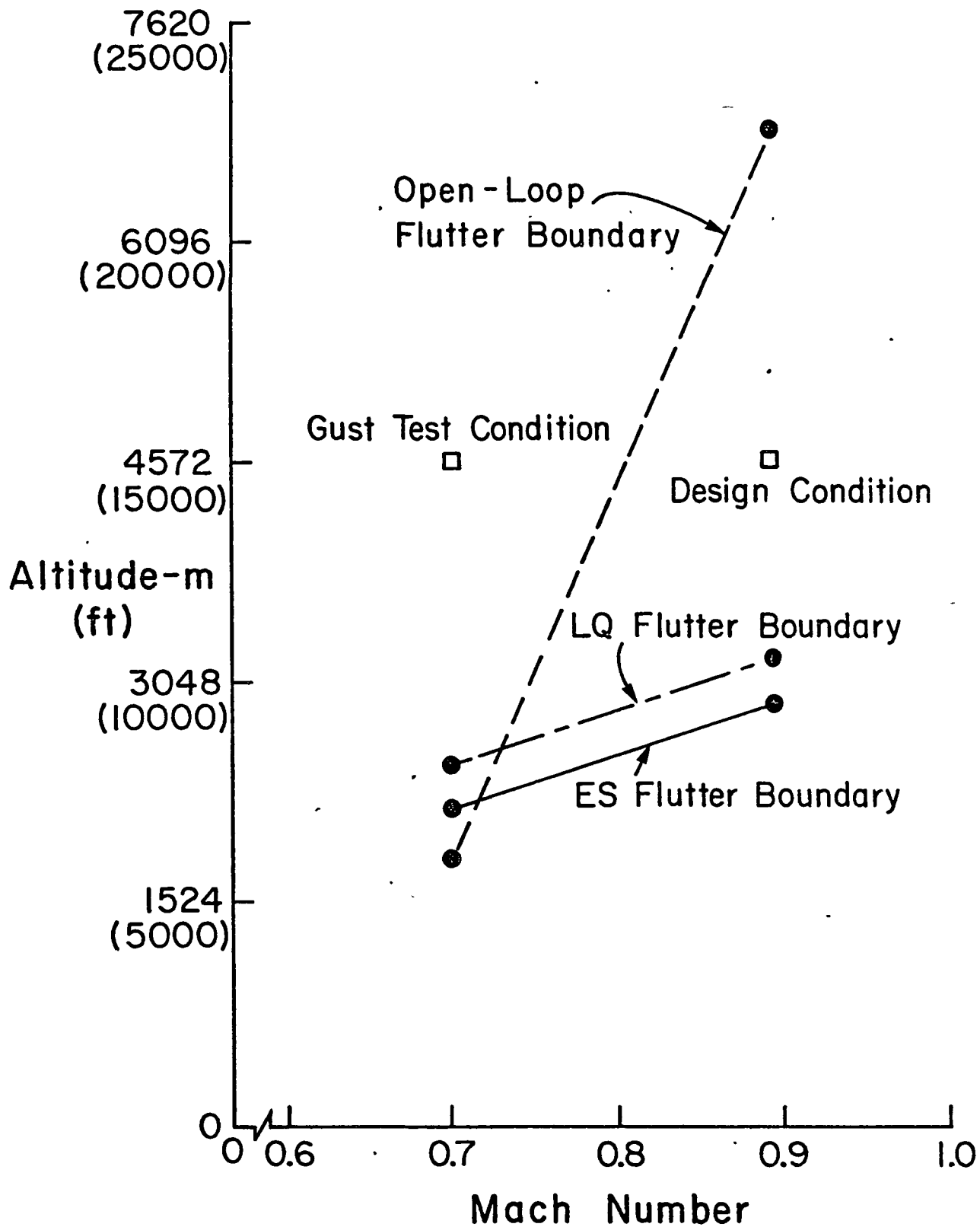


Fig. 7 Flutter Boundaries for Uncontrolled, LQ, and ES Controlled Wing

wing for altitudes above 2100 m and the LQ controller stabilizes the wing for altitudes above 2400 m.

The ability of the ES and LQ systems to stabilize the wing in case of an actuator failure yielded little difference in the performance of the two systems. In the case where an inboard actuator failure was simulated, the outboard aileron was capable of stabilizing the wing with only small increases in rms surface activity (see Table 4). When the outboard actuator failed the inboard aileron was unable to stabilize the wing in both designs.

Since the outboard aileron is critical in stabilizing the wing, it is important to examine the stability margins associated with the outboard control loop. This can be accomplished by examining the loop transfer function

$$H(s)G(s) = K^T [Is-A]^{-1}B$$

Note since $u = +Kx$, the characteristic equation is given by

$$1 - G(s)H(s) = 0$$

and the critical condition occurs when the phase angle of $H(j\omega)G(j\omega)$ is zero. For the LQ design

$$H(s)G(s)_{LQ} = \frac{-148.8(s-2.6)}{(s^2-79.8s+(124.5)^2)}$$

It is interesting to note that the actuator poles, the poles associated with the unsteady aerodynamic lag states and all of the poles associated with the stable flexure modes have been cancelled by zeros. The only remaining poles are those associated with the unstable first bending mode. Bode diagrams for the LQ loop transfer function are

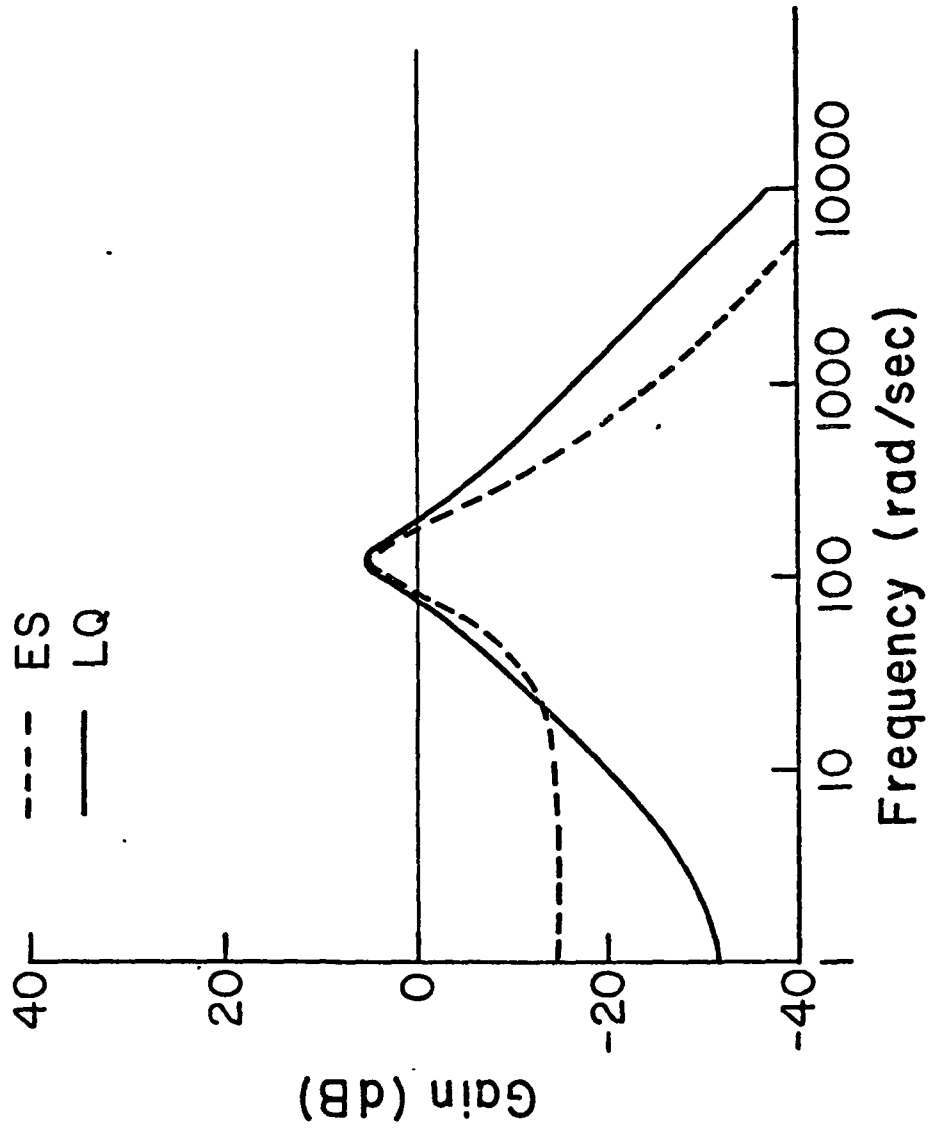


Fig. 8 Frequency Response of Magnitude of $\Pi(s)G(s)$ for LQ and ES Designs

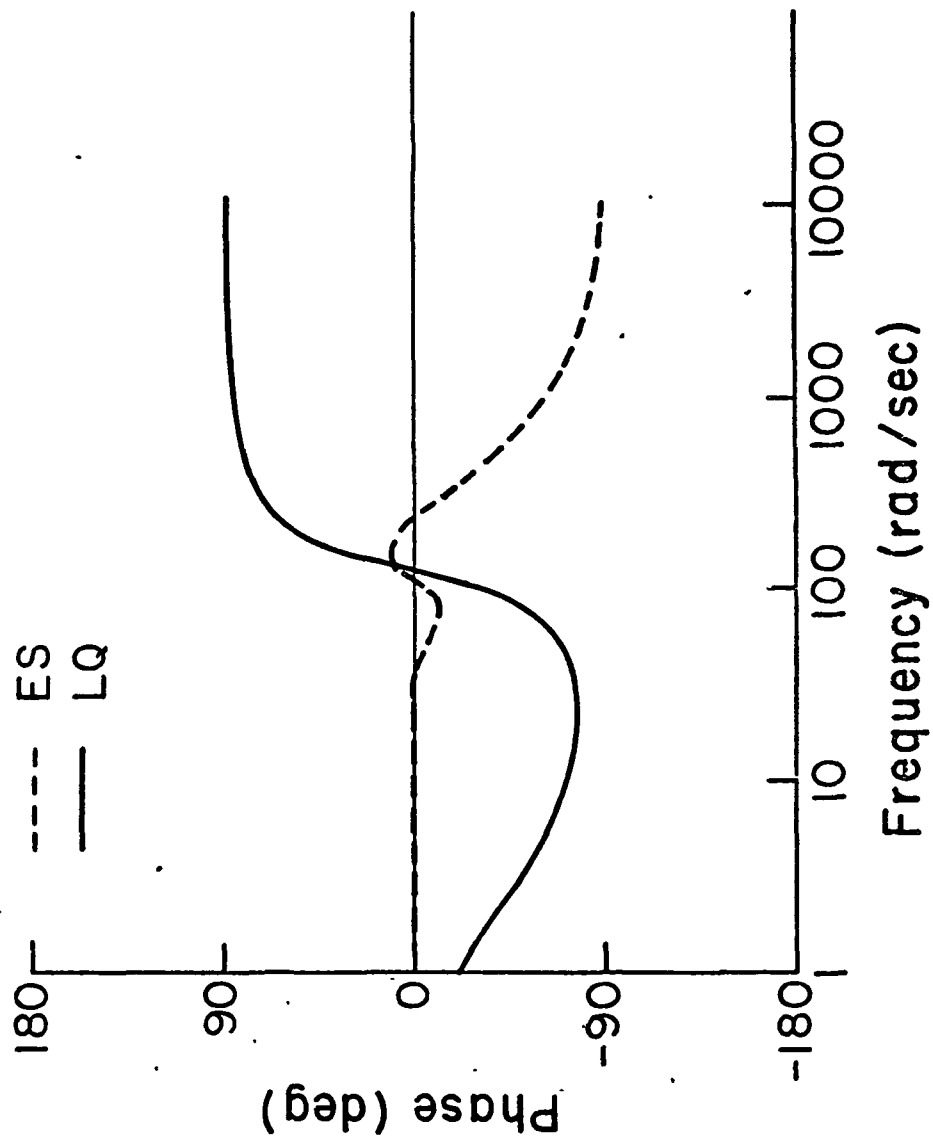


Fig. 9 Frequency Response of Phase of $H(s)G(s)$ for LQ and ES Designs

given in Figs. 8 and 9. The gain margin is 6Db and the phase margins are 60°. This is not surprising, however, since Safanov and Athans [14] show that LQ controllers yield excellent gain and phase margins.

The loop transfer function for the ES controller is

$$H(s)G(s)_{ES} = \frac{51.45(s-427.4)(s-66.5)(s+42.4)}{(s^2-79.8s+(124.5)^2)(s^2+188.6s+(142.2)^2)}$$

Again all actuator poles and poles associated with unsteady aerodynamic lag states have been cancelled by zeros. But only the poles associated with the first torsion mode have been cancelled and the unstable first bending mode and stable second bending mode remain. Bode diagrams for this transfer function are also shown in Figs. 8 and 9. Gain margins are 6 Db or better but phase margins are less than 20°. Gain versus frequency plots for the LQ and ES transfer functions are very similar for frequencies above 10 rad/s. Both curves peak at the flutter frequency with a gain of 6 Db and then roll off with increasing frequency.

Conclusions

The ES and LQ controllers give very similar results in terms of required control surface activity. At the gust test condition the ES design exhibits lower wing root bending moment and shear than the LQ design but the LQ controller provides slightly lower torsional stress. Both the ES and LQ designs provide significantly reduced torsional stress at the wing root, slightly reduced shear, and'

slightly increased bending moment compared with the open loop response. Both the LQ and ES controllers significantly increased flutter speed compared with the uncontrolled wing. The ES controller results in a slightly greater flutter speed than the LQ controller. For a fixed Mach number the ES design is stable at lower altitudes than the LQ design. The LQ design exhibits significantly better phase margins than the ES design at the flutter test condition. The gain margins are the same. Since the phase margins are determined near the flutter frequency, it is very important that the ES design be based on a model which is accurate in this frequency range. Both the LQ and ES designs exhibit excellent roll off at higher frequencies where modeling uncertainties are large.

The LQ design requires large feedback gains on the inboard actuator states. This reduces the overall inboard actuator gain. Increased forward loop gain would be required to restore open loop characteristics. This might necessitate redesign of existing actuator hardware. The ES controller does not require actuator feedback and the closed loop frequency response characteristics of the actuators is the same as the open loop case. Since the inboard actuator is not very effective for flutter control this is not an important consideration for the ARW-2, but could be critical in other applications. The ES controller requires only the structural and aerodynamic states, thus a lower-order observer could be used to realize this controller than would be required by the LQ controller. Finally the computational

algorithms required to calculate the ES controller gains are simpler and less expensive to use than those required to calculate the LQ controller gains.

The authors are currently working on ES design techniques that shape eigenvectors associated with uncontrollable states. It appears to be necessary to shape the eigenvectors associated with the gust states in order to further reduce rms control activity. Work is also in progress on improving ES stability margins and realization of ES controllers by means of observers and direct output feedback.

Doyle and Stein [15] have shown that if a Kalman Filter is used for state estimation, the robustness properties of full-state feedback can be recovered by introducing fictitious plant noise in the Ricatti equation used to obtain the filter gain matrix. As the magnitude of this plant noise approaches infinity, the filter poles asymptotically approach the transfer zeros (if the transfer zeros are in the right half of the complex plane the filter poles approach the mirror image of the zeros in the left half plane) or approach infinity in a Butterworth pattern. Since the Doyle-Stein procedure gives the location of the filter poles, it is possible to directly obtain the gain matrix which yields these poles using ES design techniques without solving the filter Ricatti equation. In the single-output, single-input case, this should yield the same result as obtained from solving the Ricatti equation; however, in the multi-input, multi-output case there is not a unique gain matrix

which yields a specified set of poles; thus, ES techniques will necessarily not yield the same results as obtained by the Doyle-Stein procedure. The utility of ES techniques in observer design should be studied in more detail. In addition, the effects on system performance of including flexure modes that were neglected during the design phase needs to be studied.

Acknowledgement

The results on ES design were obtained under Grant NAG-1-217 from NASA Langley Research Center. Mr. William Adams of NASA Langley and Mr. Sanjay Garg of the University of Minnesota were instrumental in developing and interpreting the mathematical models of the ARW-2. Mr. X.G. Li of the University of Minnesota played an important role in calculating the results for the LQ control system. Thanks are also due Dr. Thomas B. Cunningham of Honeywell, Inc. for a number of useful discussions concerning Eigenspace techniques.

References

1. Moore, B.C., "On the Flexibility Offered by Full-State Feedback in Multivariable Systems Beyond Closed Loop Eigenvalue Assignment," IEEE Trans. Auto. Control, Vol. 21, pp 682-691; 1976.
2. Cunningham, Thomas B., "Eigenspace Selection Procedures for Closed Loop Response Shaping with Modal Control," Proceedings IEEE Conference on Decision and Control; Dec. 1980.
3. Ostroff, A.J. and Pines, S., "Application of Modal Control to Wing - Flutter Suppression," NASA Tech Paper 1983; May 1982.
4. Newsom, J.R., "Control Law Synthesis for Active Flutter Suppression Using Optimal Control Theory," Journal of Guidance and Control, Vol. 2; Sept. - Oct. 1979 pp 388-394.
5. Abel, I., Newsom, J.R., and Dunn, H.J., "Application of Two Synthesis Methods for Active Flutter Suppression Using Linear Quadratic Gaussian Control Theory," AIAA Paper 79 - 1633; Aug. 1979.
6. Mahesh, J.K., Stone, C.R., Garrard, W.L., and Dunn, H.J., "Control Law Synthesis for Flutter Suppression Using Linear Quadratic Gaussian Control Theory," Journal of Guidance and Control, July - Aug., 1981, pp 415-422.
7. Mukhopadhyay, V., Newsom, J.R., and Abel, I., "Reduced-Order Optimal Feedback Control Law Synthesis for Flutter Suppression," Journal of Guidance and Control, July - Aug. 1981, pp 382-395.

8. Gangsaas, D., Ly, U., and Norman, D.C., "*Practical Gust Load Alleviation and Flutter Suppression Control Law Based on LQG Methodology*," AIAA Paper 81 - 0021, Jan. 1981.
9. Gangsaas, D., and Ly, U., "*Application of a Modified Linear Quadratic Gaussian Design to Active Control of a Transport Airplane*," AIAA Paper 79 - 1746, Aug. 1979.
10. Dowell, Earl H., "*A Simple Method for Converting Frequency Domain Aerodynamics to the Time Domain*," NASA Tech Memo 81844, Oct. 1980.
11. Noble, Ben, and Daniel, James W., "*Applied Linear Algebra*," 2nd Edition, Sections 9.6 and 11.6, Prentice Hall, 1977.
12. Adams, William M., Jr. and Tiffany, Sherwood H., "*An Updated Comparison of NASA/Boeing Predictions of the Open Loop, Symmetric Flutter Characteristics of the DAST ARW-2*," APCO TM 81-1, NASA Langley Research Center, Hampton VA, Jan. 1981.
13. Kwakernaak, H., and Sivan, Linear Optimal Control Systems, John Wiley and Sons, New York, 1972.
14. Safanov, M.G., and Athans, M., "*Gain and Phase Margins of Multi-Loop LQG Regulators*," IEEE Trans. Auto. Control, Vol. 22, pp 173-179, April 1977.
15. Doyle, J.C., and Stein, G., "*Multivariable Feedback Design: Concepts for a Classical Modern Synthesis*," IEEE Trans. Auto. Control, Feb. 1981.

Publications Issued During the Course of this Research

Active Flutter Suppression Using Eigenspace*
and Linear Quadratic Design Techniques

William L. Garrard and Bradley S. Liebst

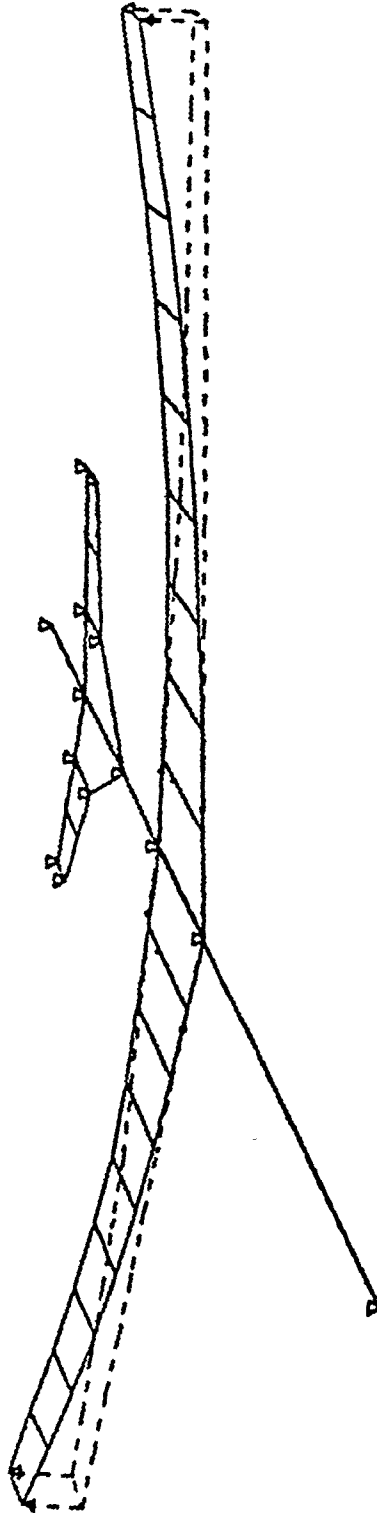
University of Minnesota
Minneapolis, Minnesota

Abstract

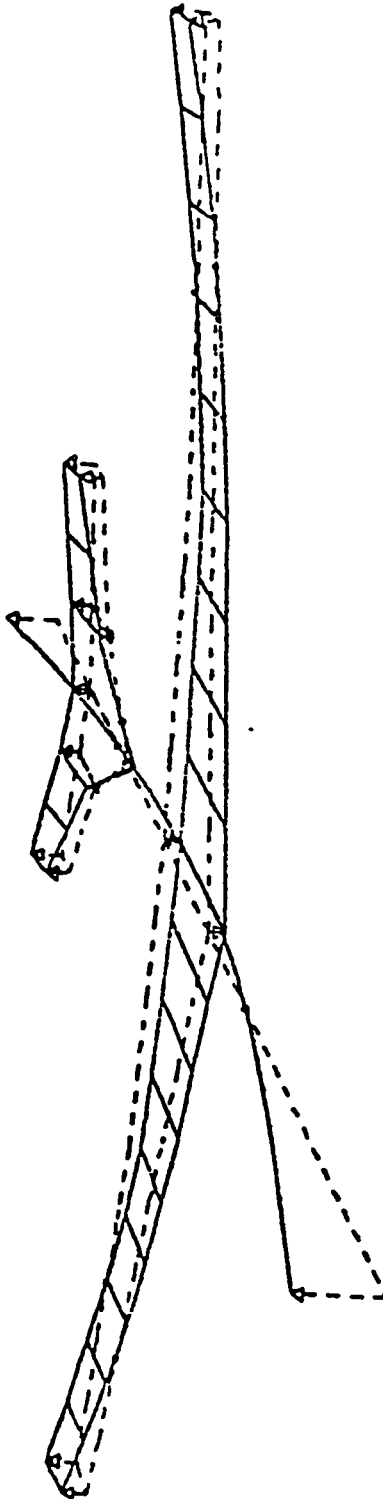
Eigenspace (ES) and Linear Quadratic (LQ) techniques are used to design an active flutter suppression system for the DAST ARW-2 flight test vehicle. The performance of the ES and LQ controllers are very similar in meeting control surface activity specifications. The ES controller provides reduced wing root bending moment and shear but torsional stress is slightly higher than with the LQ controller. The ES controller also results in improved flutter boundaries compared with the LQ controller. The LQ controller exhibits significantly better phase margins at the flutter condition than does the ES controller but the LQ design requires large feedback gains on actuator states while the ES does not. This results in reduced overall actuator gain for the LQ design.

*This paper has been submitted to the AIAA Journal of Guidance, Control, and Dynamics.

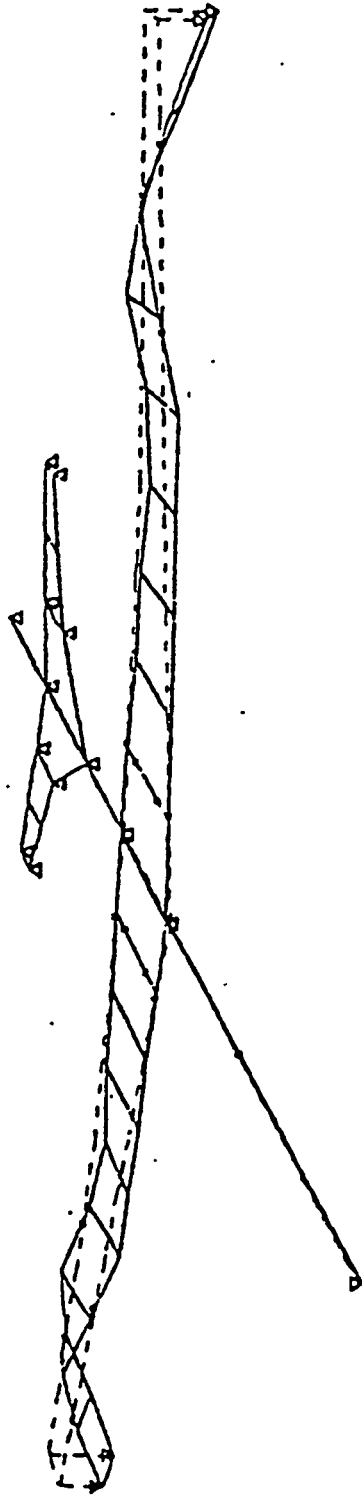
Appendix A
Flexure Mode Shapes



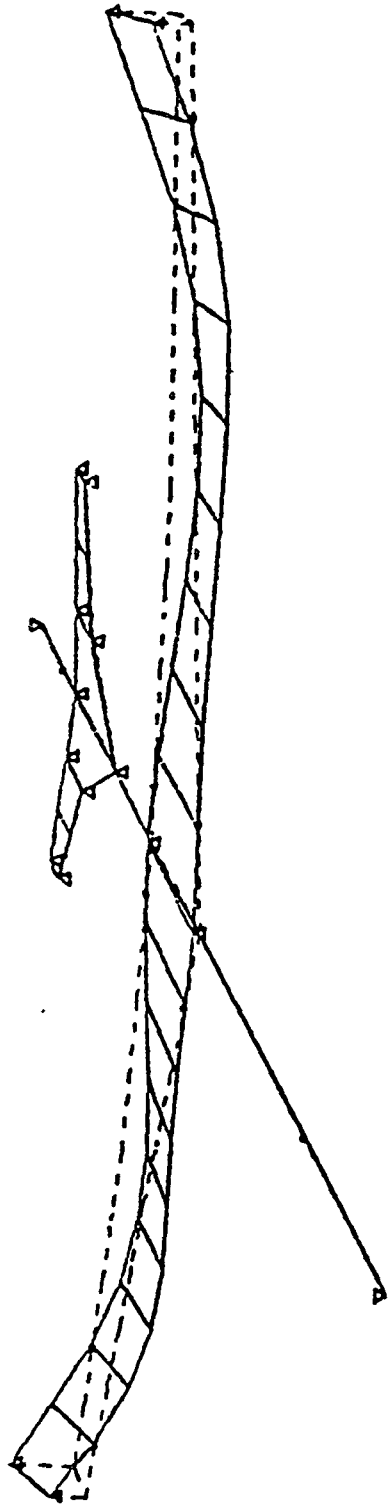
(a) mode 1, frequency = 8.44 Hz, generalized mass = 0.0169 lb-sec²/in



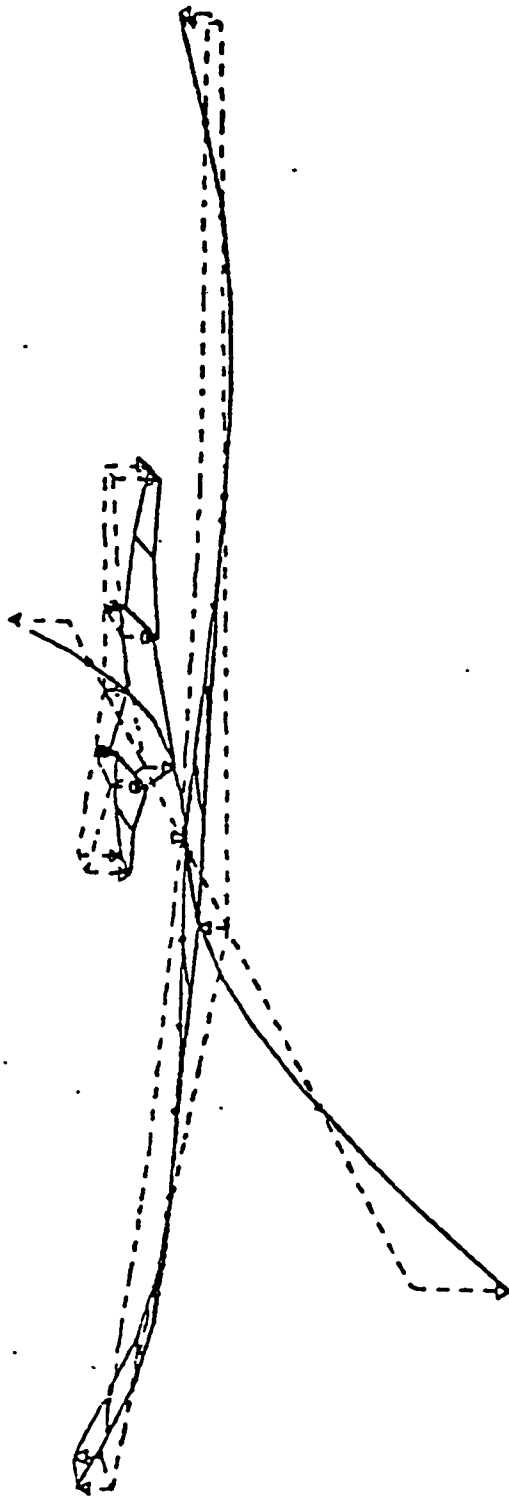
(b) mode 2, frequency = 15.7 hz, generalized mass = 0.326 lb-sec²/in



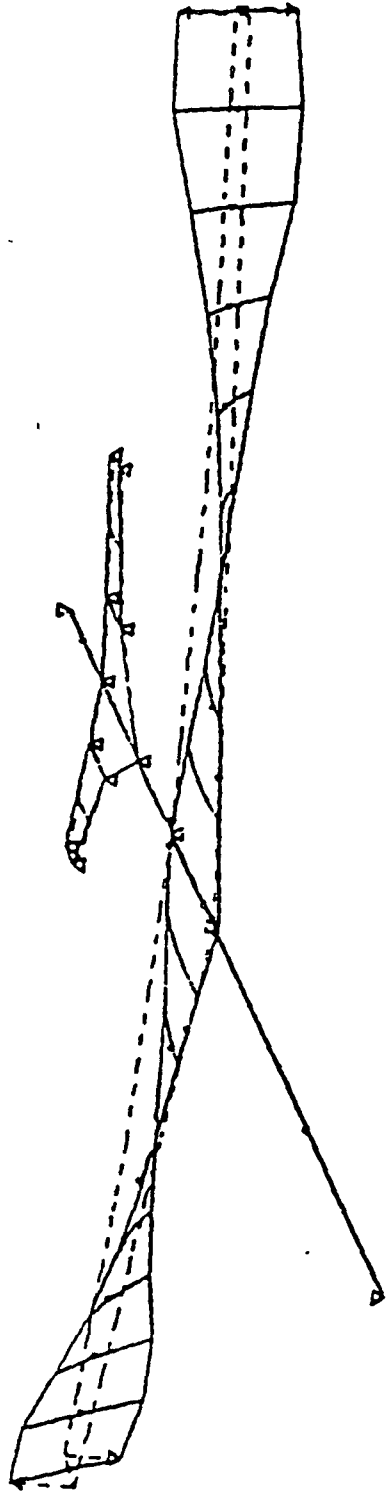
(c) mode 3, frequency = 23.7 hz, generalized mass = 0.0153 lb-sec²/in



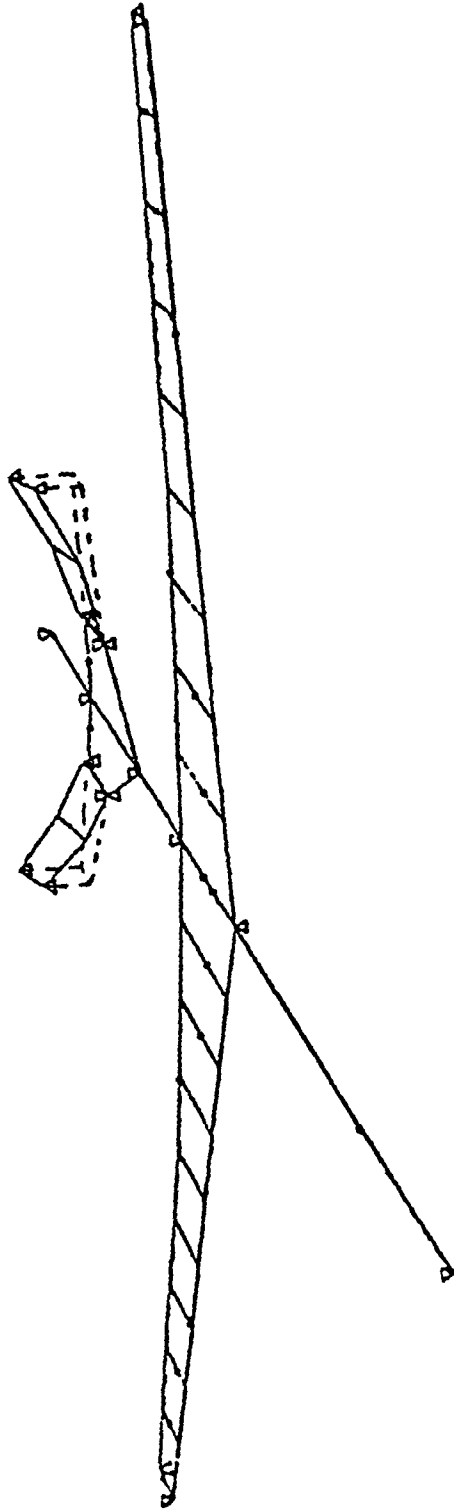
(d) mode 4, frequency = 31.7 hz, generalized mass = 0.0158 lb-sec²/in



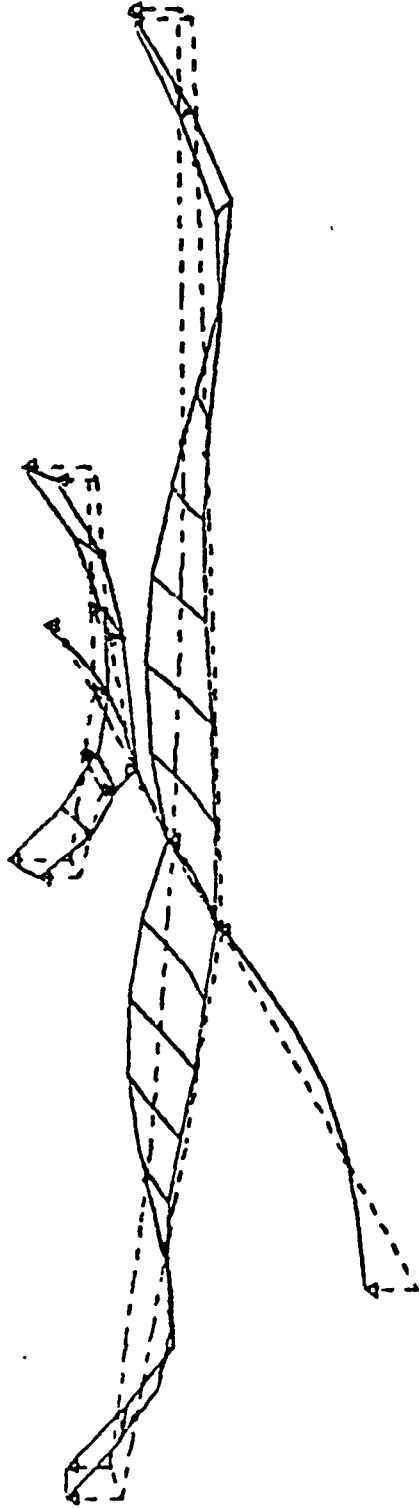
(e) mode 5, frequency = 39.3 Hz, generalized mass = 0.178 lb-sec²/in



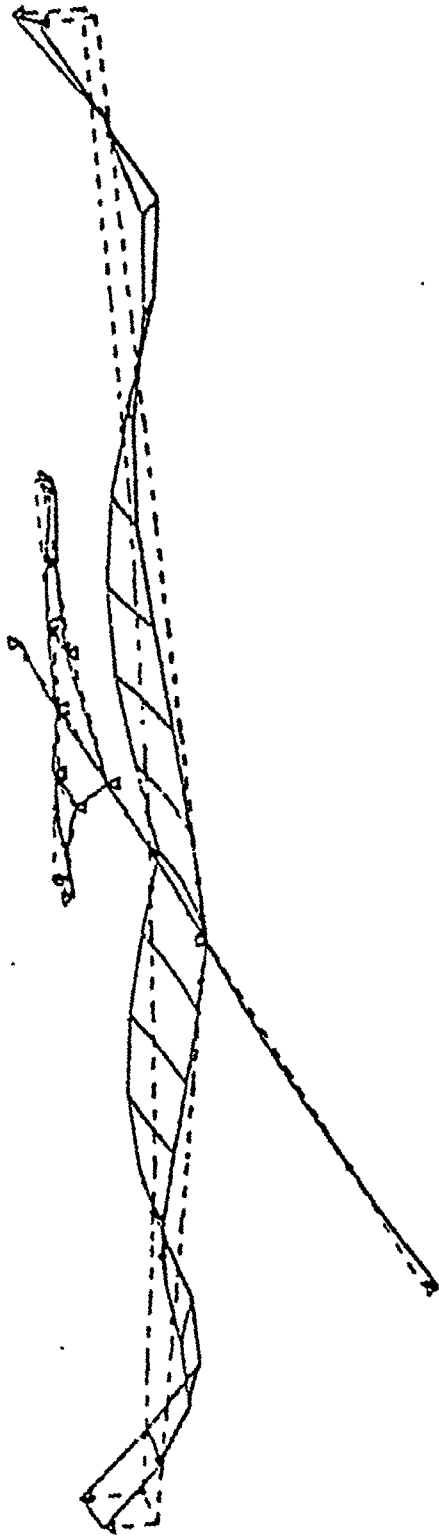
(f) mode 6, frequency = 44.6 Hz, Generalized mass = 0.593 lb-sec²/in



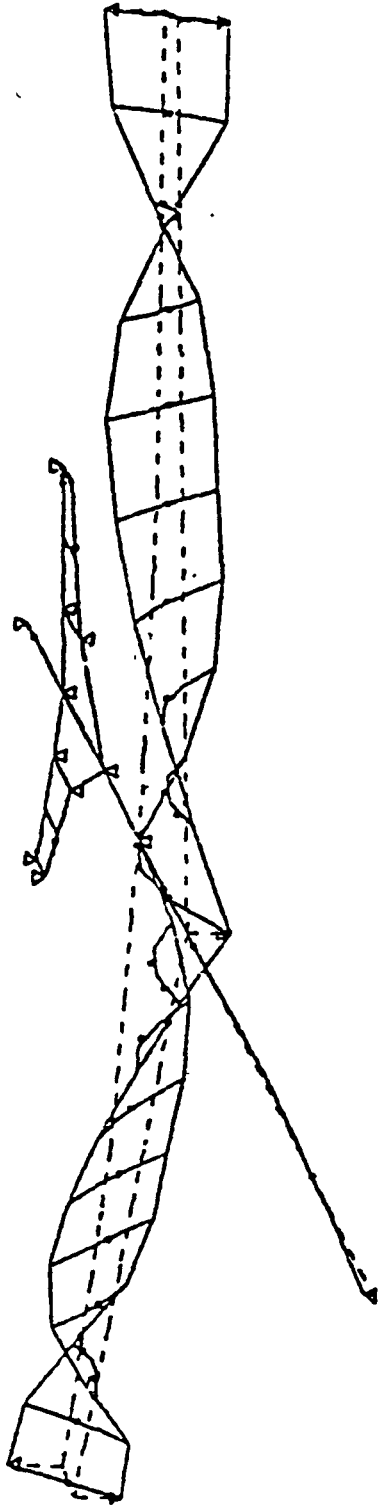
(g) mode 7, frequency = 47.3 hz, generalized mass = 0.00335 lb-sec²/in



(h) mode 8, frequency = 66.6 Hz, generalized mass = 0.0621 lb-sec²/in



(i) mode 9, frequency = 71.2 hz, generalized mass = 0.0286 lb-sec²/in



(j) mode 10, frequency = 80.5 hz, generalized mass = 0.696 lb-sec²/in

Appendix B

Program Descriptions and Listings

FLUTTER

This program generates the coefficient matrices of the first order state equation

$$\dot{\bar{x}} = A\bar{x} + Bu + \Gamma\eta$$

This first order form is derived from the second order form

$$\begin{aligned} & [M_{xx} + \bar{q} A_2^x (\frac{\bar{c}}{2V})^2] \ddot{\bar{x}} + [C_s + \bar{q} A_1^x (\frac{\bar{c}}{2V})] \dot{\bar{x}} + [K_s + \bar{q} A_0^x] \bar{x} \\ & + \sum_{i=1}^L Y_i + [M_{xu} + \bar{q} A_2^u (\frac{\bar{c}}{2V})^2] \ddot{u} + \bar{q} A_1^u (\frac{\bar{c}}{2V}) \dot{u} \\ & + \bar{q} A_2^\delta (\frac{\bar{c}}{2V})^2 \ddot{\delta} + \bar{q} A_1^\delta (\frac{\bar{c}}{2V}) \dot{\delta} + \bar{q} A_0^\delta \delta = 0 \end{aligned}$$

and $i=1,2,\dots,L$ aerodynamic lag states

$$\dot{y}_i = -I(\frac{2V}{\bar{c}})K_i y_i + D_i^x \dot{\bar{x}} + D_i^u \dot{u} + D_i^\delta \dot{\delta}$$

or

$$\begin{aligned} & M\ddot{\bar{x}} + C\dot{\bar{x}} + K\bar{x} + \sum_{i=1}^L Y_i + P\ddot{u} + Q\dot{u} + R\hat{u} + S\ddot{\delta} \\ & + T\dot{\delta} + U\delta = 0 \end{aligned}$$

and

$$\dot{y}_i = -I(\frac{2V}{\bar{c}})K_i y_i + D_i \dot{\bar{x}} + E_i \dot{u} + F_i \dot{\delta}$$

where

$$M = M_{xx} + \bar{q} A_2^x (\frac{\bar{c}}{2V})^2$$

$$C = C_s + \bar{q} A_1^x (\frac{\bar{c}}{2V})$$

$$K = K_s + \bar{q} A_0^x$$

$$P = M_{xu} + \bar{q} A_2^u \left(\frac{\bar{c}}{2V}\right)^2$$

$$Q = \bar{q} A_1^u \left(\frac{\bar{c}}{2V}\right)$$

$$R = \bar{q} A_0^u$$

$$S = \bar{q} A_2^\delta \left(\frac{\bar{c}}{2V}\right)^2 \left(\frac{1}{V}\right)$$

$$T = \bar{q} A_1^\delta \left(\frac{\bar{c}}{2V}\right) \left(\frac{1}{V}\right)$$

$$U = \bar{q} A_0^\delta \left(\frac{1}{V}\right)$$

$$D_i = D_i^x$$

$$E_i = D_i^u$$

$$F_i = D_i^\delta \left(\frac{1}{V}\right)$$

\bar{x} = modal coordinates and rigid body modes

\hat{u} = control surface deflections

δ = vertical gust velocity

V = forward velocity

$$\bar{q} = \frac{1}{2} \rho V^2$$

\bar{c} = mean aerodynamic chord

Where the actuator-aileron dynamics are

$$\hat{u} = J\bar{u}$$

$$\dot{\bar{u}} = G\bar{u} + H\hat{u}$$

where

$$u = \begin{Bmatrix} u_o \\ u_i \end{Bmatrix} = \text{commanded control inputs}$$

$$\hat{u} = \begin{Bmatrix} \bar{u}_o \\ \bar{u}_i \end{Bmatrix} = \text{control surface deflections}$$

$$u = \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \\ u_8 \\ u_9 \\ u_{10} \\ u_{11} \\ u_{12} \\ u_{13} \\ u_{14} \\ u_{15} \\ u_{16} \\ u_{17} \\ u_{18} \\ u_{19} \\ u_{20} \\ u_{21} \\ u_{22} \\ u_{23} \\ u_{24} \\ u_{25} \\ u_{26} \\ u_{27} \\ u_{28} \\ u_{29} \\ u_{30} \\ u_{31} \\ u_{32} \\ u_{33} \\ u_{34} \\ u_{35} \\ u_{36} \\ u_{37} \\ u_{38} \\ u_{39} \\ u_{40} \\ u_{41} \\ u_{42} \\ u_{43} \\ u_{44} \\ u_{45} \\ u_{46} \\ u_{47} \\ u_{48} \\ u_{49} \\ u_{50} \\ u_{51} \\ u_{52} \\ u_{53} \\ u_{54} \\ u_{55} \\ u_{56} \\ u_{57} \\ u_{58} \\ u_{59} \\ u_{60} \\ u_{61} \\ u_{62} \\ u_{63} \\ u_{64} \\ u_{65} \\ u_{66} \\ u_{67} \\ u_{68} \\ u_{69} \\ u_{70} \\ u_{71} \\ u_{72} \\ u_{73} \\ u_{74} \\ u_{75} \\ u_{76} \\ u_{77} \\ u_{78} \\ u_{79} \\ u_{80} \\ u_{81} \\ u_{82} \\ u_{83} \\ u_{84} \\ u_{85} \\ u_{86} \\ u_{87} \\ u_{88} \\ u_{89} \\ u_{90} \\ u_{91} \\ u_{92} \\ u_{93} \\ u_{94} \\ u_{95} \\ u_{96} \\ u_{97} \\ u_{98} \\ u_{99} \\ u_{100} \end{Bmatrix} = \text{control state}$$

$$G = \begin{bmatrix} 0 & 1.0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 & 0 & 0 & 0 & 0 \\ -1.774 \times 10^7 & -1.438 \times 10^5 & -431.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.0 \\ 0 & 0 & 0 & -1.614 \times 10^{11} & -5.484 \times 10^8 & -1.152 \times 10^6 & -933.0 & 0 \end{bmatrix}$$

$$J = \begin{bmatrix} .514 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & .518 & 0 & 0 & 0 \end{bmatrix}$$

$$H = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1.774 \times 10^7 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1.614 \times 10^{11} \end{bmatrix}$$

And the gust model is

$$\dot{z} = -\left(\frac{V}{\ell}\right)^2 \delta - 2\left(\frac{V}{\ell}\right) z - 2.464\left(\frac{V}{\ell}\right)^2 \eta$$

$$\dot{\delta} = z + 1.732\left(\frac{V}{\ell}\right) \eta$$

where z = an intermediate gust state

ℓ = characteristic gust length

η = white driving noise

Thus, if

$$x = \begin{pmatrix} \ddot{u} \\ \dot{x} \\ y_1 \\ y_2 \\ \vdots \\ y_L \\ \bar{u} \\ \delta \\ z \end{pmatrix}$$

then neglecting $S\ddot{\delta}$ and $P\ddot{u}$

$$\Gamma = \begin{pmatrix} 0 \\ -M^{-1}T (1.732) \left(\frac{V}{\ell}\right) \\ F1 (1.732) \left(\frac{V}{\ell}\right) \\ F2 (1.732) \left(\frac{V}{\ell}\right) \\ \vdots \\ FL (1.732) \left(\frac{V}{\ell}\right) \\ 0 \\ (1.732) \left(\frac{V}{\ell}\right) \\ (-2.464) \left(\frac{V}{\ell}\right)^2 \end{pmatrix}$$

$$A = \begin{bmatrix} 0 & I & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ -M^{-1}K & -M^{-1}C & -M^{-1} & -M^{-1} & \dots & -M^{-1} & -M^{-1}(RJ+QJG) & -M^{-1}U & -M^{-1}T \\ 0 & D1 & -I\frac{2V}{c}K_1 & 0 & \dots & 0 & E1(JG) & 0 & F1 \\ 0 & D2 & 0 & -I\frac{2V}{c}K_2 & \dots & 0 & E2(JG) & 0 & F2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & DL & 0 & 0 & \dots & -I\frac{2V}{c}K_L & EL(JG) & 0 & FL \\ 0 & 0 & 0 & 0 & \dots & 0 & G & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & 0 & 1.0 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & -(\frac{V}{l})^2 & -2(\frac{V}{l}) \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \\ -M^{-1}QJH \\ E1 (JH) \\ E2 (JH) \\ \vdots \\ \vdots \\ EL (JH) \\ H \\ 0 \\ 0 \end{bmatrix}$$

The open loop eigenvalues are also output. Open loop eigenvectors can be output as well.

MODAL

This program finds the optimal psuedo-inverse solution to the feedback gains (K) that achieve desired closed loop eigenvalue-eigenvector pairs (see Section on ES theory). The closed loop eigenvalues are also output. Closed loop eigenvectors can be output as well.

COV

This program finds the state covariance matrix for a given driving noise intensity. The rms aileron deflections and rates are determined as well.

LOADS

This program calculates the rms shear, bending moment, and torque at various station along the wing span.

FEEDBKE

This program calculates the closed loop eigenvalues at off design flight conditions.

82/12/15

PROGRAM FLUTTER

```

00100 PROGRAM FLUTTER( INPUT, OUTPUT, TAPE5=INPUT, TAPE6=OUTPUT, ONELEZ,
00110+TAPE7, MASS, TAPE1=MASS, TAPE3)
00120C*****+*****+*****+*****+*****+*****+*****+*****+*****
00130C THIS PROGRAM INPUTS THE STRUCTURAL MASS, STRUCTURAL
00140C DAMPING, STRUCTURAL STIFFNESS, AERODYNAMIC INFLUENCE
00150C MATRICES, FLIGHT CONDITIONS AND OUTPUTS THE A, B, AND W
00160C MATRICES OF THE LINEAR STATE EQUATION
00170C DX/DT = AX + BU + W
00180C -FOR THE INBOARD AND OUTBOARD ATTACHED AIRBORNS AS CONTROLS.
00190C OPEN LOOP EIGENVALUES AND EIGENVECTORS CAN BE OUTPUT
00200C AS WELL.
00210C*****+*****+*****+*****+*****+*****+*****+*****+*****
00220 REAL MXX(8,8), CS(8,8), KS(8,8), CT(8,8), KT(8,8), CC(8,8),
00230+K(8,8), A2X(8,8), A1X(8,8), AOX(8,8), MINV(8,8), A2X0(8,8)
00240+, A1X0(8,8), AOX0(8,8), A1(16,16), A2(32,32), O0(8,2), DUJ1(8,7),
00250+DUM2(8,7), DUM3(8,2), DUM4(8,7), DUM5(8,2), DUM6(8,2), DUM7(8,1)
00260 REAL DUM8(8,1), AOU(8,2), A1U(8,2), EI(8,2,4), R(8,2),
00270+EI(8,7,4), FI(8,4), A3(40,40), B3(40,2), R2(32,2), II(8,8),
00280+II(8,8,4), KI(4), AID(8,1), AOD(8,1), T(8,1), U(8,1), A4(42,4?),
00290+R4(42,2), W(42,1), B1(16,2), A5(24,24), B5(24,2)
00300 WRITE(6,8)
00310 8 FORMAT(IX,'INPUT THE ORDER OF THE STRUCTURAL-RIGID BODY STATE VECTO
00320+R(N)')
00330 READ(5,*N)
00340 WRITE(6,10)
00350 10 FORMAT(IX,'INPUT VELOCITY(U), CHORD(C), DYNAMIC PRESSURE(Q), NUMBE
00360+R OF LAG STATES(L)')
00370C INPUT FLIGHT CONDITIONS
00380C
00390C
00400 RFA0(5,1),U,C,R,L
00410 DO 20 I=1,L
00420 WRITE(6,25)I
00430C INPUT AERODYNAMIC LAG FREQUENCIES
00440C
00450C
00460 READ(5,4)KT(I)
00470 20 CONTINUE
00480 25 FORMAT(IX,'INPUT KI(,I1, )')
00490 N2=2*N
00500 KEND=N*(L+2)
00510 KEND0=KEND+7
00520 KEND2=KEND0+2
00530 K5=21*N+7

```

```

00560+,DUM5,DUM6,DUM7,DUM8,A0U,A1U,FI,R,ETI6,II,A3,B3,B2,FT,DI,K1,AID,
00570+A0D,T,U,A4,R1,B4,U,A5,B5)
-00580
STOP
00590
END
00600C
00610C
00620
SURROUTIME PROG1(N,N2,KEND,KENDD,KEND2,K5,V,C,R,L,MXX,CS,K5,CT,KT,
00630+CC,KK,A2X,A1X,A0X,MINV,A2XD,A1XD,A0XD,A1,A2,QQ,DUM1,DUM2,DUM3,DUM4
00640+,DUM5,DUM6,DUM7,DUM8,A0U,A1U,ET,R,ETI6,II,A3,B3,B2,FT,DI,K1,AID,
00650+A0D,T,U,A4,R1,B4,U,A5,B5)
00660
REAL LAMDA,LL
00670
INTEGER FLAG
00680
COMMON KRET(42)
00690
REAL MXX(N,N),CS(N,N),KS(N,N),CT(N,N),KT(N,N),CC(N,N),KK(N,N),
00700+A2X(N,N),A1X(N,N),A0X(N,N),MINV(N,N),A2XD(N,N),A1XD(N,N),A0XD(N,N),
00710+,A1(N2,N2),A2(KEND,KEND),QQ(N,2),DUM1(N,7),DUM2(N,7),DUM3(N,2),
00720+DUM4(N,7),DUM5(N,2),DUM6(N,2),DUM7(N,1),DUM8(N,1),A0U(N,2),
00730
REAL: A1U(N,2),
00740+EI(N,2,L),G(7,7),H(7,2),R(N,2),JG(2,7),EIIG(N,7,L),FI(N,L),
00750+JH(2,2),A3(KENDD,KENDD),R3(KEND,2),B2(KEND,2),II(N,N),DI(N,N,L)
00760
REAL KI(L),A1D(N,1),A0D(N,1),I(N,1),U(N,1),A4(KEND2,KEND2),
00770+B1(N2,2),B4(KEND2,2),W(KEND2,1),
00780+A5(K5,K5),R5(K5,2),JJ(2,7)
00790
REAL AA(42,42),BB(42,2)
00800
CALL GETPF('SHTAPE1,4HMASS,0,0)
00810C
00820C
READ IN STRUCTURAL DATA
00830C
00840
? FORMAT(E16.8)
00850
DO 1 I=1,N
00860
* DO 1 J=1,N
00870
READ(1,101)MXX(I,J),CS(I,J),KS(T,I)
00880
1 CONTINUE
00890
DO 3 I=1,7
00900
DO 3 J=1,7
00910C
00920C
READ IN ACTUATOR DATA
00930C
00940
READ(1,101)G(I,J)
00950
3 CONTINUE
00960
DO 4 I=1,7
00970
READ(1,101)(H(I,J),J=1,2)
00980
4 CONTINUE
00990
DO 5 J=1,7
01000
READ(1,101)(JJ(I,J),I=1,2)
01010
5 CONTINUE
01020
CALL GETPF('SHTAPE2,6HONNELE7,0,0)
01030C
01040C
READ IN AERODYNAMIC INFLUENCE DATA

```

```
01070 DO 410 J=1,N
01080 DO 410 I=1,N
01090 READ(2,109)A0X(I,J),A1X(I,J),A2X(I,J),0I(I,J,K)
01100 410 CONTINUE
01110 DO 420 J=1,2
01120 DO 420 I=1,N
01130 READ(2,101)A0U(I,J),A1U(I,J),EI(I,J,K)
01140 420 CONTINUE
01150 DO 430 I=1,N
01160 READ(2,101)A0D(I,K),A1D(I,K),FI(I,K)
01170 FI(I,K)=FI(I,K)/V
01180 430 CONTINUE
01190 101 FORMAT(3E16.8)
01200 109 FORMAT(4E16.8)
01210 EPS=1.E-10
01220 TEMP=D+(C/2./V)**2
01230 TEMP2=D+C/2./V
01240 CALL SCAMAT(TEMP,A2X,A2XD,N,N)
01250 CALL MATADD(HXX,A2XD,MINV,N,N)
01260C
01270C
01280C
01290 CALL MXLNED(MINV,N,N,DET,IRANK,EPS,A2XD,0)
01300 CALL SCAMAT(TEMP2,A1X,A1XD,N,N)
01310 CALL MATADD(CS,A1XD,CT,N,N)
01320 CALL SCAMAT(0,A0X,A0XD,N,N)
01330 CALL MATADD(KS,A0XD,KT,N,N)
01340 CALL SCAMAT(-1.,MINV,MINV,N,N)
01350 CALL MATHUL(MINV,KT,KK,N,N,N)
01360 CALL MATHUL(MINV,CT,CC,N,N,N)
01370 DO 15 I=1,N
01380 DO 14 J=1,N
01390 II(I,J)=0.
01400 14 CONTINUE
01410 JJ(I,I)=1.0
01420 15 CONTINUE
01430 N21=N2+1
01440 DO 18 I=1,N
01450 DO 18 J=1,N
01460 A1(J,I)=0.
01470 A1(I,J+N)=II(I,J)
01480 A1(I+N,J)=KK(I,J)
01490 A1(I+N,J+N)=CC(I,J)
01500 18 CONTINUE
01510 IF(L.EQ.0)GO TO 100
01520 DO 20 I=1,KEND
01530 DO 20 J=1,KEND
01540 A2(I,J)=0.
01550 20 CONTINUE
```

```
01580 A2(I,J)=A1(I,J)
01590 A2(I+N,J)=A1(I+N,J)
01600 A2(I,J+N)=A1(I,J+N)
01610 A2(I+N,J+N)=A1(I+N,J+N)
01620 DO 25 K=1,L
01630 KN=K+N
01640 A2(I+N,KN+N+J)=MINV(I,J)
01650 A2(KN+N+I,J+N)=DI(I,J,K)
01660 A2(KN+N+I,KN+N+I)=-2.*V*KI(K)/C
01670 25 CONTINUE
01680 WRITE(6,27)
01690 27 FORMAT(1X,'SHOULD THE CONTROL MATRIX B ALSO BE CALCULATED? YES(1),
01700+ NO(0)')
01710 READ(5,28)MM
01720 28 FORMAT(I1)
01730 IF(MM.EQ.1)GO TO 30
01740 FLAG=2
01750 GO TO 200
01760 30 SCA=Q+C/2./V
01770 CALL SCAMAT(SCA,A1U,00,N,2)
01780 CALL SCAMAT(R,ADU,R,N,2)
01790 DO 36 I=1,KEND
01800 DO 36 J=1,KEND
01810 A3(I,J)=0.
01820 36 CONTINUE
01830 DO 37 I=1,KEND
01840 DO 37 J=1,KEND
01850 A3(I,J)=A2(I,J)
01860 37 CONTINUE
01870 DO 38 I=1,7
01880 DO 38 J=1,7
01890 A3(KEND+I,KEND+J)=G(I,J)
01900 38 CONTINUE
01910 CALL MATMUL(JJ,G,J6,2,7,7)
01920 CALL MATMUL(OO,J6,DUM1,N,2,7)
01930 CALL MATMUL(R,JJ,DUM2,N,2,7)
01940 CALL MATADD(DUM2,DUM1,DUM2,N,7)
01950 CALL MATMUL(MINV,DUM2,DUM1,N,N,7)
01960 DO 40 K=1,L
01970 DO 39 I=1,N
01980 DO 39 J=1,2
01990 DUM3(I,J)=EI(I,J,K)
02000 39 CONTINUE
02010 CALL MATMUL(DUM3,J6,DUM4,N,2,7)
02020 DO 40 I=1,N
02030 DO 40 J=1,7
02040 EIJG(I,J,K)=DUM4(I,J)
02050 40 CONTINUE
02060 DO 45 I=1,N
```

```
02090 DO 45 K=1,L
02100 KN=K+N
02110 A3(KN+N+I,J+KEND)=EIIG(I,J,K)
02120 45 CONTINUE
02130 CALL MATMUL(JJ,H,JH,2,2,2)
02140 CALL MATMUL(OO,JH,DUM3,N,2,2)
02150 CALL MATMUL(MINV,DUM3,DUM3,DUM5,N,N,2)
02160 DO 47 K=1,L
02170 DO 46 I=1,N
02180 DO 46 J=1,2
02190 DUM3(I,J)=EI(I,J,K)
02200 46 CONTINUE
02210 CALL MATMUL(DUM3,JH,DUM6,N,2,2)
02220 DO 47 I=1,N
02230 DO 47 J=1,2
02240 EIIG(I,J,K)=DUM6(I,J)
02250 47 CONTINUE
02260 DO 48 I=1,N
02270 DO 48 J=1,2
02280 B3(I,J)=0.
02290 B3(I+N,J)=DUM5(I,J)
02300 DO 48 K=1,L
02310 KN=K*N
02320 B3(I+KN+N,J)=EI.IG(I,J,K)
02330 48 CONTINUE
02340 DO 49 I=1,7
02350 DO 49 J=1,2
02360 B3(J+KEND,J)=H(I,J)
02370 49 CONTINUE
02380 WRITE(6,50)
02390 50 FORMAT(1X,'SHOULD THE NOISE VECTOR (U) ALSO BE CALCULATED? YES(1),
02400+NO(0)')
02410 READ(5,28)MH
02420 IF(MH.EQ.1)GO TO 55
02430 FLAG=3
02440 GO TO 200
02450 55 WRITE(6,56)
02460 56 FORMAT(1X,'INPUT CHARACTERISTIC LENGTH(LL)')
02470 READ(5,4)LL
02480 TEMP3=TEMP2/U
02490 CALL SCAMAT(TEMP3,AID,T,N,1)
02500 O1=O/U
02510 CALL SCAMAT(O1,AOD,U,N,1)
02520 CALL MATMUL(MINV,U,DUM7,N,N,1)
02530 CALL MATMUL(MINV,I,DUM8,N,N,1)
02540 DO 58 I=1,KEND
02550 DO 58 J=1,2
02560 B4(I,J)=B3(I,J)
02570 B4(KFNDO+1,J)=0.
```

```
02600 DO 59 I=1,KEND2
02610 DO 59 J=1,KEND2
02620 A4(I,J)=0.
02630 59 CONTINUE
02640 DO 60 I=1,KENDD
02650 DO 60 J=1,KENDD
02660 A4(I,J)=A3(I,J)
02670 60 CONTINUE
02680 A4(KENDD+1,KENDD+2)=1.
02690 A4(KENDD+2,KENDD+1)=-1.001*(V/LL)*+?
02700 A4(KFND+2,KENDD+2)=-?.001*V/LL
02710 DO 62 I=1,N
02720 A4(I+N,KENDD+1)=DUM7(I,1)
02730 A4(I+N,KENDD+2)=DUM8(I,1)
02740 DO 62 K=1,L
02750 KN=K*N
02760 A4(I+KN+N,KENDD+2)=FI(I,K)
02770 62 CONTINUE
02780 DO 64 I=1,KEND2
02790 W(I,1)=0.
02800 64 CONTINUE
02810 XX=1.732*V/LL
02820 W(KENDD+1,1)=XX
02830 W(KENDD+2,1)=-2.464*(V/LL)*#2
02840 CALL SCAMAT(XX,DUM8,DUM8,N,1)
02850 DO 66 J=1,N
02860 W(I+N,1)=DUM8(I,1)
02870 DO 66 K=1,L
02880 KN=K*N
02890 W(I+KN+N,1)=XX*FI(I,K)
02900 66 CONTINUE
02910 FLAG=4
02920 60 TO 200
02930 100 WRITE(6,27)
02940 READ(5,28)HM
02950 IF(HM.EQ.1)GO TO 102
02960 FLAG=1
02970 60 TO 200
02980 102 SCA=0#C/2./U
02990 CALL SCAMAT(SCA,A11,00,N,2)
03000 CALL SCAMAT(Q,A0U,R,N,2)
03010 DO 106 I=1,K5
03020 DO 106 J=1,K5
03030 A5(I,J)=0.
03040 106 CONTINUE
03050 CALL MATMUL(J,J,H,JH,2,7,2)
03060 CALL MATMUL(OO,JH,DUM3,N,2,2)
03070 CALL MATMUL(HTNU,DUM3,DUM3,DUM5,N,2)
03080 DO 108 I=1,N
```



```
03110 R5(I+N, J)=DIM5(I, J)
03120 108 CONTINUE
03130 DO 110 I=1,7
03140 DO 110 J=1,2
03150 R5(I+N2, J)=H(I, J)
03160 110 CONTINUE
03170 DO 112 I=1, N2
03180 DO 112 J=1, N2
03190 A5(I, J)=A1(I, J)
03200 112 CONTINUE
03210 CALL MATMUL(JJ, 6, JG, 2, 7, 7)
03220 CALL MATMUL(QQ, JG, DUM1, N, 2, 7)
03230 CALL MATMUL(R, JJ, DUM2, N, 2, 7)
03240 CALL MATADD(DUM2, DUM1, DUM2, N, 7)
03250 CALL MATMUL(MINV, DUM2, DUM1, N, N, 7)
03260 DO 114 I=1, N
03270 DO 114 J=1, 7
03280 A5(I+N, J+N2)=DUM1(J, J)
03290 114 CONTINUE
03300 DO 116 I=1, 7
03310 DO 116 J=1, 7
03320 A5(J+N2, J+N2)=6(I, J)
03330 116 CONTINUE
03340 FLAG=5
03350 200 WRITE(6, 201)
03360 201 FORMAT(1X, 'INPUT RETAIN(I) OR DELETE(0) FOR EACH STATE MEMBER.'/)
03370 KSTATE=N
03380 KSUM=0
03390 READ(5, *) (KRET(J), J=1, KSTATE)
03400 DO 205 I=1, KSTATE
03410 KRET(I+N)=KRET(I)
03420 IF (KRET(I).EQ.1) KSUM=KSUM+1
03430 205 CONTINUE
03440 IF (L.EQ.0) GO TO 207
03450 DO 206 K=1, L
03460 DO 206 T=1, N
03470 KN=N*(K+1)+I
03480 KRET(KN)=KRET(I)
03490 206 CONTINUE
03500 DO 209 I=1, 9
03510 KRET(KEND+I)=I
03520 209 CONTINUE
03530 207 GO TO (310, 320, 330, 340, 350), FLAG
03540 310 KSUM=2*KSUM
03550 CALL PROG2(FLAG, KSUM, A1, B1, N2, AA, BB)
03560 GO TO 900
03570 320 KSUM=(L+2)*KSUM
03580 CALL PROG2(FLAG, KSUM, A2, B2, KEND, AA, BB)
03590 GO TO 900
```

```
03620 GO TO 900
03630 KSUM=9+(1+2)+KSUM
03640 DO 341 J=1,KEND2
03650 IF(KRET(I).EQ.0)GO TO 341
03660 WRITE(3,2)U(I,1)
03670 341 CONTINUE
03680 CALL PROG2(FLAG,KSUM,A4,B4,KEND2,AA,BB)
03690 GO TO 900
03700 KSUM=7+KSUM
03710 DO 351 I=N21,K5
03720 KRET(I)=1
03730 351 CONTINUE
03740 CALL PROG2(FLAG,KSUM,A5,B5,K5,AA,BB)
03750 900 RETURN
03760 END
03770C
03780C
03790 SUBROUTINE MATDEL(A,N,AA,KSUM)
03800C
03810C THIS ROUTINE DELETES DESIRED STATES FROM THE A MATRIX
03820C
03830 DIMENSION A(N,N),AA(KSUM,KSUM),JRET(40)
03840 COMMON KRET(42)
03850 J=0
03860 DO 100 I=1,N
03870 IF(KRET(I).EQ.0)GO TO 100
03880 J=J+1
03890 JRET(J)=I
03900 100 CONTINUE
03910 DO 200 I=1,KSUM
03920 DO 200 J=1,KSUM
03930 AA(I,J)=A(JRET(I),JRET(J))
03940 200 CONTINUE
03950 RETURN
03960 END
03970C
03980C
03990 SUBROUTINE MATDEL(B,N,BB,KSUM)
04000C
04010C THIS ROUTINE DELETES DESIRED STATES FROM THE B MATRIX
04020C
04030 DIMENSION B(N,2),BB(KSUM,2),JRET(40)
04040 COMMON KRET(42)
04050 J=0
04060 DO 100 I=1,N
04070 IF(KRET(I).EQ.0)GO TO 100
04080 J=J+1
04090 JRET(J)=I
04100 100 CONTINUE
```

“PAGE MISSING FROM AVAILABLE VERSION”

PAGE 73

```
04130 RR(I,J)=R(I*RET(I),J)
04140 200 CONTINUE
04150 RETURN
04160 END
04170C
04180C
04190C.....T(M,P)=A(M,N) + U(M,P)
04200 SUBROUTINE MATMUL(A,U,T,M,N,P)
04210 INTEGER P
04220 DIMENSION A(M,N),U(N,P),T(M,P)
04230 DO 1 I=1,M
04240 DO 1 J=1,P
04250 T(I,J)=0.
04260 1 CONTINUE
04270 DO 2 I=1,M
04280 DO 2 J=1,P
04290 DO 2 K=1,N
04300 T(I,J)=A(I,K)*U(K,J)+T(I,J)
04310 2 CONTINUE
04320 RETURN
04330 END
04340C
04350C
04360C.....C(M,N)=A(M,N) + B(M,N)
04370 SUBROUTINE MATADD(A,B,C,M,N)
04380 DIMENSION A(M,N),B(M,N),C(M,N)
04390 DO 3 I=1,M
04400 DO 3 J=1,N
04410 C(I,J)=A(I,J)+B(I,J)
04420 3 CONTINUE
04430 RETURN
04440 END
04450C
04460C
04470C.....B(M,N)=5 * A(M,N)
04480 SUBROUTINE SCAMAT(S,A,B,M,N)
04490 DIMENSION A(M,N),B(M,N)
04500 DO 8 I=1,M
04510 DO 8 J=1,N
04520 B(I,J)=S*A(I,J)
04530 8 CONTINUE
04540 RETURN
04550 END
04560C
04570C
04580C.....V(N,M)= A(M,N) TRANPOSED
04590 SUBROUTINE MATTRA(A,V,N,M)
04600 DIMENSION A(M,N),V(N,M)
04610 DO 14 I=1,M
```

```
V1,J,I=H1,I,J
04640 14 CONTINUE
04650 RETURN
04660 ENH
04670C
04680C
04690 SUBROUTINE PRO62(FLAG,KSUM,A,B,IP,AA,BB)
04700 INTEGER FLAG,IV1(42)
04710 REAL AA(KSUM,KSUM),BB(KSUM,2),A(IP,(P),R(IP,2),FV1(42)
04720 REAL WR(42),WI(42),Z(42,42)
04730 GO TO(310,320,330,340,350),FLAG
04740 100 FORMAT(3E16.8)
04750 310 CALL MATDEL(A,IP,AA,KSUM)
04760 DO 311 I=1,KSUM
04770 DO 311 J=1,KSUM
04780 WRITE(3,100)AA(I,J)
04790 311 CONTINUE
04800 320 CALL PRO63(KSUM,AA,UR,WI,Z,IV1,FV1)
04810 320 CALL MATDEL(A,IP,AA,KSUM)
04820 DO 321 J=1,KSUM
04830 WRITE(3,100)AA(I,J)
04840 321 CONTINUE
04850 330 CALL PRO63(KSUM,AA,UR,WI,Z,IV1,FV1)
04860 GO TO 900
04870 330 CALL MATDEL(A,IP,AA,KSUM)
04880 CALL MATDEL(B,IP,BB,KSUM)
04890 DO 331 I=1,KSUM
04900 DO 331 J=1,KSUM
04910 WRITE(3,100)AA(I,J)
04920 331 CONTINUE
04930 DO 332 I=1,KSUM
04940 WRITE(3,100)BB(I,1),BB(I,2)
04950 332 CONTINUE
04960 CALL PRO63(KSUM,AA,UR,WI,Z,IV1,FV1)
04970 GO TO 900
04980 340 CALL MATDEL(A,IP,AA,KSUM)
04990 CALL MATDEL(B,IP,BB,KSUM)
05000 DO 341 I=1,KSUM
05010 DO 341 J=1,KSUM
05020 WRITE(3,100)AA(I,J)
05030 341 CONTINUE
05040 DO 342 I=1,KSUM
05050 WRITE(3,100)BB(I,1),BB(I,2)
05060 342 CONTINUE
05070 CALL PRO63(KSUM,AA,UR,WI,Z,IV1,FV1)
05080 GO TO 900
05090 350 CALL MATDEL(A,IP,AA,KSUM)
05100 CALL MATDEL(B,IP,BB,KSUM)
05110 DO 351 I=1,KSUM
05120 DO 351 J=1,KSUM
```

```

05170 DO 352 I=1,KSUM
05180 WRITE(3,100)BB(I,1),BB(I,2)
05190 352 CONTINUE
05200 CALL PROG3(KSUM,AA,WR,WI,Z,IVI,FV1)
05210 900 CALL REPLAC(5HTAPE3,5HTLE9,0,0)
05220 RETURN
05230 END
05240C
05250C
05260 SUBROUTINE PROG3(N,AA,WR,WI,Z,IVI,FV1)
05270 COMMON KRET(42)
05280 DIMENSION AA(N,N),WR(N),WI(N),Z(N,N),IVI(N),FV1(N)
05290C
05300C CALCULATE EIGENVALUES AND EIGENVECTORS OF AA
05310C
05320 CALL RG(N,N,AA,WR,WI,1,Z,IVI,FV1,IERR)
05330 WRITE(6,23)
05340 23 FORMAT(41X,'OPEN LOOP EIGENVALUES'/31X,'REAL PART',20X,
05350+'IMAGINARY PART')
05360 DO 30 I=1,N
05370 WRITE(6,25)I,WR(I),WI(I)
05380 25 FORMAT(20X,I2,' ',2X,E15.4,15X,E15.4)
05390 30 CONTINUE
05400 WRITE(6,55)
05410 55 FORMAT(1X,'DO YOU WANT THE OPEN LOOP EIGENVECTORS OUTPUT? YES(1)
05420+',N0(0)')
05430 READ(5,*)NM
05440 IF(NM.EQ.0)GO TO 60
05450 WRITE(6,64)
05460 64 FORMAT(1X,'INPUT PRINT(1) OR SUPPRESS(0) FOR EACH EIGENVECTOR')
05470 READ(5,*)(KRET(J),J=1,N)
05480 WRITE(6,34)
05490 34 FORMAT(41X,'OPEN LOOP EIGENVECTORS'/31X,'REAL PART',20X,
05500+'IMAGINARY PART')
05510 IFLAG=0
05520 DO 50 J=1,N
05530 IF(KRET(J).EQ.0)GO TO 49
05540 IF(WI(J).NE.0.0)GO TO 35
05550 DO 37 I=1,N
05560 IF(I.NE.1)GO TO 36
05570 WRITE(6,38)J,Z(1,J)
05580 38 FORMAT(/20X,I2,' ',2X,E15.4,20X,'0.0')
05590 60 TO 37
05600 16 WRITE(6,39)Z(I,J)
05610 39 FORMAT(25X,E15.4,20X,'0.0')
05620 37 CONTINUE
05630 60 TO 50
05640 35 IF(IFLAG.EQ.1)GO TO 40
05650 IFLAG=1

```

```
05680 WRITE(6,43)J,Z(I,J),Z(I,J+1)
05690 43 FORMAT(/20X,I2,/,2X,E15.4,15X,E15.4)
05700 60 TO 41
05710 42 WRITE(6,44)Z(I,J),Z(I,J+1)
05720 44 FORMAT(25X,E15.4,15X,E15.4)
05730 41 CONTINUE
05740 60 TO 50
05750 40 J1=J-1
05760 WRITE(6,45)J,J1
05770 45 FORMAT(/20X,I2,/,5X,CONJUGATE OF EIGENVECTOR,1X,I2,/,)
05780 49 IFLAG=0
05790 50 CONTINUE
05800 60 RETURN
05810 END
```

82/17/15
MNFETS PROGRAM MODAL

```

00100 PROGRAM MODAL(INPUT,OUTPUT,TAPE5=INPUT,SAFE6=OUTPUT,FILE3,
00110+TAPE1=FILE3,TAPE3)
00120C*****
00130C THIS PROGRAM INPUTS THE STATE MATRICES(OUTPUT
00140C OF FLUTTER) A, B, W, DESIRFD CLOSED
00150C LOOP EIGENVALUES AND EIGENVECTORS, THE EIGENVECTOR
00160C WEIGHTING MATRIX AND OUTPUTS THE GAIN MATRIX K WHERE
00170C U = KX
00180C THE CLOSED LOOP EIGENVALUES AND EIGENVECTORS THAT WERE
00190C ACHIEVED CAN RE OUTPUT AS UFILE.
00200C*****
00210 DIMENSION A(42,42),WR(2,42),WI(2,42),Z(42,42),IV1(42),
00220+FU1(42),VECRD(42,42),VECID(42,42),ERD(42),EID(42),KRET(2,42),U(42,1),
00230+R(42,2)
00240 WRITE(6,10)
00250 10 FORMAT(1X,'INPUT ORDER OF SYSTEM')
00260 READ(5,*)N
00270 CALL PROG1(N,A,WR,WI,Z,IV1,VECRD,VECID,ERD,EID,KRET,
00280+W,R)
00290 STOP
00300 END
00310C
00320C
00330 SUBROUTINE PROG1(N,A,WR,WI,Z,IV1,VECRD,VECID,ERD,EID,
00340+KRET,W,B)
00350 DIMENSION A(N,N),WR(2,N),WI(2,N),Z(N,N),IV1(N),FU1(N),
00360+VECRD(N,N),VECID(N,N),ERD(N),EID(N),KRET(2,N),U(N,1),B(N,2)
00370 REAL IJ(42,42),LB(84,4),LBT(4,84)
00380 DIMENSION OB(84,84),VB(84,1),YB(84,84),PP(42,2),TV(42,1),
00390+BR(84,4),C(42,42),P(84,1)
00400 N2=N*2
00410 NM1=N-1
00420 NM2=N-2
00430 WRITE(6,11)
00440 11 FORMAT(1X,'IS THE NOISE VECTOR (W) BEING INPUT? YES(1), NO(0)')
00450 READ(5,12)NM
00460 12 FORMAT(I1)
00470C
00480C READ IN STATE MATRICES OUTPUT FROM PROGRAM: FLUTTER
00490C
00500 CALL GETPF(5HTAPE1,5HFILE3,0,0)
00510 IF(MM.EQ.0)GO TO 14
00520 DO 13 I=1,N
00530 READ(1,15)U(I,1)

```



```

00560 14 DO 16 I=1,N
00570 DO 16 J=1,N
00580 READ(1,15)A(I,J)
00590 16 CONTINUE
00600 DO 17 I=1,N
00610 READ(1,15)B(I,1),B(I,2)
00620 17 CONTINUE
00630 DO 24 I=1,N
00640 DO 24 J=1,N
00650 VECRD(I,J)=A(I,J)
00660 24 CONTINUE
00670C
00680C
00690C
00700 CALL RG(N,N,VECRD,ERD,EID,1,Z,IV1,FU1,IERR)
00710 WRITE(6,23)
00720 23 FORMAT(41X,'OPEN LOOP EIGENVALUES'/31X,'REAL PART',20X,
00730+'IMAGINARY PART')
00740 DO 30 I=1,N
00750 WRITE(6,25)I,ERD(I),EID(I)
00760 25 FORMAT(20X,I2,' ',2X,E15.4,15X,E15.4)
00770 30 CONTINUE
00780 IFLAG=0
00790 DO 50 J=1,N
00800 IF(EID(J).NE.0.0)GO TO 35
00810 KRET(1,J)=0
00820 DO 37 I=1,N
00830 IF(T.NE.1)GO TO 36
00840 VECRD(I,J)=7(1,J)
00850 VECTD(I,J)=0.0
00860 GO TO 37
00870 36 VECRD(I,J)=Z(I,J)
00880 VECID(I,J)=0.0
00890 37 CONTINUE
00900 GO TO 50
00910 35 KRET(1,J)=IFLAG
00920 7F(IFLAG.EQ.1)GO TO 40
00930 IFLAG=1
00940 DO 41 I=1,N
00950 IF(T.NE.1)GO TO 42
00960 VECRD(I,J)=Z(1,J)
00970 VECID(I,J)=Z(1,J+1)
00980 GO TO 41
00990 42 VECRD(I,J)=Z(I,J)
01000 VECID(I,J)=Z(I,J+1)
01010 41 CONTINUE
01020 GO TO 50
01030 40 J1=J-1
01040 DO 46 T=1,N

```

```

01070 44 CONTINUE
01080 TFLAB=0
01090 50 CONTINUE
01100 WRITE(6,100)
01110 100 FORMAT(1X,'INPUT RETAIN(1) OR CHANGE(1) FOR EACH EIGENSOLUTION )
01120 READ(5,*)(KRET(2,J),J=1,N)
01130 DO 110 J=1,N
01140 IF(KRET(2,J).EQ.1)GO TO 120
01150 IF(KRET(1,J).EQ.0)GO TO 105
01160 I1=I-1
01170 WRITE(6,107)I,I1
01180 107 FORMAT(1X,'DESIRED EIGENVALUE ',I2,', WILL BE ENTERED AS THE CONJ
01190+UGATE OF DESIRED EIGENVALUE ',I2,',')
01200 ERD(I)=ERD(I1)
01210 EID(I)=-EID(I1)
01220 GO TO 110
01230 105 IF(VECRD(NM1,I).NE.0..OR.VECRD(N,I).NE.0.)GO TO 110
01240 WRITE(6,101)I
01250 101 FORMAT(1X,'INPUT DESIRED REAL PART, IMAGINARY PART OF NEW',I,
01260+'EIGENVALUE ',I2,',')
01270 READ(5,*)FRD(I),EID(I)
01280 GO TO 110
01290 120 DO 121 J=1,2
01300 WR(J,I)=0.
01310 WI(J,I)=0.
01320 121 CONTINUE
01330 110 CONTINUE
01340 DO 140 I=1,N
01350 IF(KRET(2,I).EQ.1)GO TO 140
01360 IF(KRET(1,I).EQ.0)GO TO 113
01370 I1=I-1
01380 WRITE(6,114)I,I1
01390 114 FORMAT(1X,'DESIRED EIGENVECTOR ',I2,', WILL BE ENTERED AS THE CONJ
01400+UGATE OF DESIRED EIGENVECTOR ',I2,',')
01410 DO 115 J=1,N
01420 VECR(J,I)=VECR(J,I1)
01430 VECI(J,I)=-VECI(J,I1)
01440 115 CONTINUE
01450 GO TO 140
01460 113 WRITE(6,111)I
01470 111 FORMAT(1X,'INPUT DESIRED REAL PART, IMAGINARY PART OF NEW EIGENVECT
01480+'R ',I2,',')
01490 JN=N
01500 IF(VECRD(NM1,I).NE.0..OR.VECRD(N,I).NE.0.) JN=NM2
01510 DO 112 J=1,JN
01520 READ(5,*)VECRD(J,I),VECI(J,I)
01530 112 CONTINUE
01540 140 CONTINUE
01550 DO 150 I=1,N

```

```
01580 T1=I-1
01590 DO 151 J=1,N
01600 VECRD(J,J)=VECRD(J,I1)
01610 VECID(J,I)=-VECID(J,I1)
01620 151 CONTINUE
01630 DO 152 J=1,2
01640 WR(J,I)=WR(J,I1)
01650 WI(J,I)=-WI(J,I1)
01660 152 CONTINUE
01670 60 TO 150
01680 153 IF(VECRD(NM1,I).NE.0..OR.VECRD(N,I).NE.0.)GO TO 154
01690 CALL PRO62(I,ERD,ETD,WR,WI,VECRD,VECID,A,B,N,N2,OB,LB,YB,
01700+BR,JI,IRI,VR,P,C)
01710 60 TO 150
01720 154 CALL PRO64(I,ERD,WR,VECRD,A,B,N,NM2,NM1,OB,LB,YB,BR,II,
01730+LB,VB,P,C,PP,DU)
01740 150 CONTINUE
01750 DO 160 I=1,N
01760 J1=J-1
01770 IF(KRET(I,I).EQ.0)GO TO 160
01780 DO 161 J=1,N
01790 VECRD(J,I)=VECID(J,I1)
01800 161 CONTINUE
01810 DO 162 J=1,2
01820 WR(J,I)=WI(J,I1)
01830 162 CONTINUE
01840 160 CONTINUE
01850 EPS=1.E-10
01860C
01870C FIND INVERSE(VECRD)
01880C
01890 CALL MXLNER(VECRD,N,N,DET,IRANK,FRS,VECTD)
01900 CALL MATHUL(WR,VECRD,UI,2,N,N)
01910 DO 165 I=1,N
01920 WRITE(3,164)(UI(J,I),J=1,2)
01930 164 FORMAT(3F16.8)
01940 165 CONTINUE
01950 CALL MATHUL(B,WI,VECTD,N,N)
01960 CALL MATADD(A,VECID,VECRD,N,N)
01970 CALL PRO63(N,VECRD,ERD,ETD,Z,I,VI,FUI,KRET)
01980 CALL REPLACE(5HTAPE3,5HGAINS,0,0)
01990 RETURN
02000 END
02010C
02020C
02030C.....T(M,P)=A(M,N) * U(N,P)
02040 SUBROUTINE MATHUL(A,U,T,M,N,P)
02050 INTEGER P
02060 DIMENSION A(M,N),U(N,P),T(M,P)
```

```
02090 T(I,J)=0.
02100 1 CONTINUE
02110 DO 2 I=1,M
02120 DO 2 J=1,P
02130 DO 2 K=1,N
02140 T(I,J)=A(I,K)+U(K,J)+T(I,J)
02150 2 CONTINUE
02160 RETURN
02170 END
02180C
02190C
02200C.....C(M,N)=A(M,N) + B(M,N)
02210 SUBROUTINE MATADD(A,B,C,M,N)
02220 DIMENSION A(M,N),B(M,N),C(M,N)
02230 DO 3 I=1,M
02240 DO 3 J=1,N
02250 C(I,J)=A(I,J)+B(I,J)
02260 3 CONTINUE
02270 RETURN
02280 END
02290C
02300C
02310C.....B(M,N)=S * A(M,N)
02320 SUBROUTINE SCAMAT(S,A,B,M,N)
02330 DIMENSION A(M,N),B(M,N)
02340 DO 8 I=1,M
02350 DO 8 J=1,N
02360 B(I,J)=S*A(I,J)
02370 8 CONTINUE
02380 RETURN
02390 END
02400C
02410C
02420C.....V(N,M)= A(M,N) TRANSPOSED
02430 SUBROUTINE MATTRA(A,V,M,N)
02440 DIMENSION A(M,N),V(N,M)
02450 DO 14 I=1,M
02460 DO 14 J=1,N
02470 V(J,I)=A(I,J)
02480 14 CONTINUE
02490 RETURN
02500 END
02510C
02520C
02530C
02540 SUBROUTINE PROB3(N,AA,UR,U,I,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10,I11,I12,I13,I14,I15,I16,I17,I18,I19,I20,I21,I22,I23,I24,I25,I26,I27,I28,I29,I30,I31,I32,I33,I34,I35,I36,I37,I38,I39,I40,I41,I42,I43,I44,I45,I46,I47,I48,I49,I50,I51,I52,I53,I54,I55,I56,I57,I58,I59,I60,I61,I62,I63,I64,I65,I66,I67,I68,I69,I70,I71,I72,I73,I74,I75,I76,I77,I78,I79,I80,I81,I82,I83,I84,I85,I86,I87,I88,I89,I90,I91,I92,I93,I94,I95,I96,I97,I98,I99,I100)
02550 DIMENSION AA(N),UR(N),U(N),I(N),I1(N),I2(N),I3(N),I4(N),I5(N),I6(N),I7(N),I8(N),I9(N),I10(N),I11(N),I12(N),I13(N),I14(N),I15(N),I16(N),I17(N),I18(N),I19(N),I20(N),I21(N),I22(N),I23(N),I24(N),I25(N),I26(N),I27(N),I28(N),I29(N),I30(N),I31(N),I32(N),I33(N),I34(N),I35(N),I36(N),I37(N),I38(N),I39(N),I40(N),I41(N),I42(N),I43(N),I44(N),I45(N),I46(N),I47(N),I48(N),I49(N),I50(N),I51(N),I52(N),I53(N),I54(N),I55(N),I56(N),I57(N),I58(N),I59(N),I60(N),I61(N),I62(N),I63(N),I64(N),I65(N),I66(N),I67(N),I68(N),I69(N),I70(N),I71(N),I72(N),I73(N),I74(N),I75(N),I76(N),I77(N),I78(N),I79(N),I80(N),I81(N),I82(N),I83(N),I84(N),I85(N),I86(N),I87(N),I88(N),I89(N),I90(N),I91(N),I92(N),I93(N),I94(N),I95(N),I96(N),I97(N),I98(N),I99(N),I100(N))
02560C
02570C CALCULATE CLOSED LOOP EIGENVALUES AND EIGENVECTORS
```

```
02600 WRITE(6,23)
02610 23 FORMAT(41X,'CLOSED LOOP EIGENVALUES'/31X,'REAL PART',20X,
02620+'IMAGINARY PART')
02630 DO 30 I=1,N
02640 WRITE(6,25)I,WR(I),WJ(I)
02650 25 FORMAT(20X,I2,' ',2X,E15.4,15X,E15.4)
02660 30 CONTINUE
02670 WRITE(6,55)
02680 55 FORMAT(1X,'DO YOU WANT THE CLOSED LOOP EIGENVECTORS OUTPUT? YES(1)
02690+',ND(0)')
02700 READ(5,*)MM
02710 IF(MM.EQ.0)GO TO 60
02720 WRITE(6,64)
02730 64 FORMAT(1X,'INPUT PRINT(1) OR SUPPRESS(0) FOR EACH EIGENVECTOR')
02740 READ(5,*)(KRET(I),J),J=1,N)
02750 WRITE(6,34)
02760 34 FORMAT(41X,'CLOSED LOOP EIGENVECTORS'/31X,'REAL PART',20X,
02770+'IMAGINARY PART')
02780 IFLAG=0
02790 DO 50 J=1,N
02800 IF(KRET(I),J).EQ.0)GO TO 49
02810 IF(WI(J).NE.0.0)GO TO 35
02820 DO 37 I=1,N
02830 IF(I.NE.1)GO TO 36
02840 WRITE(6,38)J,Z(I),J)
02850 38 FORMAT(/20X,I2,' ',2X,E15.4,20X,'0.0')
02860 GO TO 37
02870 36 WRITE(6,39)Z(I),J)
02880 39 FORMAT(25X,E15.4,20X,'0.0')
02890 37 CONTINUE
02900 GO TO 50
02910 35 IF(IFLAG.EQ.1)GO TO 40
02920 IFLAG=1
02930 DO 41 I=1,N
02940 IF(I.NE.1)GO TO 42
02950 WRITE(6,43)I,Z(I),J),Z(I),J+1)
02960 43 FORMAT(/20X,I2,' ',2X,E15.4,15X,E15.4)
02970 GO TO 41
02980 42 WRITE(6,44)Z(I),J),Z(I),J+1)
02990 44 FORMAT(25X,E15.4,15X,E15.4)
03000 41 CONTINUE
03010 GO TO 50
03020 40 J1=J-1
03030 WRITE(6,45)I,J1)
03040 45 FORMAT(/20X,I2,' ',5X,'CONJUGATE OF EIGENVECTOR',IX,I2,' ')
03050 49 IFLAG=0
03060 50 CONTINUE
03070 60 RETURN
03080 END
```

```

SUBROUTINE PRO62(ID,ERD,ETD,UR,UI,VECRD,VECTD,A,B,N,N2,OR,LB,YR,
03110
03120+RR,II,LBT,VB,P,C)
03130 DIMENSION ERD(N),ETD(N),UR(? ,N),UI(? ,N),VECRD(N,N),VECTD(N,N),
03140+A(N,N),R(N,?),OR(N2,N2),WB(4,1),VB(N2,1),YB(N2,N2),RB(N2,4),
03150+C(N,N),R(4,4),P(N2,1),T(4,1),RR(4,4)
03160 REAL II(N,N),LB(N2,4),LBT(4,N2),LAMR,LAMI
03170 LAMR=ERD(ID)
03180 LAMI=ETD(ID)
03190 DO 5 I=1,N
03200 VB(I,1)=VECRD(I,TD)
03210 VR(J+N,1)=VECTD(I,TD)
03220 5 CONTINUE
03230 DO 10 I=1,N
03240 DO 10 J=1,N
03250 II(I,J)=0.
03260 IF(J.EQ.J)II(I,J)=1.
03270 10 CONTINUE
03280 CALL SCARAT(LAMR,II,C,N,N)
03290 DO 20 I=1,N
03300 DO 20 J=1,N
03310 YB(I,J)=C(I,J)-A(I,J)
03320 YB(I+N,J+N)=YB(I,J)
03330 20 CONTINUE
03340 CALL SCARAT(LAMI,II,C,N,N)
03350 DO 30 I=1,N
03360 DO 30 J=1,N
03370 YB(I,J+N)=-C(I,J)
03380 YB(I+N,J)=C(I,J)
03390 30 CONTINUE
03400 EPS=1.E-10
03410C
03420C FIND INVERSE(YB)
03430C
03440 CALL MXLNED(YB,N2,N2,DET,IRANK,EPS,OR)
03450 DO 40 I=1,N
03460 DO 40 J=1,2
03470 RB(I,J)=R(I,J)
03480 RB(I+N,J)=0.
03490 RB(I,J+2)=0.
03500 RB(I+N,J+2)=B(I,J)
03510 40 CONTINUE
03520 CALL MATMUL(YB,RB,LB,N2,N2,4)
03530 DO 42 I=1,N2
03540 DO 42 J=1,N2
03550 OR(I,J)=0.
03560 42 CONTINUE
03570 WRITE(6,41)ID
03580 41 FORMAT('X, INPUT REAL PART OF THE DIAGONAL OF THE OR MATRIX FOR
03590+FGVECTOR ',I2,' ')

```

READ(5,+)OB(J,J),J=1,N

```

03650C
03630 READ(5,+)OB(J,J),J=1,N
03640 WRITE(6,43)ID
03650 43 FORMAT('X,'INPUT IMAGINARY PART OF THE DIAGONAL OF THE QB MATRIX
03660+FOR EIGENVECTOR ',I2,'.')
03670 READ(5,*)(OB(J+N,J+N),J=1,N)
03680 CALL MATHUL(OB,LB,BB,N2,N2,4)
03690 CALL MATRA(LB,LBT,N2,4)
03700 CALL MATHUL(LBT,BB,R,4,N2,4)
03710C
03720C FIND INVERSE(R)
03730C
03740 CALL MXLNEO(R,4,4,DET,JRANK,EPS,RR)
03750 CALL MATHUL(OB,VR,P,N2,N2,1)
03760 CALL MATHUL(LBT,P,T,4,N2,1)
03770 CALL MATHUL(R,T,UR,4,4,1)
03780 CALL MATHUL(LB,UB,VR,N2,4,1)
03790 DO 60 J=1,N
03800 VECRD(I,ID)=VB(I,1)
03810 VECID(I,ID)=VB(I+N,1)
03820 60 CONTINUE
03830 DO 70 I=1,2
03840 UR(I,ID)=UR(I,1)
03850 UI(I,ID)=UR(I+2,1)
03860 70 CONTINUE
03870 RETURN
03880 END
03890C
03900C
03910 SUBROUTINE PROG4(ID,ERD,UR,VECRD,A,B,N,NM2,NM1,OB,LB,YB,
03920+BB,II,LBT,VB,PP,C,P,DV)
03930C
03940C THIS ROUTINE DETERMINES THE CLOSED LOOP ATTAINABLE
03950C EIGENVECTORS FOR THE UNCONTROLLABLE GUST MODES
03960C
03970 DIMENSION ERD(N),UR(2,N),VECRD(N,N),A(N,N),B(N,2),OB(NM2,
03980+NM2),UB(2,1),VB(NM2,1),YB(NM2,NM2),BB(NM2,2),C(N,N),R(2,2),
03990+PP(NM2,1),T(2,1),RR(2,2),P(NM2,2),VD(2,1),DU(NM2,1)
04000 REAL TI(N,N),LR(NM2,2),LBT(2,NM2),LAMB
04010 LAMB=ERD(ID)
04020 DO 5 I=1,NM2
04030 VB(I,1)=VECRD(I,ID)
04040 5 CONTINUE
04050 VD(1,1)=VECRD(NM1,ID)
04060 VD(2,1)=VECRD(N,ID)
04070 DO 10 I=1,N
04080 DO 10 J=1,N
04090 TI(I,J)=0.
04100 TF(T,ER,J)TI(I,1)=1.

```

CALL MATRADD(VB, DV, VB, NM2, 1)

```

04130 DO 20 I=1, NM2
04140 P(I,1)=C(I, NM1)-A(I, NM1)
04150 P(I,2)=C(I, N)-A(I, N)
04160 DO 20 J=1, NM2
04170 YR(I, J)=C(I, J)-A(I, J)
04180 20 CONTINUE
04190 EPS=1.E-10

```

```

04200C
04210C FTND INVERSE(YB)
04220C

```

```

04230 CALL MXLNFG(YB, NM2, NM2, DET, JRANK, EPS, DR)
04240 DO 40 I=1, NM2
04250 DO 40 J=1, 2
04260 BB(I, J)=R(I, J)
04270 40 CONTINUE
04280 CALL MATHUL(YB, P, LB, NM2, NM2, 2)
04290 CALL MATHUL(LB, VD, DV, NM2, 2, 1)
04300 CALL MATHUL(YB, RR, LB, NM2, NM2, 2)
04310 CALL MATADD(VR, DV, VB, NM2, 1)
04320 DO 42 I=1, NM2
04330 DO 42 J=1, NM2
04340 DB(I, J)=0.
04350 42 CONTINUE

```

```

04360 WRITE(6, A1) ID
04370 41 FORMAT('X, 'INPUJ REAL PART OF THE DIAGONAL OF THE DR MATRIX FOR
04380+EIGENVECTOR ', I2, ', ')

```

```

04390 READ(5, *) (DB(J, J), J=1, NM2)
04400 CALL MATHUL(DB, LB, BB, NM2, NM2, 2)
04410 CALL MATRA(LB, LRT, NM2, 2)
04420 CALL MATHUL(LRT, BB, R, 2, NM2, 2)

```

```

04430C
04440C FTND INVERSE(R)
04450C

```

```

04460 CALL MXLNFG(R, 2, 2, DET, IRANK, EPS, RR)
04470 CALL MATHUL(DR, VB, PP, NM2, NM2, 1)
04480 CALL MATHUL(LRT, PP, T, 2, NM2, 1)
04490 CALL MATHUL(R, T, UB, 2, 2, 1)
04500 CALL MATHUL(LB, UB, VB, NM2, 2, 1)
04510 CALL SCMAT(-1., DV, DV, NM2, 1)
04520 CALL MATADD(VB, DV, VB, NM2, 1)
04530 DO 60 I=1, NM2

```

```

04540 VECR0(I, ID)=VB(I, 1)
04550 60 CONTINUE

```

```

04560 DO 70 I=1, 2
04570 WR(I, ID)=WR(I, 1)
04580 70 CONTINUE
04590 RETURN
04600 END

```


82/12/15
NFPTS PROGRAM COV

```

00100 PROGRAM COV(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,FTLE3,
00110+TAPE1=FTLE3,GAINS,TAPE3=GAINS,TAPE2)
00120C+*****
00130C THIS PROGRAM INPUTS THE STATE MATRICES
00140C (OUTPUT OF FLUTTER) A, B, U, FEEDBACK
00150C GAINS(OUTPUT OF MODAL), INTENSITY OF THE
00160C GUST NOISE AND OUTPUTS THE CLOSED LOOP
00170C STATE COVARIANCE MATRIX, RMS CONTROL SURFACE
00180C RATES AND DEFLECTIONS IN DEGREES.(NOTE: INPUT
00190C ZERO FEEDBACK GAINS FOR OPEN LOOP RESULTS)
00200C+*****
00210 DIMENSION A(42,42),WI(2,42),Z(42,42),
00220+V1(42),VECD(42,42),VECD(42,42),W(42,1),
00230+B(42,2)
00240 WRITE(6,10)
00250 10 FORMAT(IX,'INPUT ORDER OF SYSTEM')
00260 READ(5,*)N
00270 CALL PROG1(N,A,WI,Z,FV1,VECD,VECD,
00280+U,B)
00290 STOP
00300 END
00310C
00320C
00330 SUBROUTINE PROG1(N,A,WI,Z,FV1,VECD,VECD,
00340+U,B)
00350 DIMENSION A(N,N),WI(2,N),Z(N,N),FV1(N),
00360+VECD(N,N),VECD(N,N),W(N,1),B(N,2)
00370 DIMENSION PP(42,42),DD(42,42),W(1,42)
00380 REAL JJ(42,42)
00390 CALL GETPF(5HTAPE1,5HFILE3,0,0)
00400C
00410C READ IN STATE MATRICES OUTPUT FROM PROGRAM: FLUTTER
00420C
00430 DO 13 I=1,N
00440 READ(1,15)W(I,1)
00450 15 FORMAT(3E16.8)
00460 13 CONTINUE
00470 DO 16 I=1,N
00480 DO 16 J=1,N
00490 READ(1,15)A(I,J)
00500 16 CONTINUE
00510 DO 17 I=1,N
00520 READ(1,15)R(I,1),B(I,2)
00530 17 CONTINUE

```

READ IN FEEDBACK GAINS OUTPUT FROM PROGRAM: MODAL

00560C
00570C
00580
00590
00600
00610
00620
00630
00640
00650
00660C
00670C
00680C
00690
00700
00710
00720
00730
00740
00750
00760
00770
00780
00790
00800
00810
00820
00830C
00840C
00850C
00860
00870
00880
00890
00900
00910
00920
00930
00940C
00950C
00960C
00970
00980
00990
01000
01010
01020
01030
01040

```
DO 24 I=1,N  
  READ(3,*) (U(I,J), I, J=1,2)  
24 CONTINUE  
  CALL MATMUL(R, U, VECID, N, 2, N)  
  CALL MATADD(A, VECID, VECID, N, N)  
  CALL PROG4(N, VECID, VECID, A, Z, I, PP, DD, FU1, U, UT)  
  RETURN  
END  
  
T(M,P)=A(M,N) * U(N,P)  
SUBROUTINE MATMUL(A,U,T,M,N,P)  
  INTEGER P  
  DIMENSION A(M,N), U(N,P), T(M,P)  
  DO 1 I=1,M  
    DO 1 J=1,P  
      T(I,J)=0.  
1 CONTINUE  
  DO 2 I=1,M  
    DO 2 J=1,P  
      DO 2 K=1,N  
        T(I,J)=A(I,K)*U(K,J)+T(I,J)  
2 CONTINUE  
  RETURN  
END  
  
C(M,N)=A(M,N) + B(M,N)  
SUBROUTINE MATADD(A,B,C,M,N)  
  DIMENSION A(M,N), B(M,N), C(M,N)  
  DO 3 I=1,M  
    DO 3 J=1,N  
      C(I,J)=A(I,J)+B(I,J)  
3 CONTINUE  
  RETURN  
END  
  
B(M,N)=S * A(M,N)  
SUBROUTINE SCALAT(S,A,B,M,N)  
  DIMENSION A(M,N), B(M,N)  
  DO 8 I=1,M  
    DO 8 J=1,N  
      B(I,J)=S*A(I,J)  
8 CONTINUE  
  RETURN  
END
```

```

010700.....V(N,M)=A(H,N) TRANPOSED
01080 SUBROUTINE MATTRA(A,V,H,M,N)
01090 DIMENSION A(M,N),V(N,M)
01100 DO 14 I=1,M
01110 DO 14 J=1,N
01120 V(J,I)=A(I,J)
01130 14 CONTINUE
01140 RETURN
01150 END

```

```

01160C
01170C
01180C

```

```

01190 SUBROUTINE PROG4(N,A,X,D,XN,F,P,DX,WA,D,DT)
01200 DIMENSION A(N,N),X(N,N),Q(N,N),XN(N,N),E(N,N),P(N,N),DX(N,N),
01210+WA(N),D(N,1),DT(1,N)
01220 WRITE(6,10)

```

```

01230 10 FORMAT(1X,'INPUT THE INTENSITY OF THE GUST NOISE, R.')
```

```

01240 READ(5,*)R
01250 CALL MATTRA(D,DT,N,1)
01260 CALL SCAMAT(R,D,D,N,1)
01270 CALL MATMUL(D,DT,D,N,1,N)
01280 CALL CAL(A,Q,XN,M,15,2,X,E,P,DX,WA)
01290 WRITE(6,35)

```

```

01300 NN=N-9
01310 ZZ=57.3*SQRT(XN(NN+1,NN+1))
01320 WRITE(6,38)ZZ
01330 ZZ=57.3*SQRT(XN(NN+2,NN+2))
01340 WRITE(6,39)ZZ
01350 ZZ=57.3*SQRT(XN(NN+4,NN+4))
01360 WRITE(6,40)ZZ
01370 ZZ=57.3*SQRT(XN(NN+5,NN+5))
01380 WRITE(6,41)ZZ

```

```

01390 38 FORMAT(1X,'OUTBOARD DEFLECTION =',E15.4)
01400 39 FORMAT(1X,'OUTBOARD RATE =',E15.4)
01410 40 FORMAT(1X,'INBOARD DEFLECTION =',E15.4)
01420 41 FORMAT(1X,'INBOARD RATE =',E15.4)
01430 35 FORMAT(1X,'RMS OF CONTROL RESPONSE '//)

```

```

01440 DO 24 I=1,N
01450 DO 24 J=1,N
01460 WRITE(2,164)XN(I,J)
01470 164 FORMAT(E16.8)
01480 24 CONTINUE
01490 CALL REPLACE(5HTAPE2,5HCOUST,0,0)
01500 RETURN
01510 END

```

```

01520C
01530C
01540
01550C

```

```

SUBROUTINE CAL(A,Q,XN,N,IMAX,IT,X,E,P,DX,WA)

```

```

01580 DIMENSION A(N,N), D(N,N), XN(N,N), XN(N,N)
01590 DIMENSION X(N,N), E(N,N), P(N,N), DX(N,N), WA(N)
01600 TR=0.
01610 DO 300 I=1,N
01620 300 TR=TR+A(I,I)
01630 FN=N
01640 IF (TR) 301, 302, 301
01650 301 ALF=ABS(TR)/FN
01660 302 ALF=1.
01670 303 CONTINUE
01680 EE=.01
01700 NC=N*(N+1)
01710 NC=NC/2
01720 DO 60 I=1,N
01730 DO 63 J=1,N
01740 60 TO(61,62), TT
01750 61 P(I,J)=A(I,J)
01760 X(I,J)=A(I,J)
01770 63 TO 63
01780 62 P(I,J)=A(J,I)
01790 X(J,J)=A(J,I)
01800 63 CONTINUE
01810 P(J,I)=P(I,I)-ALF
01820 X(J,I)=X(I,I)-ALF
01830 60 CONTINUE
01840C
01850C FTN= INVERSE(P)
01860C
01870 CALL MXLNER(P,N,N,DET,N,1,E-14,WA)
01880 DO 4 I=1,N
01890 DO 4 J=1,N
01900 E(I,J)=0.
01910 DO 4 K=1,N
01920 4 E(I,J)=E(I,J)+P(K,I)+0(K,J)*2.*ALF
01930 DO 5 I=1,N
01940 DO 5 J=1,N
01950 XN(I,J)=0.
01960 DO 5 K=1,N
01970 5 XN(I,J)=XN(I,J)+E(I,K)*P(K,I)
01980 DO 7 I=1,N
01990 DO 8 J=1,N
02000 8 P(I,J)=P(I,J)+2.*ALF
02010 7 P(I,J)=P(I,I)+1.
02020 ITER=0
02030 100 CONTINUE
02040 DO 9 I=1,N
02050 DO 9 J=1,N
02060 E(I,I)=0.

```

```
02080 Y E(I,J)=E(I,J)+P(K,I)*XN(K,J)
02090 DO 10 I=1,N
02100 DO 10 J=1,N
02110 DX(I,J)=0.
02120 DO 10 K=1,N
02130 DX(I,J)=DX(I,J)+E(I,K)*P(K,J)
02140 DO 20 I=1,N
02150 DO 20 J=1,N
02160 E(I,J)=P(I,J)
02170 DO 20 X(I,J)=P(I,J)
02180 DO 12 I=1,N
02190 DO 12 J=1,N
02200 XN(I,J)=XN(I,J)+DX(I,J)
02210 DO 12 XN(J,I)=XN(I,J)
02220 ICOT=0
02230 DO 15 I=1,N
02240 DO 15 J=1,N
02250 IF(XN(I,J))201,14,201
02260 201 RAT=ABS(DX(I,J)/XN(I,J))
02270 IF(RAT-EE)14,14,70
02280 14 ICOT=ICOT+1
02290 15 CONTINUE
02300 70 CONTINUE
02310 IF(ICOT-NC)16,18,16
02320 16 DO 17 I=1,N
02330 DO 17 J=1,N
02340 P(I,J)=0.
02350 DO 17 K=1,N
02360 P(I,J)=P(I,J)+E(I,K)*X(K,J)
02370 18 ITER=ITER+1
02380 IF(ICOT-NC)40,50,40
02390 40 IF(ITER-IMAX)100,50,50
02400 50 CONTINUE
02410 PRINT 600,ITER
02420 600 FORMAT(10X,'INNER LOOP TERMINATED AT ITER = ',(2)
02430 RETURN
02440 END
```

R2/12/15

MFNTS PROGRAM LOADS

```

00100 PROGRAM LOADS(INPUT,OUTPUT, TAPE5=INPUT, TAPE6=OUTPUT,COVST,
00110+TAPE3=COVST,LMAT,TAPE2=LMAT)
00120*****
00130C THIS PROGRAM INPUTS CLOSED LOOP STATE COVARIANCE MATRIX
00140C (OUTPUT FROM COV), LOADS INFLUENCE MATRIX AND OUTPUTS
00150C THE RMS SHEAR, RMS BENDING MOMENT, AND RMS TORQUE AT
00160C * VARIOUS STATIONS ALONG THE WING SPAN.
00170C*****
00180 REAL XN(42,42),T(10,42),LT(9,42),LTT(42,9),TEMP(42,9)
00190 DIMENSION KRET(8)
00200 WRITE(6,10)
00210 10 FORMAT(1X,'INPUT WHICH STATES WERE RETAINED(1) OR DELETED(0)?')
00220 READ(5,*)(KRET(J),J=1,8)
00230 KSUM=0
00240 DO 20 I=1,8
00250 KSUM=KSUM+KRET(I)
00260 20 CONTINUE
00270 WRITE(6,11)
00280 11 FORMAT(1X,'INPUT THE NUMBER OF LAG STATES?')
00290 READ(5,*)M
00300 KST=KSUM*(2+M)
00310 KKSUM=KST+9
00320 CALL PROG1(KST,KKSUM,KRET,XN,T,L,I,LT,TEMP)
00330 STOP
00340 END
00350C
00360C
00370 SUBROUTINE PROG1(KST,KKSUM,KRET,XN,T,I,I,LT,TEMP)
00380 REAL LRM(9,10),LSH(9,10),LTD(9,10),XN(KKSUM,KKSUM),T(10,KKSUM),
00390+LT(9,KKSUM),LTT(KKSUM,9),TEMP(KKSUM,9),COV(9,9),BM(9),SH(9),TD(9)
00400 DIMENSION KRET(8)
00410 CALL GETPF(5HTAPE2,4HLMAT,0,0)
00420 DO 10 I=1,10
00430 DO 10 J=1,9
00440C
00450C READ IN LOADS INFLUENCE MATRIX
00460C
00470 READ(2,15)LRM(J,I),LSH(J,I),LTD(J,I)
00480 15 FORMAT(3E16.8)
00490 10 CONTINUE
00500 CALL GETPF(5HTAPE3,5HCOVST,0,0)
00510 DO 20 I=1,KKSUM
00520 DO 20 J=1,KKSUM

```

```
005500 READ(3,15)XN(I,J)
00560 20 CONTINUE
00570 DO 40 I=1,10
00580 DO 40 J=1,KKSUM
00590 T(I,J)=0.
00600 40 CONTINUE
00610 J=0
00620 DO 50 I=1,8
00630 IF(KRET(I).EQ.0)GO TO 50
00640 J=J+1
00650 T(I,J)=1.
00660 50 CONTINUE
00670 T(9,KST+1)=.514
00680 T(10,KST+4)=.518
00690 CALL MATMUL(LRN,T,I,T,9,10,KKSUM)
00700 CALL MATTRA(LT,LTT,9,KKSUM)
00710 CALL MATMUL(XN,LTT,TEMP,KKSUM,KKSUM,9)
00720 CALL MATMUL(LT,TEMP,COV,9,KKSUM,9)
00730 DO 60 I=1,9
00740 BK(I)=SORT(COV(I,I))
00750 60 CONTINUE
00760 CALL MATMUL(LSH,T,LT,9,10,KKSUM)
00770 CALL MATTRA(LT,LTT,9,KKSUM)
00780 CALL MATMUL(XN,LTT,TEMP,KKSUM,KKSUM,9)
00790 CALL MATMUL(LT,TEMP,COV,9,KKSUM,9)
00800 DO 70 I=1,9
00810 SK(I)=SORT(COV(I,I))
00820 70 CONTINUE
00830 CALL MATMUL(ITO,T,LT,9,10,KKSUM)
00840 CALL MATTRA(LT,LTT,9,KKSUM)
00850 CALL MATMUL(XN,LTT,TEMP,KKSUM,KKSUM,9)
00860 CALL MATMUL(LT,TEMP,COV,9,KKSUM,9)
00870 DO 80 I=1,9
00880 TR(I)=SORT(COV(I,J))
00890 80 CONTINUE
00900 WRITE(6,90)
00910 90 FORMAT(35X,'RMS LOADS '/1X,'BEND MOM',27X,'SHEAR',27X,'TORQUE'//)
00920 DO 100 I=1,9
00930 100 FORMAT(5X,E16.4,10X,E16.4,10X,E16.4)
00940 WRITE(6,110)BK(I),SH(J),TO(I)
00950 110 CONTINUE
00960 RETURN
00970 ENT
00980 00990
01000 SUBROUTINE MATMIII (A,H,T,M,N,P)
01010 INTGFR P
01020 DIMENSION A(M,N),H(N,P),T(M,P)
01030
```

```
0001 1 1,2,F
01060 T(I,J)=0.
01070 1 CONTINUE
01080 DO 2 I=1,M
01090 DO 2 J=1,P
01100 DO 2 K=1,N
01110 T(I,J)=A(I,K)*U(K,J)+T(I,J)
01120 2 CONTINUE
01130 RETURN
01140 END
01150C
01160C
01170 SUBROUTINE MATTRA(A,V,M,N)
01180 DIMENSION A(M,N),V(N,M)
01190 DO 14 I=1,M
01200 DO 14 J=1,N
01210 V(J,I)=A(I,J)
01220 14 CONTINUE
01230 RETURN
01240 END
```


R7/12/15

AWFETS PROGRAM FEEDRKE

```

00100 PROGRAM FEEDRKE(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,ONELEF,
00110+TAPE2=ONELEF,MASS,TAPE1=MASS,GAIN1,TAPE3=GAIN1,TAPE4)
00120C*****
00130C THIS PROGRAM INPUTS THE STRUCTURAL MASS,STRUCTURAL STIFFNESS,
00140C STRUCTURAL DAMPING, AERODYNAMIC INFLUENCE MATRICES, FLIGHT
00150C CONDITION, FEEDBACK GAINS(OUTPUT OF MODAL) AND OUTPUTS THE
00160C CLOSED LOOP EIGENVALUES. THIS IS USEFUL TO STUDY OFF DESIGN
00170C PERFORMANCE.
00180C*****r****
00190 REAL MXX(8,8),CS(8,8),KS(8,8),CT(8,8),KT(8,8),CC(8,8),
00200+KK(8,8),A2X(8,8),A1X(8,8),A0X(8,8),MINV(8,8),A2X0(8,8)
00210+,A1X0(8,8),A0X0(8,8),A1(16,16),A2(32,32),00(8,2),DUM1(8,7),
00220+DUM2(8,7),DUM3(8,2),DUM4(8,7),DUM5(8,2),DUM6(8,2),DUM7(8,1)
00230 REAL DUM8(8,1),A0U(8,2),A1U(8,2),EI(8,2,4),R(8,2),
00240+EI16(8,7,4),FI(8,4),A3(40,40),B1(40,2),B2(32,2),II(8,8),
00250+OI(8,8,4),KI(4),A10(8,1),A0D(8,1),I(8,1),U(8,1),A4(42,42),
00260+B4(42,2),W(42,1),B1(16,2),A5(24,24),B5(24,2)
00270 WRITE(6,8)
00280 B FORMAT(1X,'INPUT THE ORDER OF THE STRUCTURAL-RIGID BODY STATE VECTO
00290+R(N)')
00300 READ(5,*)N
00310 WRITE(6,10)
00320 10 FORMAT(1X,'INPUT VELOCITY(V), CHORD(C), DYNAMIC PRESSURE(Q), NUMBE
00330+R OF LAG STATES(L)')
00340C
00350C INPUT FLIGHT CONDITIONS
00360C
00370 READ(5,*)V,C,Q,L
00380 DO 20 I=1,L
00390 WRITE(6,25)I
00400C
00410C INPUT AERODYNAMIC LAG FREQUENCIES
00420C
00430 READ(5,*)KI(I)
00440 20 CONTINUE
00450 25 FORMAT(1X,'INPUT KI(',I1,')')
00460 N2=?*N
00470 KEND=N*(L+2)
00480 KEND1=KEND+7
00490 KEND2=KEND+2
00500 K5=2*N+7
00510 CALL PROG1(N,N2,KEND,KEND2,K5,V,C,Q,L,MXX,CS,KS,CT,KI,
00520+CC,KK,A2X,A1X,A0X,MINV,A2X0,A1X0,A0X0,A1,A2,00,DUM1,DUM2,DUM3,DUM4
00530+,DUM5,DUM6,DUM7,DUM8,A0U,A1U,EI,R,EI16,IJ,A3,B3,B2,FI,DI,KI,A1D,

```

```

00540 SUBROUTINE PROGI(N,N2,KEND,KENDD,KEND2,K5,V,C,D,L,MXX,CS,KS,CT,KT,
00560 END
00570F
00580C
00590 SUBROUTINE PROGI(N,N2,KEND,KENDD,KEND2,K5,V,C,D,L,MXX,CS,KS,CT,KT,
00600+CC, KK, A2X, A1X, A0X, MINV, A2XD, A1XD, A0XD, A1, A2, A0, DUM1, DUM2, DUM3, DUM4
00610+; DUM5, DUM6, DUM7, DUM8, A0U, A1U, ET, R, ETJ6, IT, A3, B3, B2, F1, D1, KI, A1D,
00620+A0D, T, U, A4, R1, B4, W, A5, B5)
00630 REAL LAMDA, LL
00640 INTEGER FLAG
00650 COMMON KRET(42)
00660 REAL MXX(N,N), CS(N,N), KS(N,N), CT(N,N), KT(N,N), CC(N,N), KI(N,N),
00670+A2X(N,N), A1X(N,N), A0X(N,N), MINV(N,N), A2XD(N,N), A1XD(N,N), A0XD(N,N),
00680+A1(N2,N2), A2(KEND,KEND), DD(N,2), DUM1(N,7), DUM2(N,7), DUM3(N,2),
00690+DUM4(N,7), DUM5(N,2), DUM6(N,2), DUM7(N,1), DUM8(N,1), A0U(N,2)
00700 REAL A1J(N,2),
00710+EI(N,2,L), G(7,7), H(7,2), R(N,2), J6(2,7), ET J6(N,7,L), FT(N,L),
00720+JH(2,2), A3(KEND,KEND), B3(KEND,2), B2(KEND,2), II(N,N), DI(N,N,L),
00730 REAL KI(L), A1B(N,1), A0D(N,1), I(N,1), U(N,1), A4(KEND2,KEND2),
00740+B1(N2,2), B4(KEND2,2), W(KEND2,1),
00750+A5(K5,K5), B5(K5,2), JJ(2,7)
00760 REAL AA(42,42), BB(42,2)
00770 CALL GETPF(5HTAPE1,4HMASS,0,0)
00780 DO 1 I=1,N
00790 DO 1 J=1,N
00800C
00810C READ IN STRUCTURAL DATA
00820C
00830 READ(1,101)MXX(I,J),CS(I,J),KS(I,J)
00840 1 CONTINUE
00850 DO 3 I=1,7
00860 DO 3 J=1,7
00870C
00880C READ IN ACTUATOR DATA
00890C
00900 READ(1,101)G(I,J)
00910 3 CONTINUE
00920 DO 4 I=1,7
00930 READ(1,101)H(I,J),J=1,2)
00940 4 CONTINUE
00950 DO 5 J=1,7
00960 READ(1,101)(JJ(I,J),I=1,2)
00970 5 CONTINUE
00980 CALL GETPF(5HTAPE2,6HONELEE,0,0)
00990 DO 430 K=1,L
01000 DO 410 J=1,N
01010 DO 410 I=1,N
01020C
01030C READ IN AERODYNAMIC INFLUENCE DATA
01040C

```

```
01070 DO 420 I=1,2
01080 DO 420 I=1,N
01090 READ(2,101)A00(I,1),A10(I,J),E1(I,I7)G
01100 420 CONTINUE
01110 DO 430 I=1,N
01120 READ(2,101)A00(I,K),A10(I,K),FI(I,K)
01130 FI(I,K)=FI(I,K)/V
01140 430 CONTINUE
01150 101 FORMAT(3E16.8)
01160 109 FORMAT(4E16.8)
01170 EPS=1.E-10
01180 TEMP=0*(C/2./V)**2
01190 TEMP2=0+C/2./V
01200 CALL SCAMAT(TEMP,A2X,A2X0,MJNV,M,N)
01210 CALL MATADD(MXX,A2X0,MJNV,M,N)
01220C
01230C
01240C
01250 CALL MXLNEQ(MJNV,N,N,DET,IRANK,EPS,A2X0,0)
01260 CALL SCAMAT(TEMP2,A1X,A1X0,M,N)
01270 CALL MATADD(CS,A1X0,CT,N,N)
01280 CALL SCAMAT(0,A0X,A0X0,N,N)
01290 CALL MATADD(KS,A0X0,KT,N,N)
01300 CALL SCAMAT(-1.,MINV,MJNV,M,N)
01310 CALL MATMUL(MJNV,KT,KN,M,N)
01320 CALL MATMUL(MJNV,CT,CC,N,M,N)
01330 DO 15 I=1,N
01340 DO 14 J=1,N
01350 II(I,J)=0.
01360 14 CONTINUE
01370 II(I,I)=1.0
01380 15 CONTINUE
01390 N21=N2+1
01400 DO 18 I=1,N
01410 DO 18 J=1,N
01420 A1(I,J)=0.
01430 A1(I,J+N)=II(I,J)
01440 A1(I+N,J)=KK(I,J)
01450 A1(I+N,J+N)=CC(I,J)
01460 18 CONTINUE
01470 IF(1.E0,0)GO TO 102
01480 DO 20 I=1,KEND
01490 DO 20 J=1,KEND
01500 A2(I,J)=0.
01510 20 CONTINUE
01520 DO 25 I=1,N
01530 DO 25 J=1,N
01540 A2(I,J)=A1(I,J)
01550 A2(I+e,J)=A1(I+N,I)
```

```
01580 DO 25 K=1,L
01590 KN=K*N
01600 A2(T+N,KN+N+J)=MINV(I,J)
01610 A2(KN+N+I,J+N)=DI(I,J,K)
01620 A2(KN+N+I,KN+N+I)=-2.*U*(I(K)/C
25 CONTINUE
01630 SCA=0*C/2./U
01640 CALL SCAMAT(SCA,A11,00,N,2)
01650 CALL SCAMAT(D,AOU,R,N,2)
01660 DO 36 I=1,KENDI
01670 DO 36 J=1,KENDJ
01680 A3(I,J)=0.
01690
36 CONTINUE
01700 DO 37 I=1,KEND
01710 DO 37 J=1,KEND
01720 A3(I,J)=A2(I,J)
01730
37 CONTINUE
01740 DO 38 I=1,7
01750 DO 38 J=1,7
01760 A3(KEND+I,KEND+J)=6(I,J)
01770
38 CONTINUE
01780 CALL MATMUL(JJ,G,JG,2,7,7)
01790 CALL MATMUL(OO,JG,DUM1,N,2,7)
01800 CALL MATMUL(R,JJ,DUM2,N,2,7)
01810 CALL MATADD(DUM2,DUM1,DUM2,N,7)
01820 CALL MATMUL(MINV,DUM2,DUM1,N,N,7)
01830 DO 40 K=1,L
01840 DO 39 I=1,N
01850 DO 39 J=1,2
01860 DUM3(I,J)=EI(I,J,K)
01870
39 CONTINUE
01880 CALL MATMUL(DUM3,JG,DUM4,N,2,7)
01890 DO 40 I=1,N
01900 DO 40 J=1,7
01910 EIJG(I,J,K)=DUM4(I,J)
01920
40 CONTINUE
01930 DO 45 I=1,N
01940 DO 45 J=1,7
01950 A3(I+N,J+KEND)=DUM1(I,I)
01960 DO 45 K=1,L
01970 KN=K*N
01980 A3(KN+N+I,J+KEND)=EIJG(I,I,K)
01990
45 CONTINUE
02000 CALL MATMUL(JJ,H,JH,2,7,2)
02010 CALL MATMUL(OO,JH,DUM3,N,2,2)
02020 CALL MATMUL(MINV,DUM3,DUM5,N,N,2)
02030 DO 47 K=1,L
02040 DO 46 I=1,N
02050 DO 46 J=1,2
02060
```

```
02090 CALL MATMUL(DUM3, JH, DIM6, N, 2, 2)
02100 DO 47 I=1, N
02110 DO 47 J=1, 2
02120 EIG(I, J, K)=DIM6(I, J)
02130 47 CONTINUE
02140 DO 48 I=1, N
02150 DO 48 J=1, 2
02160 R3(I, J)=0.
02170 R3(I+N, J)=DIM5(I, J)
02180 DO 48 K=1, L
02190 KN=K*N
02200 R3(I+KN+N, J)=EIG(I, J, K)
02210 48 CONTINUE
02220 DO 49 I=1, 7
02230 DO 49 J=1, 2
02240 R3(I+KEND, J)=H(I, J)
02250 49 CONTINUE
02260 WRITE(6, 56)
02270 56 FORMAT(IX, 'INPUT CHARACTERISTIC LENGTH(LL) OF NOISE//')
02280 READ(5, *)LL
02290 TEMP3=TEMP2/V
02300 CALL SCAMAT(TEMP3, A1D, T, N, 1)
02310 Q1=D/V
02320 CALL SCAMAT(Q1, A0D, U, N, 1)
02330 CALL MATMUL(MINV, U, DIM7, N, N, 1)
02340 CALL MATMUL(MINV, T, DIM8, N, N, 1)
02350 DO 58 I=1, KEND
02360 DO 58 J=1, 2
02370 R4(I, J)=R3(I, J)
02380 R4(KEND+1, J)=0.
02390 R4(KEND+2, J)=0.
02400 58 CONTINUE
02410 DO 59 I=1, KEND
02420 DO 59 J=1, KEND
02430 A4(I, J)=0.
02440 59 CONTINUE
02450 DO 60 I=1, KEND
02460 DO 60 J=1, KEND
02470 A4(I, J)=A3(I, J)
02480 60 CONTINUE
02490 A4(KEND+1, KEND+2)=1.
02500 A4(KEND+2, KEND+1)=-1.001*(U/LL)*#2
02510 A4(KEND+2, KEND+2)=-2.001*U/LL
02520 DO 62 I=1, N
02530 A4(I+N, KEND+1)=DIM7(I, 1)
02540 A4(I+N, KEND+2)=DIM8(I, 1)
02550 DO 62 K=1, L
02560 KN=K*N
02570 A4(I+KN+N, KEND+2)=EIG(I, K)
```

```
02600 W(J,1)=0.
02610 64 CONTINUE
02620 XX=1.732*V/LL
02630 W(KEND0+1,1)=XX
02640 W(KEND0+2,1)=-2.464*(V/LL)*#2
02650 CALL SCAMAT(XX,DUM8,DUM8,N,1)
02660 DO 66 J=1,N
02670 W(I+N,1)=DUM8(I,1)
02680 DO 66 K=1,L
02690 KN=K*N
02700 W(I+N+KN,1)=XX*F(I,K)
02710 66 CONTINUE
02720 FLAG=4
02730 60 TO 200
02740 102 SCA=0+C/2./U
02750 CALL SCAMAT(SCA,AIU,00,N,2)
02760 CALL SCAMAT(0,AOU,R,N,2)
02770 DO 106 I=1,K5
02780 DO 106 J=1,K5
02790 A5(I,J)=0.
02800 106 CONTINUE
02810 CALL MATMUL(JJ,H,JH,2,7,2)
02820 CALL MATMUL(OO,JH,DUM3,N,2,2)
02830 CALL MATMUL(MINV,DUM3,DUM3,N,N,2)
02840 DO 108 I=1,N
02850 DO 108 J=1,2
02860 B5(I,J)=0.
02870 B5(I+N,J)=DUM5(I,J)
02880 108 CONTINUE
02890 DO 110 I=1,7
02900 DO 110 J=1,2
02910 B5(I+N2,J)=H(I,J)
02920 110 CONTINUE
02930 DO 112 I=1,N2
02940 DO 112 J=1,N2
02950 A5(I,J)=A1(I,J)
02960 112 CONTINUE
02970 CALL MATMUL(JJ,G,JG,2,7,2)
02980 CALL MATMUL(OO,JG,DUM1,N,2,2)
02990 CALL MATMUL(R,JJ,DUM2,N,2,7)
03000 CALL MATADD(DUM2,DUM1,DUM2,N,7)
03010 CALL MATMUL(MINV,DUM2,DUM1,N,N,7)
03020 DO 114 I=1,N
03030 DO 114 J=1,7
03040 A5(I+N,J+N2)=DUM1(I,J)
03050 114 CONTINUE
03060 DO 116 I=1,7
03070 DO 116 J=1,7
03080 A5(I+N2,J+N2)=G(I, J)
```

```
03110 200 WRITE(6,201)
03120 201 FORMAT(1X,' INPUT RETAIN(I) OR DELETE(0) FOR EACH STATE MEMBER')
03130 KSTATE=N
03140 KSUM=0
03150 READ(5,*)(KRET(J),J=1,KSTATE)
03160 DO 205 I=1,KSTATE
03170 KRET(I+N)=KRET(I)
03180 IF(KRET(I).EQ.1)KSUM=KSUM+1
03190 205 CONTINUE
03200 IF(L.EQ.0)GO TO 207
03210 DO 206 K=1,L
03220 DO 206 I=1,N
03230 KN=N*(K+1)+I
03240 KRET(KN)=KRET(I)
03250 206 CONTINUE
03260 DO 209 I=1,9
03270 KRET(KEND+I)=1
03280 209 CONTINUE
03290 207 GO TO(310,370,330,340,350),FLAG
03300 310 KSUM=2*KSUM
03310 CALL PR062(FLAG,KSUM,A1,B1,N2,AA,BB)
03320 GO TO 900
03330 320 KSUM=(L+2)*KSUM
03340 CALL PR062(FLAG,KSUM,A2,B2,KEND,AA,BB)
03350 GO TO 900
03360 330 KSUM=7+(L+2)*KSUM
03370 CALL PR062(FLAG,KSUM,A3,B3,KEND,AA,BB)
03380 GO TO 900
03390 340 KSUM=9+(L+2)*KSUM
03400 CALL PR062(FLAG,KSUM,A4,B4,KEND,AA,BB)
03410 GO TO 900
03420 350 KSUM=7+2*KSUM
03430 DO 351 I=N21,K5
03440 KRET(I)=1
03450 351 CONTINUE
03460 CALL PR062(FLAG,KSUM,A5,B5,K5,AA,BB)
03470 900 RETURN
03480 END
03490C
03500C
03510C
03520C
03530C
03540C
03550C
03560C
03570C
03580C
03590C
```

SUBROUTINE MATDEL(A,N,AA,KSUM)

THIS ROUTINE DELETES DESIRED STATES FROM THE A MATRIX

DIMENSION A(N,N),AA(KSUM,KSUM),JRET(40)

COMMON KRET(42)

I=0

DO 100 I=1,N

IF(KRET(I).EQ.0)GO TO 100

```
03620 100 CONTINUE
03630 DO 200 I=1,KSUM
03640 DO 200 J=1,KSUM
03650 AA(I,J)=A(JRET(I),JRET(J))
03660 200 CONTINUE
03670 RETURN
03680 END
03690C
03700C
03710 SUBROUTINE MATDEL(B,N,BB,KSUM)
03720C
03730C THIS ROUTINE DELETES DESIRED STATES FROM THE B MATRIX(X
03740C
03750 DIMENSION B(N,2),BB(KSUM,2),JRET(40)
03760 COMMON KRET(42)
03770 J=0
03780 DO 100 I=1,N
03790 IF(KRET(I).EQ.0)GO TO 100
03800 J=J+1
03810 JRET(J)=I
03820 100 CONTINUE
03830 DO 200 I=1,KSUM
03840 DO 200 J=1,2
03850 BB(I,J)=B(JRET(I),J)
03860 200 CONTINUE
03870 RETURN
03880 END
03890C
03900C
03910C.....T(M,P)=A(M,N) * U(N,P)
03920 SUBROUTINE MATMUL(A,I,T,H,N,P)
03930 INTEGER P
03940 DIMENSION A(M,N),U(N,P),T(M,P)
03950 DO 1 T=1,H
03960 DO 1 J=1,P
03970 T(I,J)=0.
03980 1 CONTINUE
03990 DO 2 T=1,H
04000 DO 2 J=1,P
04010 DO 2 K=1,H
04020 T(T,J)=A(T,K)*U(K,J)+T(T,J)
04030 2 CONTINUE
04040 RETURN
04050 END
04060C
04070C
04080C.....C(M,N)=A(M,N) + B(M,N)
04090 SUBROUTINE MATADD(A,B,C,M,N)
04100 DIMENSION A(M,N),B(M,N),C(M,N)
```



```
04130 C(I,J)=A(I,J)+B(J,I)
04140 3 CONTINUE
04150 RETURN
04160 END
04170C
04180C
04190C.....B(N,N)=S * A(N,N)
04200 SUBROUTINE SCAMAT(S,A,B,M,N)
04210 DIMENSION A(N,N),B(M,N)
04220 DO 8 I=1,M
04230 DO 8 J=1,N
04240 R(I,J)=S*A(I,J)
04250 8 CONTINUE
04260 RETURN
04270 END
04280C
04290C
04300C.....V(N,M)= A(N,N) TRANPOSED
04310 SUBROUTINE MATTRA(A,V,M,N)
04320 DIMENSION A(N,N),V(N,M)
04330 DO 14 I=1,M
04340 DO 14 J=1,N
04350 V(J,I)=A(I,J)
04360 14 CONTINUE
04370 RETURN
04380 END
04390C
04400C
04410 SUBROUTINE PROG2(FLAG,KSUM,A,B,IP,AA,BB)
04420 INTEGER FLAG,IV1(42)
04430 REAL AA(KSUM,KSUM),BB(KSUM,2),A(IP,IP),R(IP,2),FU1(42)
04440 REAL UR(42),UI(42),Z(47,47),G(2,42)
04450 GO TO(710,320,330,340,350),FLAG
04460 100 FORMAT(3E16.8)
04470 710 CALL MATDEL(A,IP,AA,KSUM)
04480 DO 311 I=1,KSUM
04490 DO 311 J=1,KSUM
04500 WRITE(4,100)AA(I,J)
04510 311 CONTINUE
04520 CALL PROG3(KSUM,AA,UR,UI,Z,IV1,FU1,BB,G)
04530 GO TO 900
04540 320 CALL MATDEL(A,IP,AA,KSUM)
04550 DO 321 I=1,KSUM
04560 DO 321 J=1,KSUM
04570 WRITE(4,100)AA(I,J)
04580 321 CONTINUE
04590 CALL PROG3(KSUM,AA,UR,UI,Z,IV1,FU1,BB,G)
04600 GO TO 900
04610 330 CALL MATDEL(A,IP,AA,KSUM)
```

```
04650
04640 DO 331 J=1,KSUM
04650 WRITE(4,100)AA(I,J)
04660 331 CONTINUE
04670 DO 332 I=1,KSUM
04680 WRITE(4,100)BB(I,1),BB(I,2)
04690 332 CONTINUE
04700 CALL PROG3(KSUM,AA,UR,UI,Z,IV1,FV1,RR,6)
04710 GO TO 900
04720 340 CALL MATREL(A,IP,AA,KSUM)
04730 CALL MATDEL(B,IP,RR,KSUM)
04740 DO 341 I=1,KSUM
04750 DO 341 J=1,KSUM
04760 WRITE(4,100)AA(I,J)
04770 341 CONTINUE
04780 DO 342 I=1,KSUM
04790 WRITE(4,100)BB(I,1),BB(I,2)
04800 342 CONTINUE
04810 CALL PROG3(KSUM,AA,UR,UI,Z,IV1,FV1,RR,6)
04820 GO TO 900
04830 350 CALL MATDEL(A,IP,AA,KSUM)
04840 CALL MATDEL(B,IP,RR,KSUM)
04850 DO 351 I=1,KSUM
04860 DO 351 J=1,KSUM
04870 WRITE(4,100)AA(I,J)
04880 351 CONTINUE
04890 DO 352 I=1,KSUM
04900 WRITE(4,100)BB(I,1),BB(I,2)
04910 352 CONTINUE
04920 CALL PROG3(KSUM,AA,UR,UI,Z,IV1,FV1,RR,6)
04930 900 RETURN
04940 END
04950C
04960C
04970 SURROUTINE PROG3(N,AA,UR,UI,Z,IV1,FV1,RR,G)
04980 COMMON KRET(42)
04990 DIMENSION AA(N,N),UR(N),UI(N),Z(N,N),IV1(N),FV1(N),6(2,N),BB(N,2)
05000C
05010C READ IN FEEDBACK GAINS OUTPUT FROM PROGRAM: MODAL
05020C
05030 CALL GETPF(5HTAPE3,5HGAINL,0,0)
05040 DO 24 I=1,N
05050 READ(7,*)G(J,I),J=1,2)
05060 24 CONTINUE
05070 CALL MATMUL(BB,G,Z,N,2,N)
05080 CALL MATADD(AA,Z,AA,N,N)
05090C
05100C CALCULATE CLOSED LOOP ETGENVALUES
05110C
05120 CALL RR(N,N,AA,UR,UI,Z,IV1,FV1,RR)
```

01170 3 CONTINUE(20X,I2,/,2X,E15.4,15X,E15.4)

05150+ THASINARY PART')

05160 DO 30 I=1,N

05170 WRITE(6,25)I,WR(I),WI(I)

05180 25 FORMAT(20X,I2,/,2X,E15.4,15X,E15.4)

05190 30 CONTINUE

05200 RETURN

05210 END