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
Λ = 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

List of Figures

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 precedure 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°/s. The rms deflection of the outboard aileron was limited to 15° and the deflection rate to 740°/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

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

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

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

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

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

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

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

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

The first column of the matrix $[A_o]$ 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 $[A_o]$ 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]

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

The singular values of E are defined as the positive square roots of the eigenvalues of E^*E. If the β_m's are chose to minimize $\overline{\sigma}(E)$, 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 $\overline{\sigma}(E)$ 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 $[A_o]$ was set equal to zero, the second column of $[A_o]$ 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 $[A_o]$, $[A_1]$, $[A_2]$ 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	$-1.74 \pm j6.29$
1st flexure	$41.6 \pm j118.1$
3rd flexure	$-0.7 \pm j136.6$
4th flexure	$-97.1 \pm j108.4$
6th flexure	$-16.9 \pm j218.3$
8th flexure	$-4.1 \pm j397.4$
9th flexure	$-11.0 \pm j425.0$
10th flexure	$-24.8 \pm j452.5$

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 $0.0\pm j1.1\times10^{-3}$. 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.

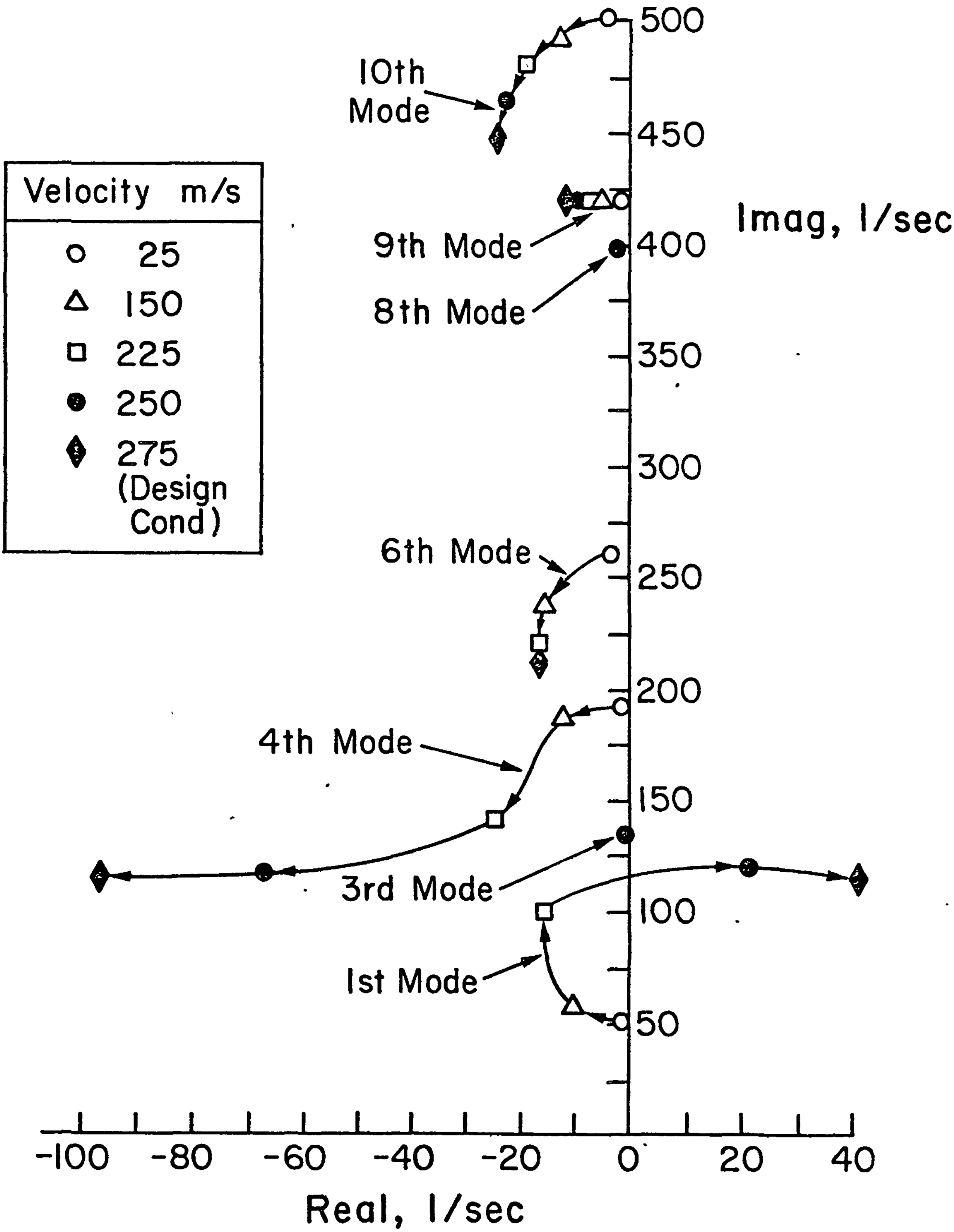


Fig. 1 Root Locus of Seven Mode Model of Uncontrolled Wing as Velocity is Varied

Some possible indices are as follows:

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

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

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.

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

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 exculsively 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

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

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

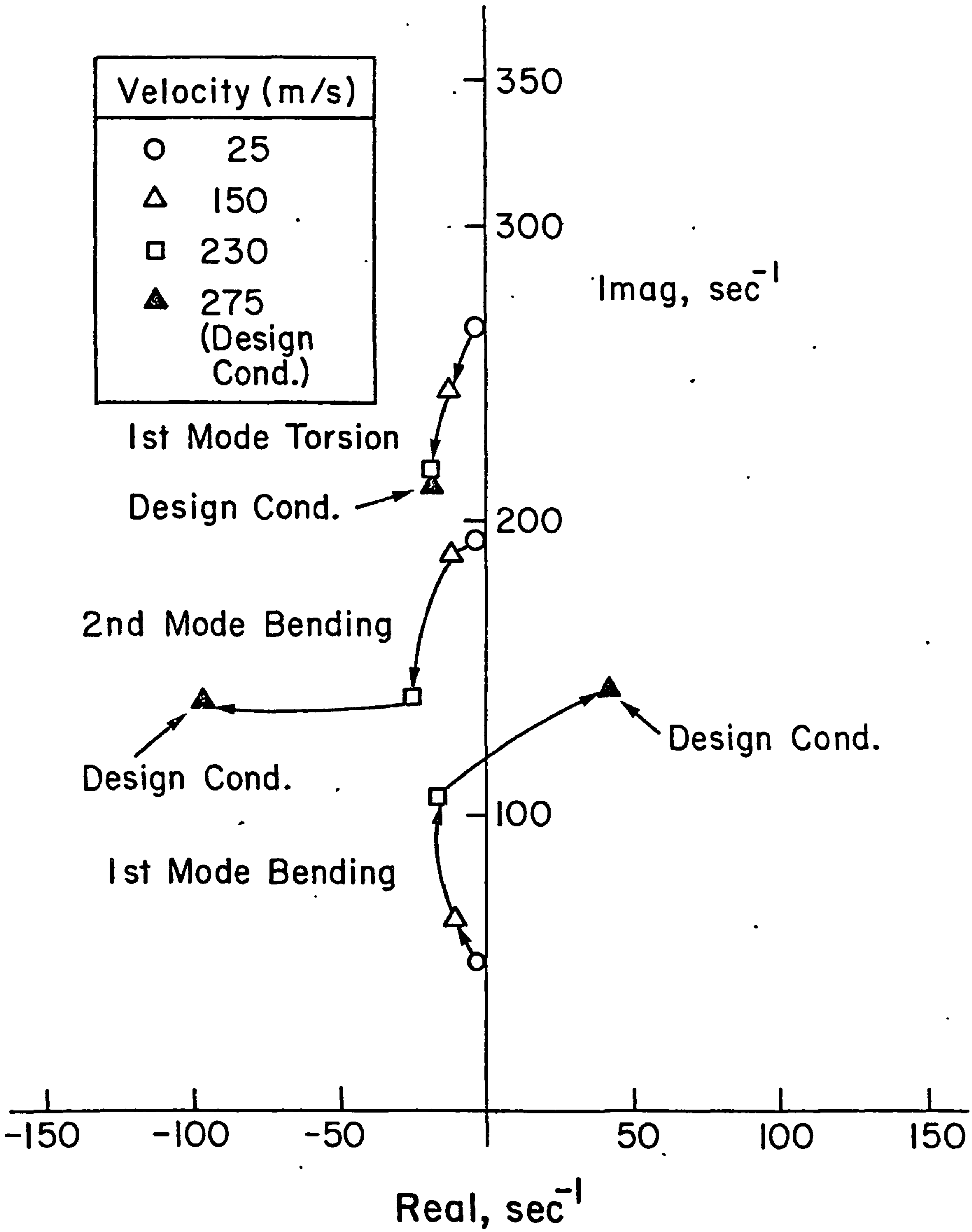


Fig. 2 Root Locus of Three Mode Model of Uncontrolled Wing as Velocity is Varied

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

$$\dot{x} = Ax + Bu + \Gamma\eta$$

with measurements

$$y = Cx$$

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

$$u = Ky = KCx$$

the the closed loop system response is

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

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

$$(A + BKC)v_i = \lambda_i v_i \tag{10}$$

Introduce $w \in R^m$ where

$$w_i = KCv_i \tag{11}$$

The problem becomes one of determing 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

$$(\lambda_i I - A)v_i = Bw_i \tag{12}$$

and

$$w_i = Kv_i \tag{13}$$

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

$$V\Lambda - AV = BW \tag{14}$$

Likewise from Eq. (13)

$$W = KV \tag{15}$$

therefore

$$K = WV^{-1} \tag{16}$$

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,...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 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

$$v_i = (\lambda_i I-A)^{-1}Bw_i \qquad (17)$$

or

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

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

$$(\lambda_i I - A) v_i = B w_i$$

becomes in entirely real terms

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

and

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

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

where

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

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

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

where

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

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

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

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

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

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

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

yields

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

Now the left hand eigenvector matrix is nonsingular so

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

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*
Unstable Roots Rotated About Imag. Axis ES	0.5	5.0	108.0	412.0	3.3	31.4	509.0	1464.0
Unstable Roots Rotated About Imag. Axis LQ	1.8	*	271.0	*	2.8	*	407.0	*
Eigenvector shaping to Reduce Inbd Rate (Final ES)	0.9	8.1	86.0	279.7	4.7	31.2	612.0	1464.0
Weighting Inbd Rate in Perf Index (Final LQ)	0.7	4.6	85.0	284.1	3.1	30.2	486.0	1335.1
Final ES Inbd Failed	-	-	-	-	4.4	26.5	572.0	1427.0
Final LQ Inbd Failed	-	-	-	-	3.5	30.0	519.0	1410.0
Max Allowable	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

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

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.

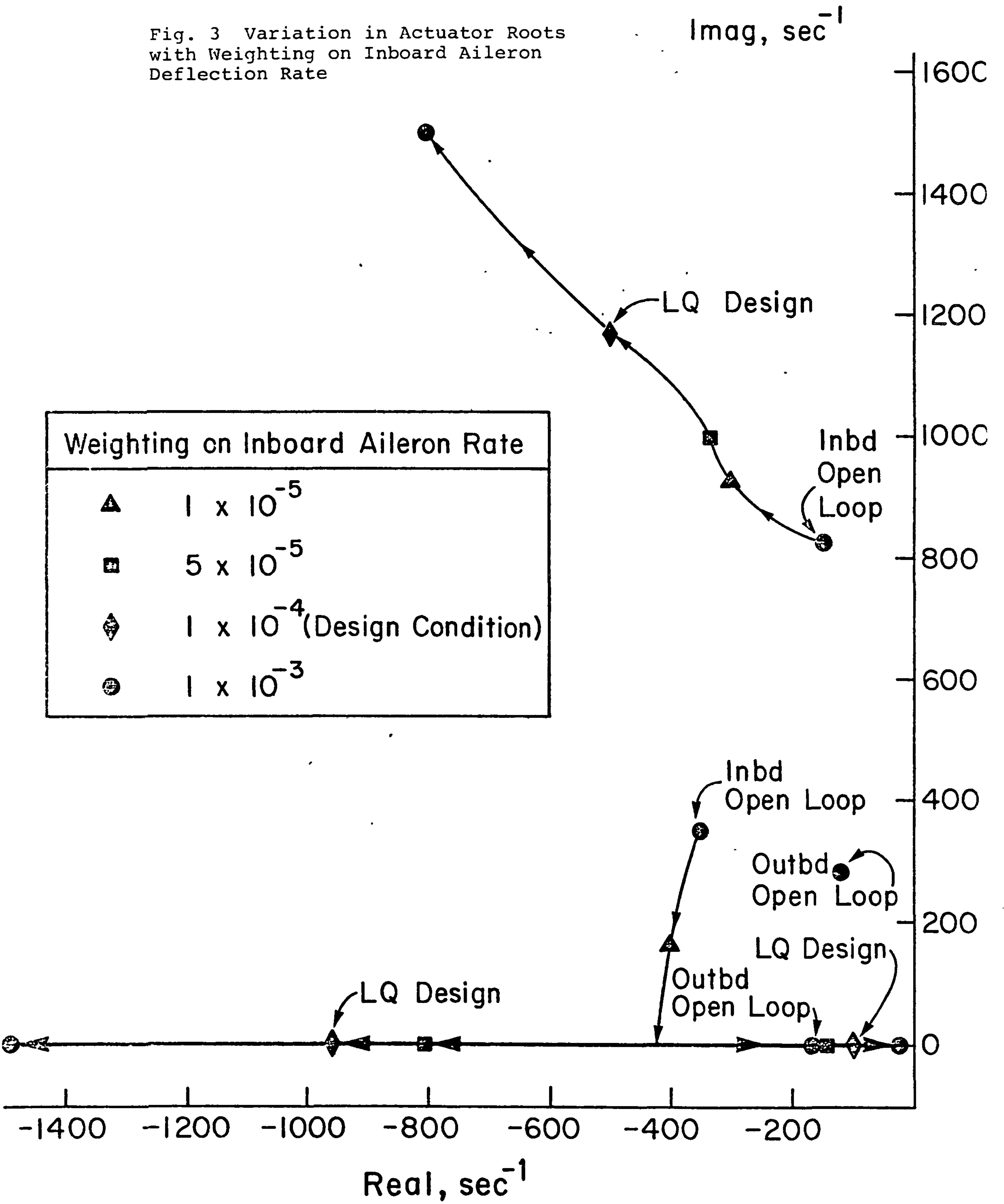


Fig. 3 Variation in Actuator Roots with Weighting on Inboard Aileron Deflection Rate

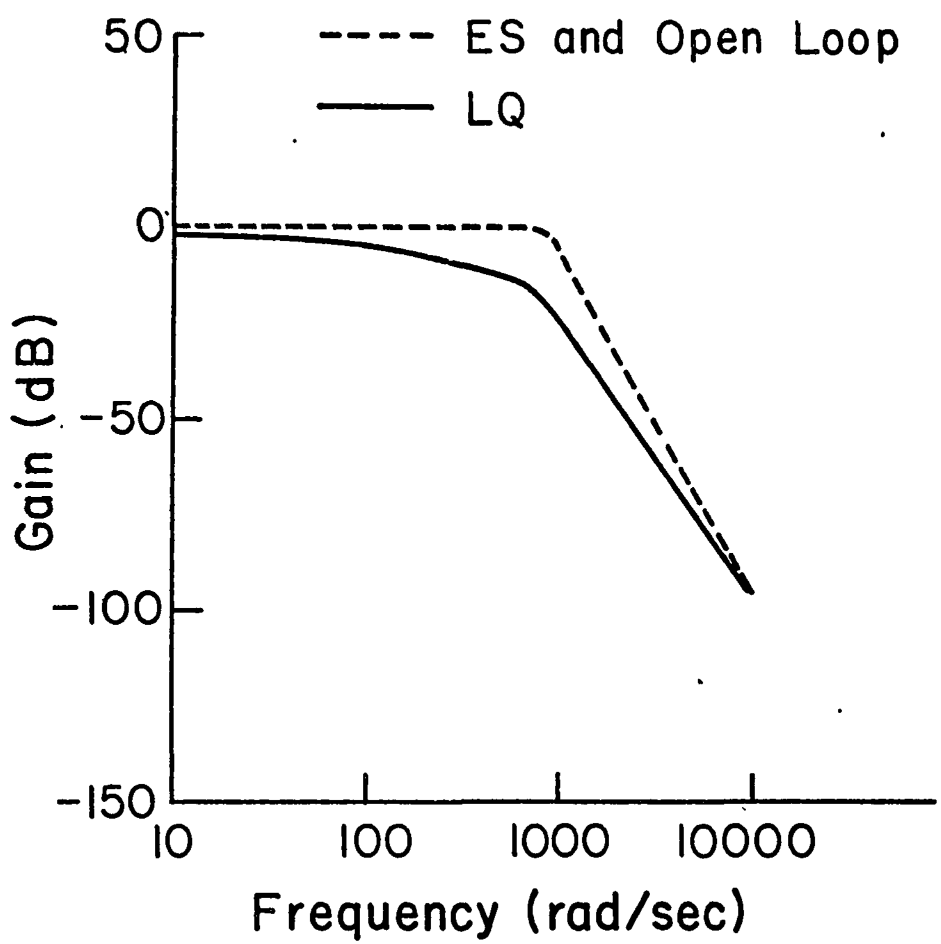


Fig. 4 Inboard Actuator Transfer Function Frequency Response

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 manuever 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	*Bending Moment (N-M)*	5740	4530	3426	2444	1371	889	377	83	5
	Shear (N)	92	82	74	63	42	36	24	6	1
	Torque (N-M)	36	54	73	82	73	63	52	6	2
FINAL ES	*Bending Moment (N-M)*	5558	4412	3359	2418	1374	900	389	91	5
	Shear (N)	87	79	71	61	42	36	24	6	1
	Torque (N-M)	46	63	79	85	76	65	53	7	1
OPEN LOOP	*Bending Moment (N-M)*	5545	4262	3116	2125	1115	671	248	47	3
	Shear (N)	98	85	75	61	38	31	18	3	1
	Torque (N-M)	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

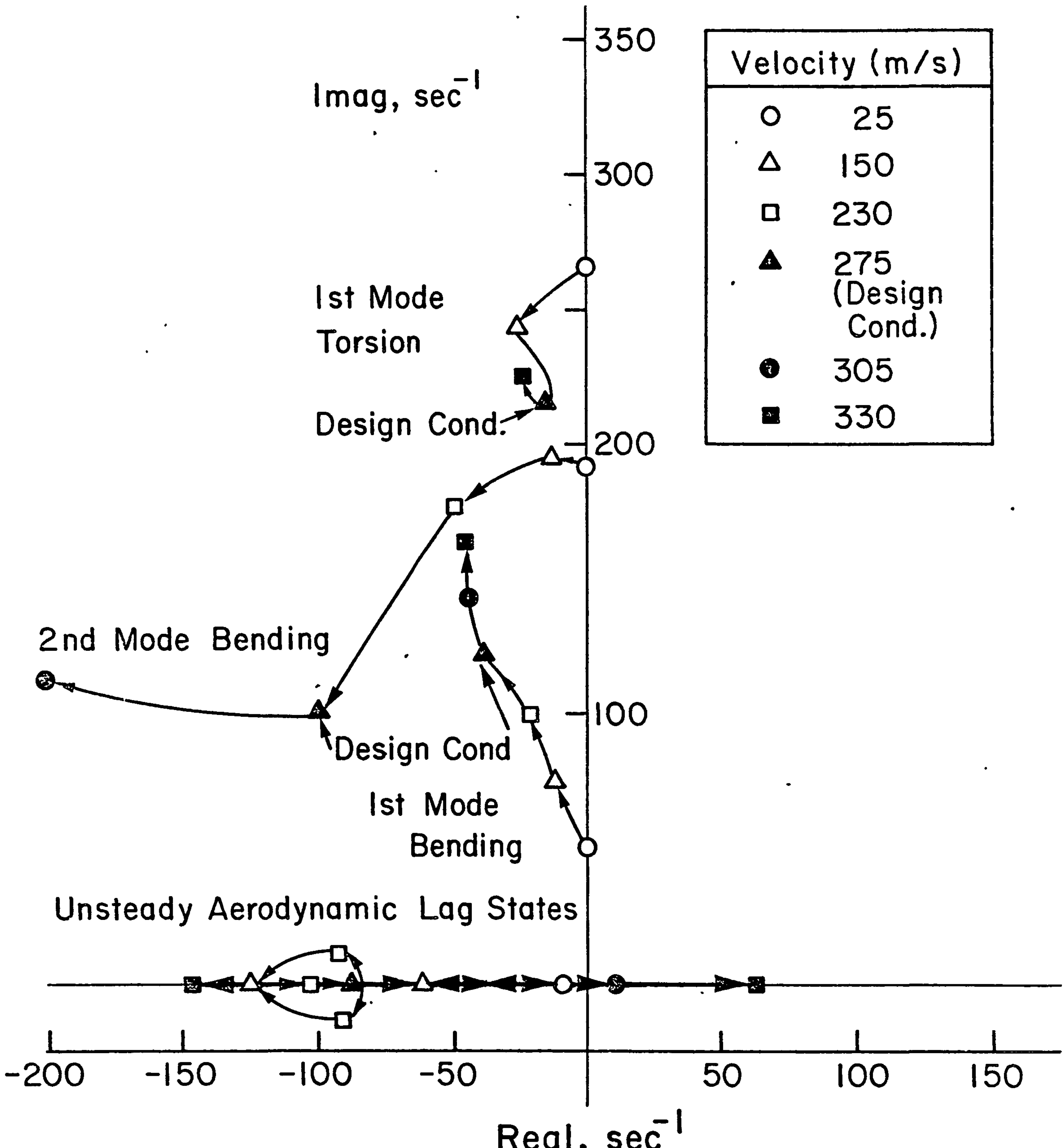


Fig. 5 Root Locus of LQ Controlled Wing as Velocity is Varied

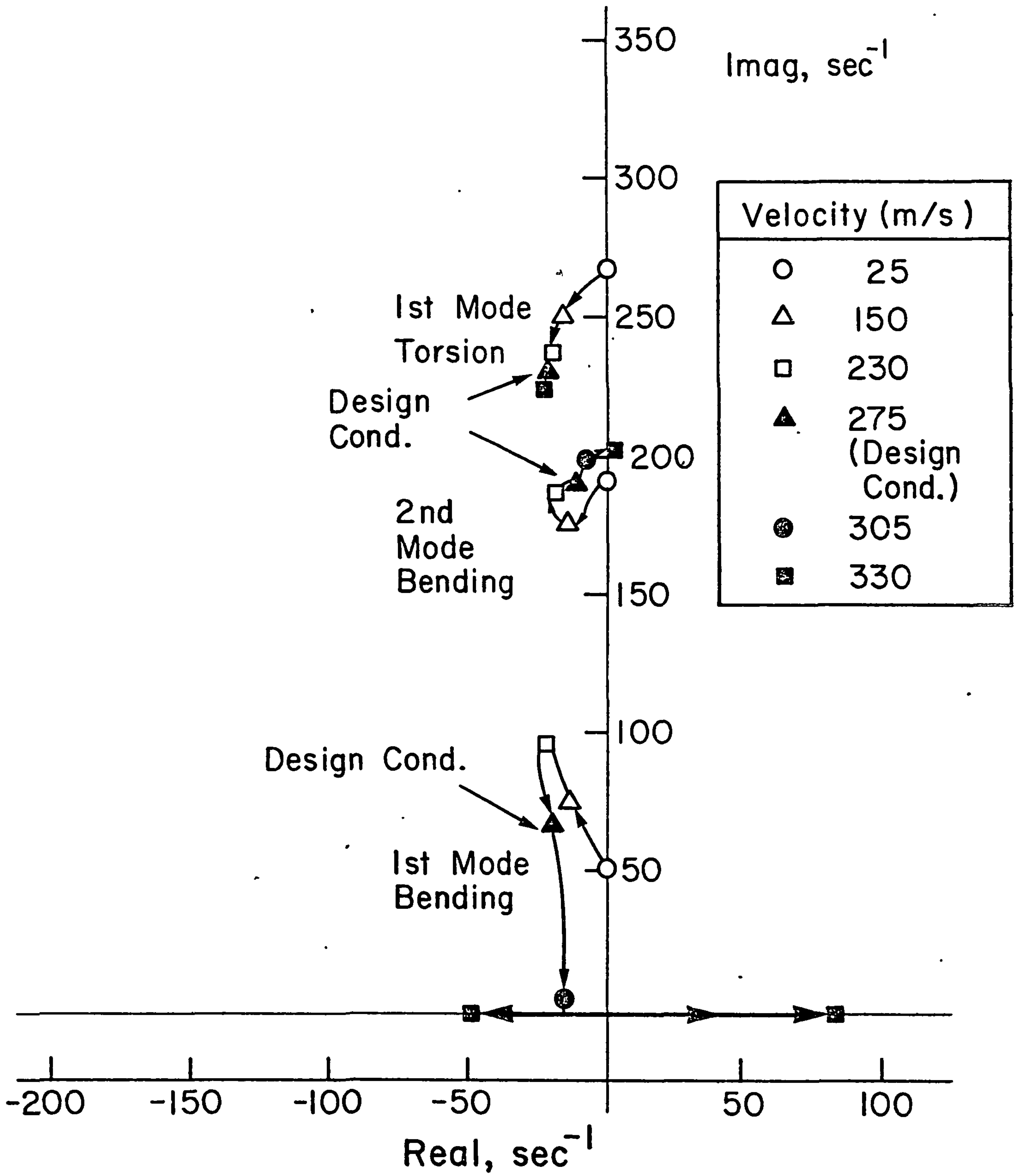


Fig. 6 Root Locus of ES Controlled Wing as Velocity is Varied

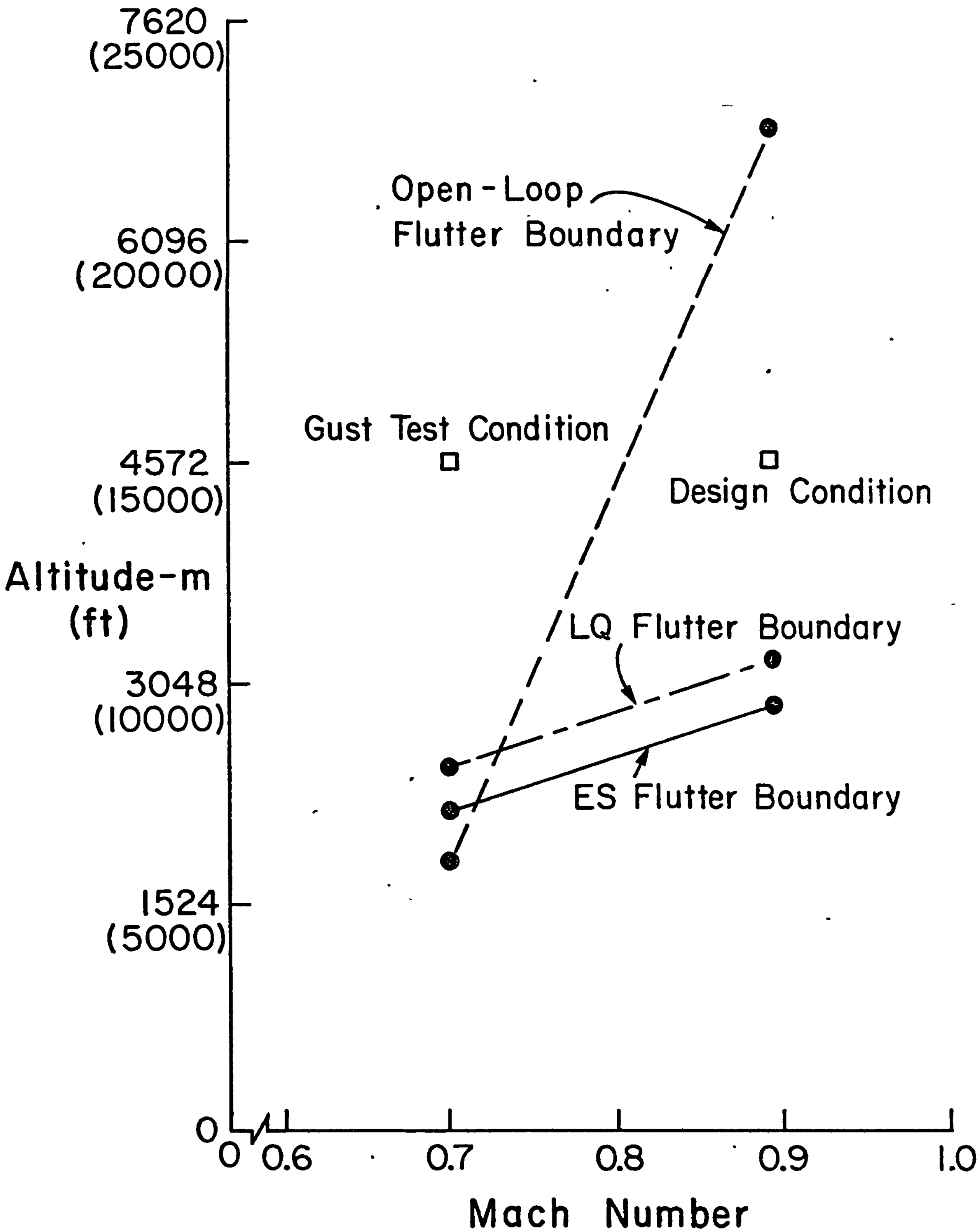


Fig. 7 Flutter Boundaries for Uncontrolled, LQ, and ES Controlled Wing

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

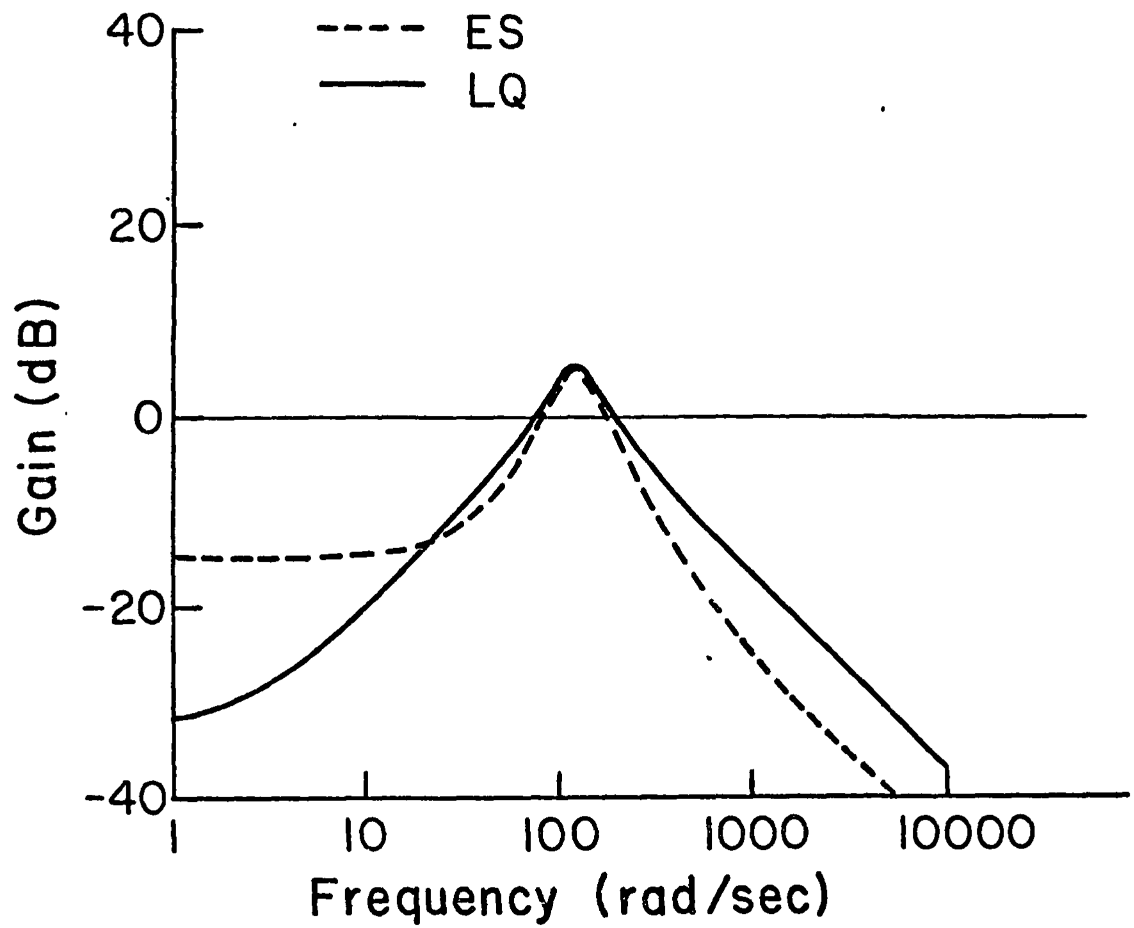


Fig. 8 Frequency Response of Magnitude of $H(s)G(s)$ for LQ and ES Designs

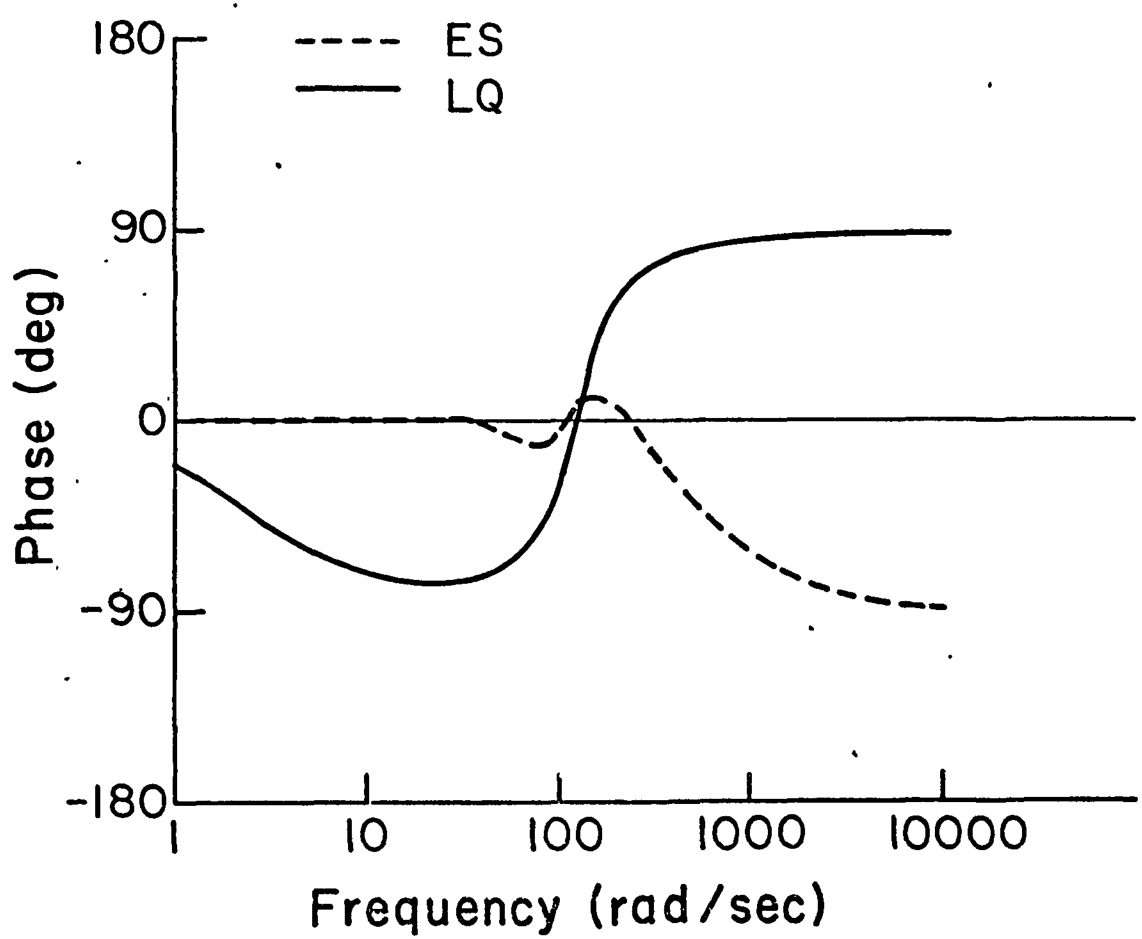


Fig. 9 Frequency Response of Phase of H(s)G(s) for LQ and ES Designs

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

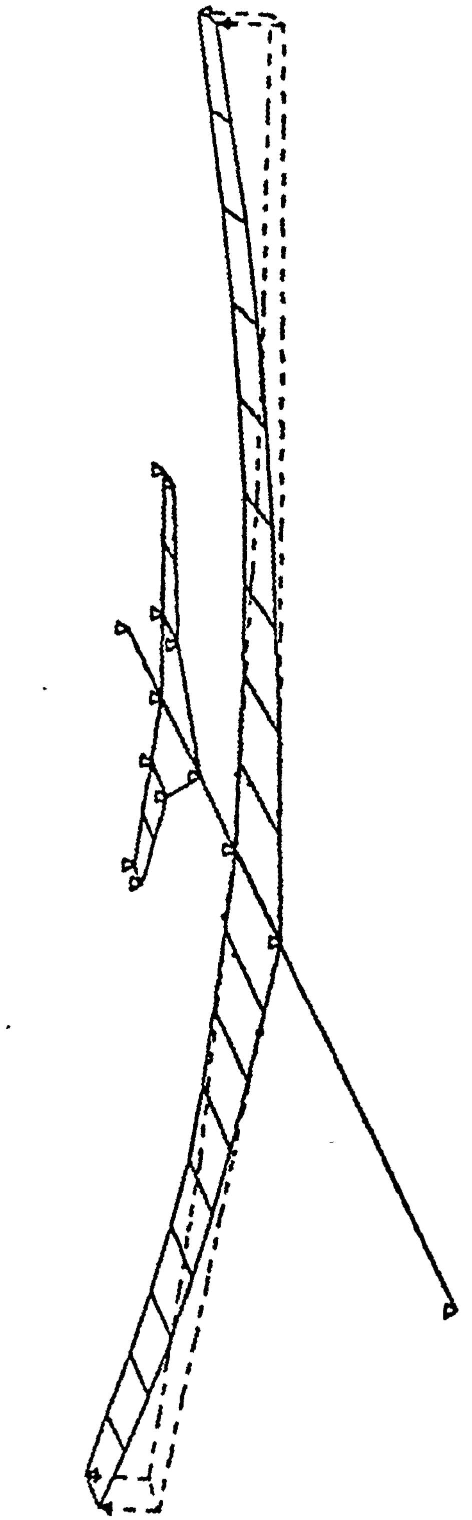
Abstract

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.

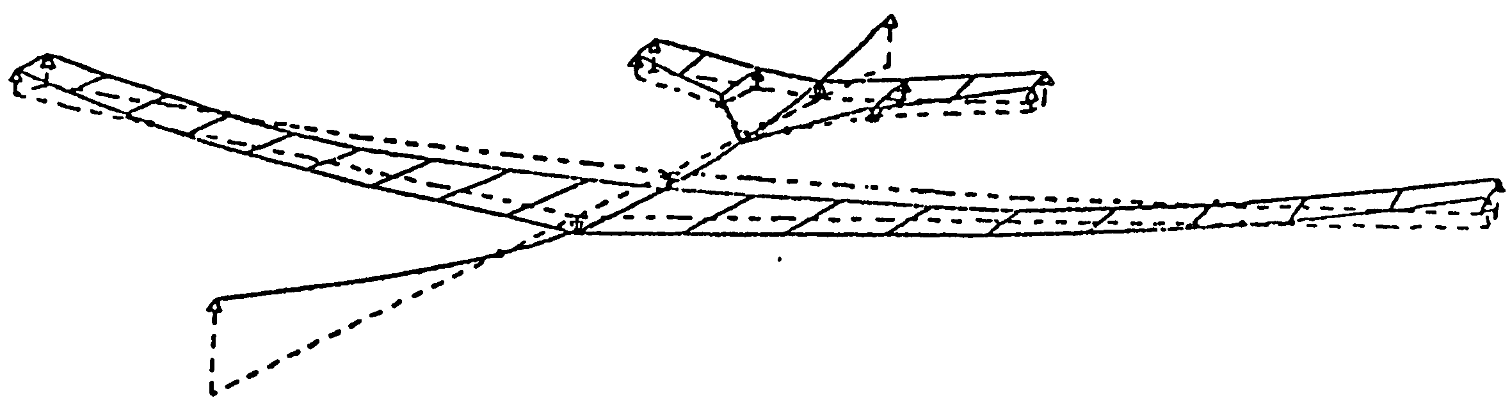
*This paper has been submitted to the AIAA Journal of Guidance, Control, and Dynamics.

Appendix A

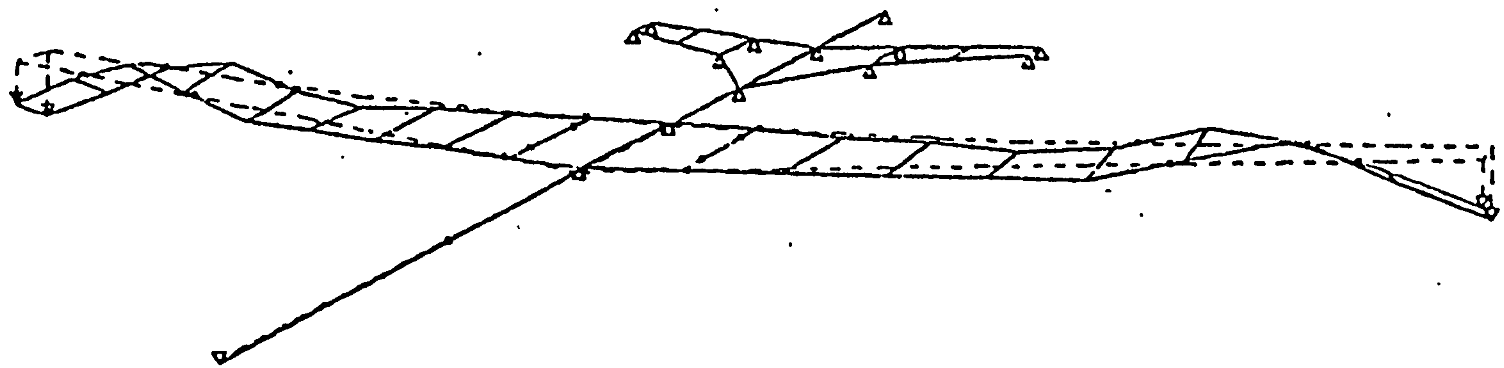
Flexure Mode Shapes



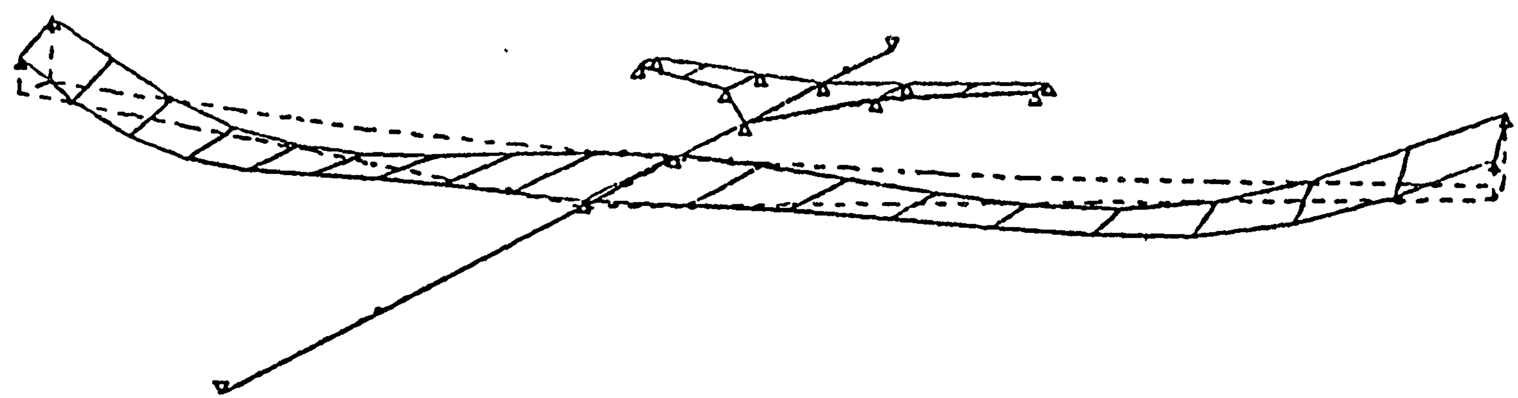
(a) mode 1, frequency = 8.44 hz, generalized mass = 0.0169 lb-sec^2/in



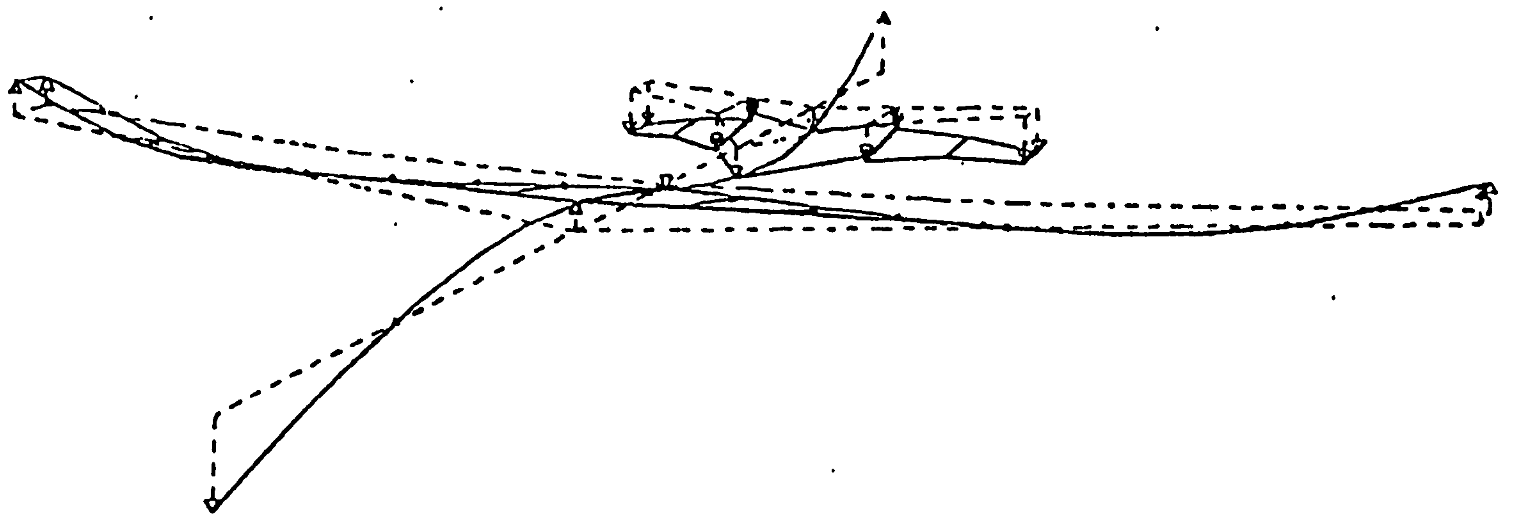
(b) mode 2, frequency = 15.7 hz, generalized mass = 0.326 $lb\text{-}sec^2/in$



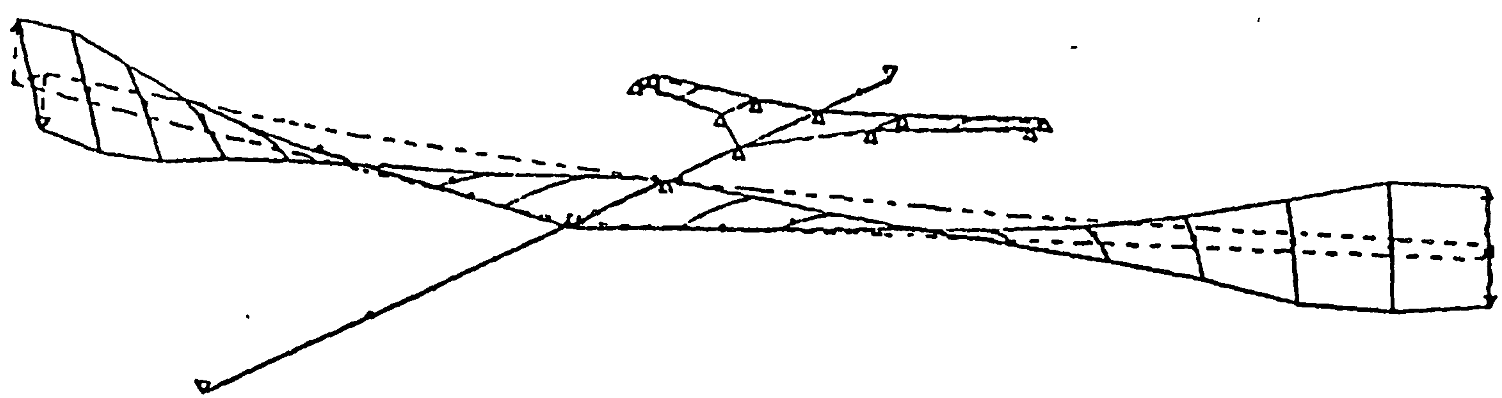
(c) mode 3, frequency = 23.7 hz, generalized mass = 0.0153 $lb\text{-}sec^2/in$



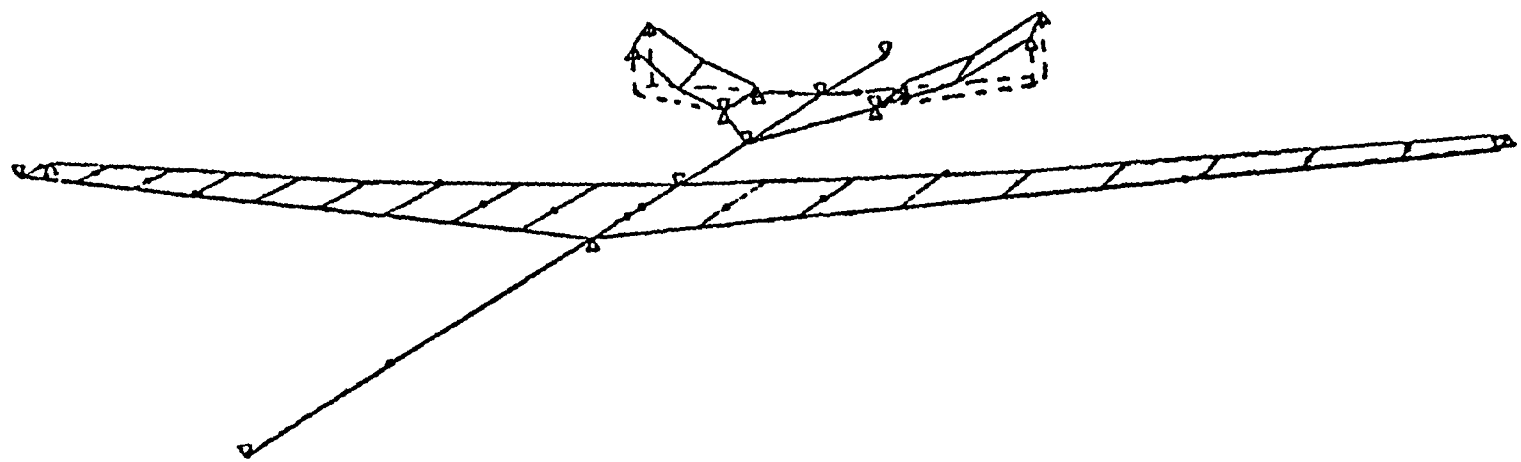
(d) mode 4, frequency = 31.7 hz, generalized mass = 0.0158 lb-sec^2/in



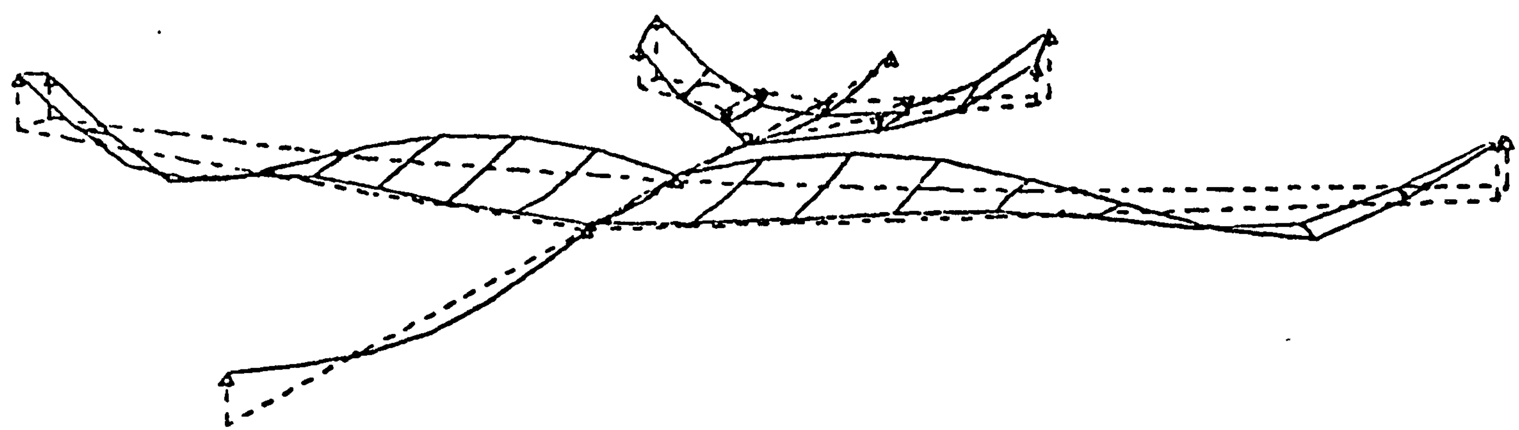
(e) mode 5, frequency = 39.3 hz, generalized mass = 0.178 lb-sec^2/in



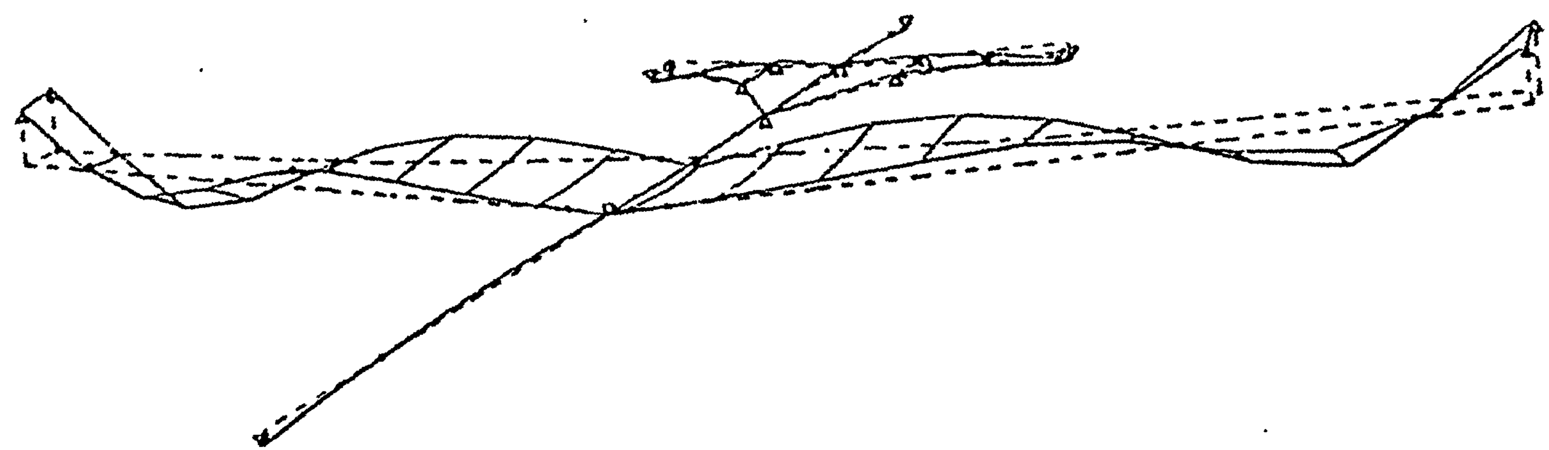
(f) mode 6, frequency = 44.8 hz, generalized mass = 0.593 lb-sec^2/in



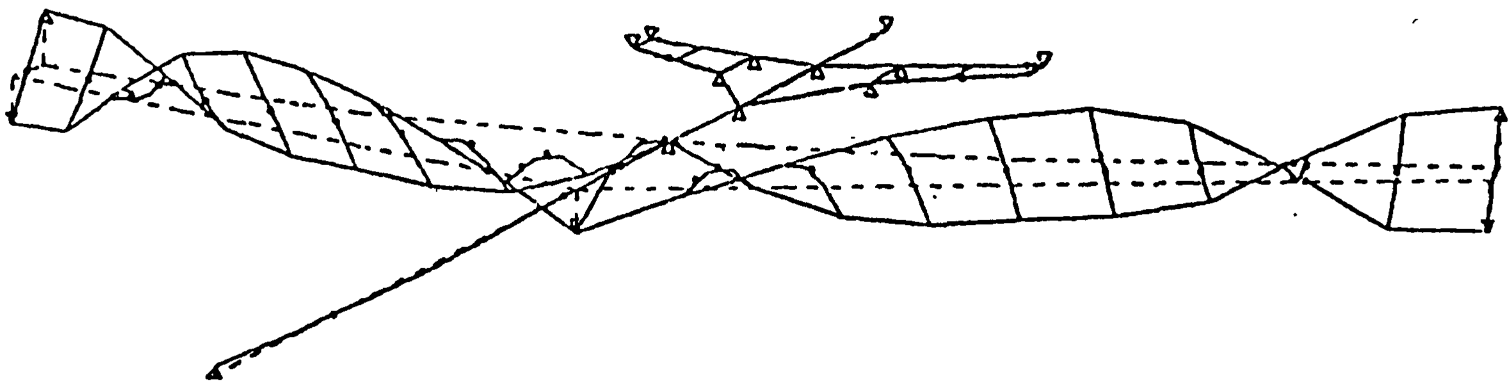
(g) mode 7, frequency = 47.3 hz, generalized mass = 0.00335 lb-sec^2/in



(h) mode 8, frequency = 66.6 hz, generalized mass = 0.0621 $lb\text{-}sec^2/in$



(i) mode 9, frequency = 71.2 hz, generalized mass = 0.0286 $lb\text{-}sec^2/in$



(j) mode 10, frequency = 80.3 hz, generalized mass = 0.696 lb-sec^2/in

Appendix B

Program Descriptions and Listings

FLUTTER

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

$$\dot{x} = Ax + Bu + \Gamma\eta$$

This first order form is derived from the second order form

$$[M_{xx} + \bar{q}\, A_2^x (\frac{\bar{c}}{2V})^2]\ddot{\bar{x}} + [C_s + \bar{q}\, A_1^x (\frac{\bar{c}}{2V})]\dot{\bar{x}} + [K_s + \bar{q}\, A_0^x]\bar{x}$$

$$+ \sum_{i=1}^{L} y_i + [M_{xu} + \bar{q}\, A_2^u (\frac{\bar{c}}{2V})^2]\ddot{\hat{u}} + \bar{q}\, A_1^u (\frac{\bar{c}}{2V})\dot{\hat{u}}$$

$$+ \bar{q}\, A_2^\delta (\frac{\bar{c}}{2V})^2\, \frac{\ddot{\delta}}{V} + \bar{q}\, A_1^\delta (\frac{\bar{c}}{2V})\, \frac{\dot{\delta}}{V} + \bar{q}\, A_0^\delta\, \frac{\delta}{V} = 0$$

and i=1,2,...L aerodynamic lag states

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

or

$$M\ddot{\bar{x}} + C\dot{\bar{x}} + K\bar{x} + \sum_{i=1}^{L} y_i + P\ddot{\hat{u}} + Q\dot{\hat{u}} + R\hat{u} + S\ddot{\delta}$$

$$+ T\dot{\delta} + U\delta = 0$$

and

$$\dot{y}_i = -I(\frac{2V}{\bar{c}})K_i y_i + Di\dot{\bar{x}} + Ei\dot{\hat{u}} + Fi\dot{\delta}$$

where

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

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

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

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

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

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

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

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

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

$$Di = D_i^x$$

$$Ei = D_i^u$$

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

$\bar{x}$ = 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

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

$$\dot{\bar{u}} = G\bar{u} + Hu$$

where

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

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

$$\bar{u} = \begin{Bmatrix} \bar{u}_o \\ \dot{\bar{u}}_o \\ \ddot{\bar{u}}_o \\ \bar{u}_i \\ \dot{\bar{u}}_i \\ \ddot{\bar{u}}_i \\ \dddot{\bar{u}}_i \end{Bmatrix} = \text{control state}$$

$$G = \begin{bmatrix} 0 & 1.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 & 0 & 0 & 0 \\ -1.774\times10^{7} & -1.438\times10^{5} & -431.0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1.0 \\ 0 & 0 & 0 & -1.614\times10^{11} & -5.484\times10^{8} & -1.152\times10^{6} & -933.0 \end{bmatrix}$$

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

$$H = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1.774\times10^{7} & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1.614\times10^{11} \end{bmatrix}$$

And the gust model is

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

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

where z = an intermediate gust state

ℓ = characteristic gust length

η = white driving noise

Thus, if

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

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

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

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

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

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
MNETS PROGRAM FLUTTER

```
      PROGRAM FLUTTER(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,ONELE7,
+TAPE2=ONELE7,MASS,TAPE1=MASS,TAPE3)
C**************************************************************
C     THIS PROGRAM INPUTS THE STRUCTURAL MASS,STRUCTURAL
C     DAMPING, STRUCTURAL STIFFNESS, AERODYNAMIC INFLUENCE
C     MATRICES, FLIGHT CONDITIONS AND OUTPUTS THE A, B, AND W
C     MATRICES OF THE LINEAR STATE EQUATION
C           DX/DT = AX + BU + WN
C     FOR THE INBOARD AND OUTBOARD AILERONS AS CONTROLS.
C     OPEN LOOP EIGENVALUES AND EIGENVECTORS CAN BE OUTPUT
C     AS WELL.
C**************************************************************
      REAL MXX(8,8),CS(8,8),KS(8,8),CT(8,8),KT(8,8),CC(8,8),
+KK(8,8),A2X(8,8),A1X(8,8),A0X(8,8),MINV(8,8),A2XQ(8,8)
+,A1XQ(8,8),A0XQ(8,8),A1(16,16),A2(32,32),DQ(8,2),DUM1(8,7),
+DUM2(8,7),DUM3(8,2),DUM4(8,7),DUM5(8,2),DUM6(8,2),DUM7(8,1)
      REAL DUM8(8,1),A0U(8,2),A1U(8,2),EI(8,2,4),R(8,2),
+EIIG(8,7,4),FI(8,4),A3(40,40),B3(40,2),B2(32,2),II(8,8),
+DT(8,8,4),KI(4),A1D(8,1),A0D(8,1),T(8,1),U(8,1),A4(42,42),
+B4(42,2),W(42,1),B1(16,2),A5(24,24),B5(24,2)
      WRITE(6,8)
    8 FORMAT(1X,'INPUT THE ORDER OF THE STRUCTURAL-RIGID BODY STATE VECTO
+R(N)'/)
      READ(5,*)N
      WRITE(6,10)
   10 FORMAT(1X,'INPUT VELOCITY(V), CHORD(C), DYNAMIC PRESSURE(Q), NUMBE
+R OF LAG STATES(L)'/)
C
C     INPUT FLIGHT CONDITIONS
C
      READ(5,*)V,C,Q,L
      DO 20 I=1,L
      WRITE(6,25)I
C
C     INPUT AERODYNAMIC LAG FREQUENCIES
C
      READ(5,*)KI(I)
   20 CONTINUE
   25 FORMAT(1X,'INPUT KI(',I1,')'/)
      N2=2*N
      KEND=N*(L+2)
      KENDD=KEND+7
      KEND2=KENDD+2
      K5=2*N+7
```

```
+,DUM5,DUM6,DUM7,DUM8,AOU,A1U,FT,R,FTJG,TT,A3,B3,B2,FT,DT,KI,AID,
+AOD,T,U,A4,B1,B4,W,A5,B5)
      STOP
      END
C
C
      SUBROUTINE PROG1(N,N2,KEND,KENDD,KEND2,K5,V,C,Q,L,MXX,CS,KS,CT,KT,
+CC,KK,A2X,A1X,A0X,MINV,A2XQ,A1XQ,A0XQ,A1,A2,QQ,DUM1,DUM2,DUM3,DUM4
+,DUM5,DUM6,DUM7,DUM8,AOU,A1U,ET,R,ETJG,II,A3,B3,B2,FT,DI,KI,AID,
+AOD,T,U,A4,B1,B4,W,A5,B5)
      REAL LAMDA,LL
      INTEGER FLAG
      COMMON KRET(42)
      REAL MXX(N,N),CS(N,N),KS(N,N),CT(N,N),KT(N,N),CC(N,N),KK(N,N),
+A2X(N,N),A1X(N,N),A0X(N,N),MINV(N,N),A2XQ(N,N),A1XQ(N,N),A0XQ(N,N)
+,A1(N2,N2),A2(KEND,KEND),QQ(N,2),DUM1(N,7),DUM2(N,7),DUM3(N,2),
+DUM4(N,7),DUM5(N,2),DUM6(N,2),DUM7(N,1),DUM8(N,1),AOU(N,2)
      REAL A1U(N,2),
+EI(N,2,L),G(7,7),H(7,2),R(N,2),JG(2,7),EIJG(N,7,L),FI(N,L),
+JH(2,2),A3(KENDD,KENDD),B3(KENDD,2),B2(KEND,2),II(N,N),DI(N,N,L)
      REAL KI(L),A1D(N,1),AOD(N,1),T(N,1),U(N,1),A4(KEND2,KEND2),
+B1(N2,2),B4(KEND2,2),W(KEND2,1),
+A5(K5,K5),B5(K5,2),JJ(2,7)
      REAL AA(42,42),BB(42,2)
      CALL GETPF(5HTAPE1,4HMASS,0,0)
C
C     READ IN STRUCTURAL DATA
C
    2 FORMAT(E16.8)
      DO 1 I=1,N
      DO 1 J=1,N
      READ(1,101)MXX(I,J),CS(I,J),KS(I,J)
    1 CONTINUE
      DO 3 I=1,7
      DO 3 J=1,7
C
C     READ IN ACTUATOR DATA
C
      READ(1,101)G(I,J)
    3 CONTINUE
      DO 4 I=1,7
      READ(1,101)(H(I,J),J=1,2)
    4 CONTINUE
      DO 5 J=1,7
      READ(1,101)(JJ(I,J),I=1,2)
    5 CONTINUE
      CALL GETPF(5HTAPE2,6HONELE7,0,0)
C
C     READ IN AERODYNAMIC INFLUENCE DATA
```

```
      DO 430 K=1,[illegible]
      DO 410 J=1,N
      DO 410 I=1,N
      READ(2,109)A0X(I,J),A1X(I,J),A2X(I,J),DI(I,J,K)
  410 CONTINUE
      DO 420 J=1,2
      DO 420 I=1,N
      READ(2,101)A0U(I,J),A1U(I,J),EI(I,J,K)
  420 CONTINUE
      DO 430 I=1,N
      READ(2,101)A0D(I,K),A1D(I,K),FI(I,K)
      FI(I,K)=FI(I,K)/V
  430 CONTINUE
  101 FORMAT(3E16.8)
  109 FORMAT(4E16.8)
      EPS=1.E-10
      TEMP=Q*(C/2./V)**2
      TEMP2=Q*C/2./V
      CALL SCAMAT(TEMP,A2X,A2XQ,N,N)
      CALL MATADD(MXX,A2XQ,MINV,N,N)
C
C     FIND INVERSE(MINV)
C
      CALL MXLNEQ(MINV,N,N,DET,JRANK,EPS,A2XQ,0)
      CALL SCAMAT(TEMP2,A1X,A1XQ,N,N)
      CALL MATADD(CS,A1XQ,CT,N,N)
      CALL SCAMAT(Q,A0X,A0XQ,N,N)
      CALL MATADD(KS,A0XQ,KT,N,N)
      CALL SCAMAT(-1.,MINV,MINV,N,N)
      CALL MATMUL(MINV,KT,KK,N,N,N)
      CALL MATMUL(MINV,CT,CC,N,N,N)
      DO 15 I=1,N
      DO 14 J=1,N
      II(I,J)=0.
   14 CONTINUE
      II(I,I)=1.0
   15 CONTINUE
      N21=N2+1
      DO 18 I=1,N
      DO 18 J=1,N
      A1(I,J)=0.
      A1(I,J+N)=II(I,J)
      A1(I+N,J)=KK(I,J)
      A1(I+N,J+N)=CC(I,J)
   18 CONTINUE
      IF(L.EQ.0)GO TO 100
      DO 20 I=1,KEND
      DO 20 J=1,KEND
      A2(I,J)=0.
   20 CONTINUE
```

```
      A2(I,J)=A1(I,J)
      A2(I+N,J)=A1(I+N,J)
      A2(I,J+N)=A1(I,J+N)
      A2(I+N,J+N)=A1(I+N,J+N)
      DO 25 K=1,L
      KN=K+N
      A2(I+N,KN+N+J)=MINV(I,J)
      A2(KN+N+I,J+N)=DI(I,J,K)
      A2(KN+N+I,KN+N+I)=-2.*V*KI(K)/C
   25 CONTINUE
      WRITE(6,27)
   27 FORMAT(1X,'SHOULD THE CONTROL MATRIX B ALSO BE CALCULATED? YES(1),
     + NO(0)'/)
      READ(5,28)MM
   28 FORMAT(I1)
      IF(MM.EQ.1)GO TO 30
      FLAG=2
      GO TO 200
   30 SCA=Q+C/2./V
      CALL SCAMAT(SCA,A1U,QQ,N,2)
      CALL SCAMAT(Q,A0U,R,N,2)
      DO 36 I=1,KENDD
      DO 36 J=1,KENDD
      A3(I,J)=0.
   36 CONTINUE
      DO 37 I=1,KEND
      DO 37 J=1,KEND
      A3(I,J)=A2(I,J)
   37 CONTINUE
      DO 38 I=1,7
      DO 38 J=1,7
      A3(KEND+I,KEND+J)=G(I,J)
   38 CONTINUE
      CALL MATMUL(JJ,G,JG,2,7,7)
      CALL MATMUL(QQ,JG,DUM1,N,2,7)
      CALL MATMUL(R,JJ,DUM2,N,2,7)
      CALL MATADD(DUM2,DUM1,DUM2,N,7)
      CALL MATMUL(MINV,DUM2,DUM1,N,N,7)
      DO 40 K=1,L
      DO 39 I=1,N
      DO 39 J=1,2
      DUM3(I,J)=EI(I,J,K)
   39 CONTINUE
      CALL MATMUL(DUM3,JG,DUM4,N,2,7)
      DO 40 I=1,N
      DO 40 J=1,7
      EIJG(I,J,K)=DUM4(I,J)
   40 CONTINUE
      DO 45 I=1,N
```

```
      A3(KN,J+KEND)=DUM(I,J)
      DO 45 K=1,L
      KN=K*N
      A3(KN+N+I,J+KEND)=EIJG(I,J,K)
   45 CONTINUE
      CALL MATMUL(JJ,H,JH,2,2,2)
      CALL MATMUL(DQ,JH,DUM3,N,2,2)
      CALL MATMUL(MINV,DUM3,DUM5,N,N,2)
      DO 47 K=1,L
      DO 46 I=1,N
      DO 46 J=1,2
      DUM3(I,J)=EI(I,J,K)
   46 CONTINUE
      CALL MATMUL(DUM3,JH,DUM6,N,2,2)
      DO 47 I=1,N
      DO 47 J=1,2
      EIJG(I,J,K)=DUM6(I,J)
   47 CONTINUE
      DO 48 I=1,N
      DO 48 J=1,2
      B3(I,J)=0.
      B3(I+N,J)=DUM5(I,J)
      DO 48 K=1,L
      KN=K*N
      B3(I+KN+N,J)=EIJG(I,J,K)
   48 CONTINUE
      DO 49 I=1,7
      DO 49 J=1,2
      B3(I+KEND,J)=H(I,J)
   49 CONTINUE
      WRITE(6,50)
   50 FORMAT(1X,'SHOULD THE NOISE VECTOR (W) ALSO BE CALCULATED? YES(1),
+NO(0)'/)
      READ(5,28)MM
      IF(MM.EQ.1)GO TO 55
      FLAG=3
      GO TO 200
   55 WRITE(6,56)
   56 FORMAT(1X,'INPUT CHARACTERISTIC LENGTH(LL)'/)
      READ(5,*)LL
      TEMP3=TEMP2/V
      CALL SCAMAT(TEMP3,AID,T,N,1)
      Q1=Q/V
      CALL SCAMAT(Q1,AOD,U,N,1)
      CALL MATMUL(MINV,U,DUM7,N,N,1)
      CALL MATMUL(MINV,T,DUM8,N,N,1)
      DO 58 I=1,KENDD
      DO 58 J=1,2
      B4(I,J)=B3(I,J)
      B4(KENDD+1,J)=0.
```

```
      DO 59 I=1,KEND2
      DO 59 J=1,KEND2
      A4(I,J)=0.
   59 CONTINUE
      DO 60 I=1,KENDD
      DO 60 J=1,KENDD
      A4(I,J)=A3(I,J)
   60 CONTINUE
      A4(KENDD+1,KENDD+2)=1.
      A4(KENDD+2,KENDD+1)=-1.001*(V/LL)**2
      A4(KENDD+2,KENDD+2)=-2.001*V/LL
      DO 62 I=1,N
      A4(I+N,KENDD+1)=DUM7(I,1)
      A4(I+N,KENDD+2)=DUM8(I,1)
      DO 62 K=1,L
      KN=K*N
      A4(I+KN+N,KENDD+2)=FI(I,K)
   62 CONTINUE
      DO 64 I=1,KEND2
      W(I,1)=0.
   64 CONTINUE
      XX=1.732*V/LL
      W(KENDD+1,1)=XX
      W(KENDD+2,1)=-2.464*(V/LL)**2
      CALL SCAMAT(XX,DUM8,DUM8,N,1)
      DO 66 I=1,N
      W(I+N,1)=DUM8(I,1)
      DO 66 K=1,L
      KN=K*N
      W(I+N+KN,1)=XX*FI(I,K)
   66 CONTINUE
      FLAG=4
      GO TO 200
  100 WRITE(6,27)
      READ(5,28)MM
      IF(MM.EQ.1)GO TO 102
      FLAG=1
      GO TO 200
  102 SCA=Q*C/2./V
      CALL SCAMAT(SCA,A1U,QQ,N,2)
      CALL SCAMAT(Q,A0U,R,N,2)
      DO 106 I=1,K5
      DO 106 J=1,K5
      A5(I,J)=0.
  106 CONTINUE
      CALL MATMUL(JJ,H,JH,2,7,2)
      CALL MATMUL(QQ,JH,DUM3,N,2,2)
      CALL MATMUL(MINV,DUM3,DUM5,N,N,2)
      DO 108 I=1,N
```

```
      B5(I+N,J)=DUM5(I,J)
  108 CONTINUE
      DO 110 I=1,7
      DO 110 J=1,2
      B5(I+N2,J)=H(I,J)
  110 CONTINUE
      DO 112 I=1,N2
      DO 112 J=1,N2
      A5(I,J)=A1(I,J)
  112 CONTINUE
      CALL MATMUL(JJ,G,JG,2,7,7)
      CALL MATMUL(QQ,JG,DUM1,N,2,7)
      CALL MATMUL(R,JJ,DUM2,N,2,7)
      CALL MATADD(DUM2,DUM1,DUM2,N,7)
      CALL MATMUL(MINV,DUM2,DUM1,N,N,7)
      DO 114 I=1,N
      DO 114 J=1,7
      A5(I+N,J+N2)=DUM1(I,J)
  114 CONTINUE
      DO 116 I=1,7
      DO 116 J=1,7
      A5(I+N2,J+N2)=G(I,J)
  116 CONTINUE
      FLAG=5
  200 WRITE(6,201)
  201 FORMAT(1X,'INPUT RETAIN(1) OR DELETE(0) FOR EACH STATE MEMBER'/)
      KSTATE=N
      KSUM=0
      READ(5,*)(KRET(J),J=1,KSTATE)
      DO 205 I=1,KSTATE
      KRET(I+N)=KRET(I)
      IF(KRET(I).EQ.1)KSUM=KSUM+1
  205 CONTINUE
      IF(L.EQ.0)GO TO 207
      DO 206 K=1,L
      DO 206 I=1,N
      KN=N*(K+1)+I
      KRET(KN)=KRET(I)
  206 CONTINUE
      DO 209 I=1,9
      KRET(KEND+I)=1
  209 CONTINUE
  207 GO TO(310,320,330,340,350),FLAG
  310 KSUM=2*KSUM
      CALL PROG2(FLAG,KSUM,A1,B1,N2,AA,BB)
      GO TO 900
  320 KSUM=(L+2)*KSUM
      CALL PROG2(FLAG,KSUM,A2,B2,KEND,AA,BB)
      GO TO 900
```

```
      GO TO 900
  340 KSUM=9+(I+2)*KSUM
      DO 341 I=1,KEND2
      IF(KRET(I).EQ.0)GO TO 341
      WRITE(3,2)W(I,1)
  341 CONTINUE
      CALL PROG2(FLAG,KSUM,A4,B4,KEND2,AA,BB)
      GO TO 900
  350 KSUM=7+2*KSUM
      DO 351 I=N21,K5
      KRET(I)=1
  351 CONTINUE
      CALL PROG2(FLAG,KSUM,A5,B5,K5,AA,BB)
  900 RETURN
      END
C
C
      SUBROUTINE MATDEL(A,N,AA,KSUM)
C
C     THIS ROUTINE DELETES DESIRED STATES FROM THE A MATRIX
C
      DIMENSION A(N,N),AA(KSUM,KSUM),JRET(40)
      COMMON KRET(42)
      J=0
      DO 100 I=1,N
      IF(KRET(I).EQ.0)GO TO 100
      J=J+1
      JRET(J)=I
  100 CONTINUE
      DO 200 I=1,KSUM
      DO 200 J=1,KSUM
      AA(I,J)=A(JRET(I),JRET(J))
  200 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE MATDELB(B,N,BB,KSUM)
C
C     THIS ROUTINE DELETES DESIRED STATES FROM THE B MATRIX
C
      DIMENSION B(N,2),BB(KSUM,2),JRET(40)
      COMMON KRET(42)
      J=0
      DO 100 I=1,N
      IF(KRET(I).EQ.0)GO TO 100
      J=J+1
      JRET(J)=I
  100 CONTINUE
```

"PAGE MISSING FROM AVAILABLE VERSION"

PAGE 73

```
      BB(I,J)=B(IRET(I),J)
  200 CONTINUE
      RETURN
      END
C
C
C.....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
C
C.....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
C
C
C.....B(M,N)=S * A(M,N)
      SUBROUTINE SCAMAT(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
C
C
C.....V(N,M)= A(M,N) TRANSPOSED
      SUBROUTINE MATTRA(A,V,M,N)
      DIMENSION A(M,N),V(N,M)
      DO 14 I=1,M
```

```
      V(J,I)=A(I,J)
   14 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE PROG2(FLAG,KSUM,A,B,IP,AA,BB)
      INTEGER FLAG,IV1(42)
      REAL AA(KSUM,KSUM),BB(KSUM,2),A(IP,IP),B(IP,2),FV1(42)
      REAL WR(42),WI(42),Z(42,42)
      GO TO(310,320,330,340,350),FLAG
  100 FORMAT(3E16.8)
  310 CALL MATDEL(A,IP,AA,KSUM)
      DO 311 I=1,KSUM
      DO 311 J=1,KSUM
      WRITE(3,100)AA(I,J)
  311 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1)
      GO TO 900
  320 CALL MATDEL(A,IP,AA,KSUM)
      DO 321 I=1,KSUM
      DO 321 J=1,KSUM
      WRITE(3,100)AA(I,J)
  321 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1)
      GO TO 900
  330 CALL MATDEL(A,IP,AA,KSUM)
      CALL MATDELB(B,IP,BB,KSUM)
      DO 331 I=1,KSUM
      DO 331 J=1,KSUM
      WRITE(3,100)AA(I,J)
  331 CONTINUE
      DO 332 I=1,KSUM
      WRITE(3,100)BB(I,1),BB(I,2)
  332 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1)
      GO TO 900
  340 CALL MATDEL(A,IP,AA,KSUM)
      CALL MATDELB(B,IP,BB,KSUM)
      DO 341 I=1,KSUM
      DO 341 J=1,KSUM
      WRITF(3,100)AA(I,J)
  341 CONTINUE
      DO 342 I=1,KSUM
      WRITE(3,100)BB(I,1),BB(I,2)
  342 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1)
      GO TO 900
  350 CALL MATDEL(A,IP,AA,KSUM)
      CALL MATDELB(B,IP,BB,KSUM)
      DO 351 I=1,KSUM
      DO 351 J=1,KSUM
```

```
      351 CONTINUE
          DO 352 I=1,KSUM
          WRITE(3,100)BB(I,1),BB(I,2)
      352 CONTINUE
          CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1)
      900 CALL REPLACE(5HTAPE3,5HFILE9,0,0)
          RETURN
          END
C
C
          SUBROUTINE PROG3(N,AA,WR,WI,Z,IV1,FV1)
          COMMON KRET(42)
          DIMENSION AA(N,N),WR(N),WI(N),Z(N,N),IV1(N),FV1(N)
C
C         CALCULATE EIGENVALUES AND EIGENVECTORS OF AA
C
          CALL RG(N,N,AA,WR,WI,1,Z,IV1,FV1,IERR)
          WRITE(6,23)
       23 FORMAT(41X,'OPEN LOOP EIGENVALUES'/31X,'REAL PART',20X,
+'IMAGINARY PART')
          DO 30 I=1,N
          WRITE(6,25)I,WR(I),WI(I)
       25 FORMAT(20X,I2,'.',2X,E15.4,15X,E15.4)
       30 CONTINUE
          WRITE(6,55)
       55 FORMAT(1X,'DO YOU WANT THE OPEN LOOP EIGENVECTORS OUTPUT? YES(1)
+,NO(0)')
          READ(5,*)MM
          IF(MM.EQ.0)GO TO 60
          WRITE(6,64)
       64 FORMAT(1X,'INPUT PRINT(1) OR SUPPRESS(0) FOR EACH EIGENVECTOR')
          READ(5,*)(KRET(J),J=1,N)
          WRITE(6,34)
       34 FORMAT(41X,'OPEN LOOP EIGENVECTORS'/31X,'REAL PART',20X,
+'IMAGINARY PART')
          IFLAG=0
          DO 50 J=1,N
          IF(KRET(J).EQ.0)GO TO 49
          IF(WI(J).NE.0.0)GO TO 35
          DO 37 I=1,N
          IF(I.NE.1)GO TO 36
          WRITE(6,38)J,Z(1,J)
       38 FORMAT(//20X,I2,'.',2X,E15.4,20X,'0.0')
          GO TO 37
       36 WRITE(6,39)Z(I,J)
       39 FORMAT(25X,E15.4,20X,'0.0 )
       37 CONTINUE
          GO TO 50
       35 IF(IFLAG.EQ.1)GO TO 40
          IFLAG=1
```

```
      WRITE(6,43)J,Z(I,J),Z(I,J+1)
   43 FORMAT(//20X,I2,'.',2X,E15.4,15X,E15.4)
      GO TO 41
   42 WRITE(6,44)Z(I,J),Z(I,J+1)
   44 FORMAT(25X,E15.4,15X,E15.4)
   41 CONTINUE
      GO TO 50
   40 J1=J-1
      WRITE(6,45)J,J1
   45 FORMAT(//20X,I2, .',5X,'CONJUGATE OF EIGENVECTOR',1X,I2,'.')
   49 IFLAG=0
   50 CONTINUE
   60 RETURN
      END
```

82/12/15
MNETS   PROGRAM   MODAL

```
      PROGRAM MODAL(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,FTLE3,
+TAPE1=FTLE3,TAPE3)
C**************************************************************
C     THIS PROGRAM INPUTS THE STATE MATRICES(OUTPUT
C     OF FLUTTER) A, B, W, DESTRFD CLOSED
C     LOOP EIGENVALUES AND EIGENVECTORS, THE EIGENVECTOR
C     WEIGHTING MATRIX AND OUTPUTS THE GAIN MATRIX K WHERE
C                 U = KX
C     THE CLOSED LOOP EIGENVALUES AND EIGENVECTORS THAT WERE
C     ACHTEVED CAN BE OUTPUT AS WELL.
C**************************************************************
      DTMENSTON A(42,42),WR(2,42),WI(2,42),Z(42,42),IV1(42),
+FV1(42),VECRD(42,42),VECID(42,42),ERD(42),ETD(42),KRET(2,42),W(42,1),
+B(42,2)
      WRITE(6,10)
   10 FORMAT(1X,'INPUT ORDER OF SYSTEM')
      READ(5,*)N
      CALL PROG1(N,A,WR,WI,Z,IV1,FV1,VECRD,VECID,ERD,EID,KRET,
+W,B)
      STOP
      END
C
C
      SUBROUTINE PROG1(N,A,WR,WI,Z,IV1,FV1,VECRD,VECID,ERD,EID,
+KRET,W,B)
      DIMENSION A(N,N),WR(2,N),WI(2,N),Z(N,N),IV1(N),FV1(N),
+VECRD(N,N),VECID(N,N),ERD(N),EID(N),KRET(2,N),W(N,1),B(N,2)
      REAL II(42,42),LB(84,4),LBT(4,84)
      DTMENSTON AB(84,84),VB(84,1),YB(84,84),PP(42,2),DV(42,1),
+BB(84,4),C(42,42),P(84,1)
      N2=N*2
      NM1=N-1
      NM2=N-2
      WRITE(6,11)
   11 FORMAT(1X,'IS THE NOISE VECTOR (W) BEING INPUT? YES(1), NO(0)')
      READ(5,12)MM
   12 FORMAT(I1)
C
C     READ IN STATE MATRICES OUTPUT FROM PROGRAM: FLUTTER
C
      CALL GETPF(5HTAPE1,5HFILE3,0,0)
      IF(MM.EQ.0)GO TO 14
      DO 13 I=1,N
      READ(1,15)W(I,1)
```

```
   14 DO 16 I=1,N
      DO 16 J=1,N
      READ(1,15)A(I,J)
   16 CONTINUE
      DO 17 I=1,N
      READ(1,15)B(I,1),B(I,2)
   17 CONTINUE
      DO 24 I=1,N
      DO 24 J=1,N
      VECRD(I,J)=A(I,J)
   24 CONTINUE
C
C     CALCULATE OPEN LOOP EIGENVALUES AND EIGENVECTORS
C
      CALL RG(N,N,VECRD,ERD,EID,1,Z,IV1,FV1,IERR)
      WRITE(6,23)
   23 FORMAT(41X,'OPEN LOOP EIGENVALUES'/31X,'REAL PART',20X,
+'IMAGINARY PART')
      DO 30 I=1,N
      WRITE(6,25)I,ERD(I),EID(I)
   25 FORMAT(20X,I2,'.',2X,E15.4,15X,E15.4)
   30 CONTINUE
      IFLAG=0
      DO 50 J=1,N
      IF(EID(J).NE.0.0)GO TO 35
      KRET(1,J)=0
      DO 37 I=1,N
      IF(I.NE.1)GO TO 36
      VECRD(1,J)=Z(1,J)
      VECID(1,J)=0.0
      GO TO 37
   36 VECRD(I,J)=Z(I,J)
      VECID(I,J)=0.0
   37 CONTINUE
      GO TO 50
   35 KRET(1,J)=IFLAG
      IF(IFLAG.EQ.1)GO TO 40
      IFLAG=1
      DO 41 I=1,N
      IF(I.NE.1)GO TO 42
      VECRD(1,J)=Z(1,J)
      VECID(1,J)=Z(1,J+1)
      GO TO 41
   42 VECRD(I,J)=Z(I,J)
      VECID(I,J)=Z(I,J+1)
   41 CONTINUE
      GO TO 50
   40 J1=J-1
      DO 46 I=1,N
```

```
      [illegible]
   46 CONTINUE
      IFLAG=0
   50 CONTINUE
      WRITE(6,100)
  100 FORMAT(1X,'INPUT RETAIN(1) OR CHANGE(0) FOR EACH EIGENSOLUTION )
      READ(5,*)(KRET(2,J),J=1,N)
      DO 110 I=1,N
      IF(KRET(2,I).EQ.1)GO TO 120
      IF(KRET(1,I).EQ.0)GO TO 105
      I1=I-1
      WRITE(6,107)I,I1
  107 FORMAT(1X,'DESIRED EIGENVALUE ',I2,'. WILL BE ENTERED AS THE CONJ
     +UGATE OF DESIRED EIGENVALUE ',I2,'.')
      ERD(I)=ERD(I1)
      EID(I)=-EID(I1)
      GO TO 110
  105 IF(VECRD(NM1,I).NE.0..OR.VECRD(N,I).NE.0.)GO TO 110
      WRITE(6,101)I
  101 FORMAT(1X,'INPUT DESIRED REAL PART, IMAGINARY PART OF NEW',1X,
     +'EIGENVALUE ',I2,'.'/)
      READ(5,*)ERD(I),EID(I)
      GO TO 110
  120 DO 121 J=1,2
      WR(J,I)=0.
      WI(J,I)=0.
  121 CONTINUE
  110 CONTINUE
      DO 140 I=1,N
      IF(KRET(2,I).EQ.1)GO TO 140
      IF(KRET(1,I).EQ.0)GO TO 113
      I1=I-1
      WRITE(6,114)I,I1
  114 FORMAT(1X,'DESIRED EIGENVECTOR ',I2,'. WILL BE ENTERED AS THE CONJ
     +UGATE OF DESIRED EIGENVECTOR ',I2,'.')
      DO 115 J=1,N
      VECRD(J,I)=VECRD(J,I1)
      VECID(J,I)=-VECID(J,I1)
  115 CONTINUE
      GO TO 140
  113 WRITE(6,111)I
  111 FORMAT(1X,'INPUT DESIRED REAL PART, IMAGINARY PART OF NEW EIGENVECT
     +OR ',I2,'.')
      JN=N
      IF(VECRD(NM1,I).NE.0..OR.VECRD(N,I).NE.0.)JN=NM2
      DO 112 J=1,JN
      READ(5,*)VECRD(J,I),VECID(J,I)
  112 CONTINUE
  140 CONTINUE
      DO 150 I=1,N
```

```
      IF(KRET(1,I).EQ.0)GO TO 153
      I1=I-1
      DO 151 J=1,N
      VECRD(J,I)=VECRD(J,I1)
      VECID(J,I)=-VECID(J,I1)
  151 CONTINUE
      DO 152 J=1,2
      WR(J,I)=WR(J,I1)
      WI(J,I)=-WI(J,I1)
  152 CONTINUE
      GO TO 150
  153 IF(VECRD(NM1,I).NE.0..OR.VECRD(N,I).NE.0.)GO TO 154
      CALL PROG2(I,ERD,EID,WR,WI,VECRD,VECID,A,B,N,N2,OB,LB,YB,
     +BR,II,LBT,VB,P,C)
      GO TO 150
  154 CALL PROG4(I,ERD,WR,VECRD,A,B,N,NM2,NM1,OB,LB,YB,BR,II,
     +LBT,VB,P,C,PP,DV)
  150 CONTINUE
      DO 160 I=1,N
      I1=I-1
      IF(KRET(1,I).EQ.0)GO TO 160
      DO 161 J=1,N
      VECRD(J,I)=VECID(J,I1)
  161 CONTINUE
      DO 162 J=1,2
      WR(J,I)=WI(J,I1)
  162 CONTINUE
  160 CONTINUE
      EPS=1.E-10
C
C     FIND INVERSE(VECRD)
C
      CALL MXLNEQ(VECRD,N,N,DET,JRANK,EPS,VECID)
      CALL MATMUL(WR,VECRD,WI,2,N,N)
      DO 165 I=1,N
      WRITE(3,164)(WI(J,I),J=1,2)
  164 FORMAT(3E16.8)
  165 CONTINUE
      CALL MATMUL(B,WI,VECID,N,2,N)
      CALL MATADD(A,VECID,VECRD,N,N)
      CALL PROG3(N,VECRD,ERD,EID,Z,IVI,FVI,KRET)
      CALL REPLACE(5HTAPE3,5HGAINS,0,0)
      RETURN
      END
C
C
C.....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)
```

```
      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
C
C.....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
C
C
C.....B(M,N)=S * A(M,N)
      SUBROUTINE SCAMAT(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
C
C
C.....V(N,M)= A(M,N) TRANSPOSED
      SUBROUTINE MATTRA(A,V,M,N)
      DIMENSION A(M,N),V(N,M)
      DO 14 I=1,M
      DO 14 J=1,N
      V(J,I)=A(I,J)
   14 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE PROG3(N,AA,WR,WI,Z,IV1,FV1,KRET)
      DIMENSION AA(N,N),WR(N),WI(N),Z(N,N),IV1(N),FV1(N),KRET(2,N)
C
C     CALCULATE CLOSED LOOP EIGENVALUES AND EIGENVECTORS
```

```
      WRITE(6,23)
   23 FORMAT(41X,'CLOSED LOOP EIGENVALUES'/31X,'REAL PART',20X,
+'IMAGINARY PART')
      DO 30 I=1,N
      WRITE(6,25)I,WR(I),WI(I)
   25 FORMAT(20X,I2,'.',2X,E15.4,15X,E15.4)
   30 CONTINUE
      WRITE(6,55)
   55 FORMAT(1X,'DO YOU WANT THE CLOSED LOOP EIGENVECTORS OUTPUT? YES(1)
+,NO(0)')
      READ(5,*)MM
      IF(MM.EQ.0)GO TO 60
      WRITE(6,64)
   64 FORMAT(1X,'INPUT PRINT(1) OR SUPPRESS(0) FOR EACH EIGENVECTOR')
      READ(5,*)(KRET(1,J),J=1,N)
      WRITE(6,34)
   34 FORMAT(41X,'CLOSED LOOP EIGENVECTORS'/31X,'REAL PART',20X,
+'IMAGINARY PART')
      IFLAG=0
      DO 50 J=1,N
      IF(KRET(1,J).EQ.0)GO TO 49
      IF(WI(J).NE.0.0)GO TO 35
      DO 37 I=1,N
      IF(I.NE.1)GO TO 36
      WRITE(6,38)J,Z(1,J)
   38 FORMAT(//20X,I2,'.',2X,E15.4,20X,'0.0')
      GO TO 37
   36 WRITE(6,39)Z(I,J)
   39 FORMAT(25X,E15.4,20X,'0.0')
   37 CONTINUE
      GO TO 50
   35 IF(IFLAG.EQ.1)GO TO 40
      IFLAG=1
      DO 41 I=1,N
      IF(I.NE.1)GO TO 42
      WRITE(6,43)J,Z(1,J),Z(1,J+1)
   43 FORMAT(//20X,I2,'.',2X,E15.4,15X,E15.4)
      GO TO 41
   42 WRITE(6,44)Z(I,J),Z(I,J+1)
   44 FORMAT(25X,E15.4,15X,E15.4)
   41 CONTINUE
      GO TO 50
   40 J1=J-1
      WRITE(6,45)J,J1
   45 FORMAT(//20X,I2,'.',5X,'CONJUGATE OF EIGENVECTOR',1X,I2,'.')
   49 IFLAG=0
   50 CONTINUE
   60 RETURN
      END
```

```
      SUBROUTINE PROG2(ID,ERD,EID,WR,WI,VECRD,VECID,A,B,N,N2,OB,LB,YB,
+BB,II,LBT,VB,P,C)
      DIMENSION ERD(N),EID(N),WR(2,N),WI(2,N),VECRD(N,N),VECID(N,N),
+A(N,N),B(N,2),OB(N2,N2),WB(4,1),VB(N2,1),YB(N2,N2),BB(N2,4),
+C(N,N),R(4,4),P(N2,1),T(4,1),RR(4,4)
     REAL II(N,N),LB(N2,4),LBT(4,N2),LAMR,LAMI
      LAMR=ERD(ID)
      LAMI=EID(ID)
      DO 5 I=1,N
      VB(I,1)=VECRD(I,ID)
      VB(I+N,1)=VECID(I,ID)
    5 CONTINUE
      DO 10 I=1,N
      DO 10 J=1,N
      II(I,J)=0.
      IF(I.EQ.J)II(I,J)=1.
   10 CONTINUE
      CALL SCAMAT(LAMR,II,C,N,N)
      DO 20 I=1,N
      DO 20 J=1,N
      YB(I,J)=C(I,J)-A(I,J)
      YB(I+N,J+N)=YB(I,J)
   20 CONTINUE
      CALL SCAMAT(LAMI,II,C,N,N)
      DO 30 I=1,N
      DO 30 J=1,N
      YB(I,J+N)=-C(I,J)
      YB(I+N,J)=C(I,J)
   30 CONTINUE
      EPS=1.E-10
C
C     FIND INVERSE(YB)
C
      CALL MXLNEQ(YB,N2,N2,DET,JRANK,EPS,OB)
      DO 40 I=1,N
      DO 40 J=1,2
      BB(I,J)=B(I,J)
      BB(I+N,J)=0.
      BB(I,J+2)=0.
      BB(I+N,J+2)=B(I,J)
   40 CONTINUE
      CALL MATMUL(YB,BB,LB,N2,N2,4)
      DO 42 I=1,N2
      DO 42 J=1,N2
      OB(I,J)=0.
   42 CONTINUE
      WRITE(6,41)ID
   41 FORMAT(1X,'INPUT REAL PART OF THE DIAGONAL OF THE OB MATRIX FOR
+EIGENVECTOR ',I2,'.')
```

```
C
      READ(5,*)(QB(J,J),J=1,N)
      WRITE(6,43)ID
   43 FORMAT(1X,'INPUT IMAGINARY PART OF THE DIAGONAL OF THE QB MATRIX
+FOR EIGENVECTOR ',I2,'.')
      READ(5,*)(QB(J+N,J+N),J=1,N)
      CALL MATMUL(QB,LB,BB,N2,N2,4)
      CALL MATTRA(LB,LBT,N2,4)
      CALL MATMUL(LBT,BB,R,4,N2,4)
C
C     FIND INVERSE(R)
C
      CALL MXLNEQ(R,4,4,DET,JRANK,EPS,RR)
      CALL MATMUL(QB,VB,P,N2,N2,1)
      CALL MATMUL(LBT,P,T,4,N2,1)
      CALL MATMUL(R,T,WB,4,4,1)
      CALL MATMUL(LB,WB,VB,N2,4,1)
      DO 60 I=1,N
      VECRD(I,ID)=VB(I,1)
      VECID(I,ID)=VB(I+N,1)
   60 CONTINUE
      DO 70 I=1,2
      WR(I,ID)=WB(I,1)
      WI(I,ID)=WB(I+2,1)
   70 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE PROG4(ID,ERD,WR,VECRD,A,B,N,NM2,NM1,QB,LB,YB,
+BB,II,LBT,VB,PP,C,P,DV)
C
C     THIS ROUTINE DETERMINES THE CLOSED LOOP ATTAINABLE
C     EIGENVECTORS FOR THE UNCONTROLLABLE GUST MODES
C
      DIMENSION ERD(N),WR(2,N),VECRD(N,N),A(N,N),B(N,2),QB(NM2,
+NM2),WB(2,1),VB(NM2,1),YB(NM2,NM2),BB(NM2,2),C(N,N),R(2,2),
+PP(NM2,1),T(2,1),RR(2,2),P(NM2,2),VD(2,1),DV(NM2,1)
      REAL II(N,N),LB(NM2,2),LBT(2,NM2),LAMD
      LAMD=ERD(ID)
      DO 5 I=1,NM2
      VB(I,1)=VECRD(I,ID)
    5 CONTINUE
      VD(1,1)=VECRD(NM1,ID)
      VD(2,1)=VECRD(N,ID)
      DO 10 I=1,N
      DO 10 J=1,N
      II(I,J)=0.
      IF(I.EQ.J)II(I,J)=1.
```

```
      DO 20 I=1,NM2
      P(I,1)=C(I,NM1)-A(I,NM1)
      P(I,2)=C(I,N)-A(I,N)
      DO 20 J=1,NM2
      YB(I,J)=C(I,J)-A(I,J)
   20 CONTINUE
      EPS=1.E-10
C
C     FIND INVERSE(YB)
C
      CALL MXLNEQ(YB,NM2,NM2,DET,JRANK,EPS,QB)
      DO 40 I=1,NM2
      DO 40 J=1,2
      BB(I,J)=R(I,J)
   40 CONTINUE
      CALL MATMUL(YB,P,LB,NM2,NM2,2)
      CALL MATMUL(LB,VD,DV,NM2,2,1)
      CALL MATMUL(YB,BB,LB,NM2,NM2,2)
      CALL MATADD(VB,DV,VB,NM2,1)
      DO 42 I=1,NM2
      DO 42 J=1,NM2
      QB(I,J)=0.
   42 CONTINUE
      WRITE(6,41)ID
   41 FORMAT(1X,'INPUT REAL PART OF THE DIAGONAL OF THE QB MATRIX FOR
     +EIGENVECTOR ',I2,'.')
      READ(5,*)(QB(J,J),J=1,NM2)
      CALL MATMUL(QB,LB,BB,NM2,NM2,2)
      CALL MATTRA(LB,LBT,NM2,2)
      CALL MATMUL(LBT,BB,R,2,NM2,2)
C
C     FIND INVERSE(R)
C
      CALL MXLNEQ(R,2,2,DET,IRANK,EPS,RR)
      CALL MATMUL(QB,VB,PP,NM2,NM2,1)
      CALL MATMUL(LBT,PP,T,2,NM2,1)
      CALL MATMUL(R,T,WB,2,2,1)
      CALL MATMUL(LB,WB,VB,NM2,2,1)
      CALL SCAMAT(-1.,DV,DV,NM2,1)
      CALL MATADD(VB,DV,VB,NM2,1)
      DO 60 I=1,NM2
      VECRD(I,ID)=VB(I,1)
   60 CONTINUE
      DO 70 I=1,2
      WR(I,ID)=WB(I,1)
   70 CONTINUE
      RETURN
      END
```

```
      PROGRAM COV(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,FILE3,
+TAPE1=FILE3,GAINS,TAPE3=GAINS,TAPE2)
C++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
C     THIS PROGRAM INPUTS THE STATE MATRICES
C     (OUTPUT OF FLUTTER) A, B, W, FEEDBACK
C     GAINS(OUTPUT OF MODAL), INTENSITY OF THE
C     GUST NOISE AND OUTPUTS THE CLOSED LOOP
C     STATE COVARIANCE MATRIX, RMS CONTROL SURFACE
C     RATES AND DEFLECTIONS IN DEGREES.(NOTE: INPUT
C     ZERO FEEDBACK GAINS FOR OPEN LOOP RESULTS)
C**********************************************************
      DIMENSION A(42,42),WI(2,42),Z(42,42),
+FV1(42),VECRD(42,42),VECID(42,42),W(42,1),
+B(42,2)
      WRITE(6,10)
   10 FORMAT(1X,'INPUT ORDER OF SYSTEM')
      READ(5,*)N
      CALL PROG1(N,A,WI,Z,FV1,VECRD,VECID,
+W,B)
      STOP
      END
C
C
      SUBROUTINE PROG1(N,A,WI,Z,FV1,VECRD,VECID,
+W,B)
      DIMENSION A(N,N),WI(2,N),Z(N,N),FV1(N),
+VECRD(N,N),VECID(N,N),W(N,1),B(N,2)
      DIMENSION PP(42,42),DD(42,42),WT(1,42)
      REAL II(42,42)
      CALL GETPF(5HTAPE1,5HFILE3,0,0)
C
C     READ IN STATE MATRICES OUTPUT FROM PROGRAM: FLUTTER
C
      DO 13 I=1,N
      READ(1,15)W(I,1)
   15 FORMAT(3E16.8)
   13 CONTINUE
      DO 16 I=1,N
      DO 16 J=1,N
      READ(1,15)A(I,J)
   16 CONTINUE
      DO 17 I=1,N
      READ(1,15)B(I,1),B(I,2)
   17 CONTINUE
```

```
C     READ IN FEEDBACK GAINS OUTPUT FROM PROGRAM: MODAL
C
      DO 24 I=1,N
      READ(3,*)(WI(J,I),J=1,2)
   24 CONTINUE
      CALL MATMUL(B,WI,VECID,N,2,N)
      CALL MATADD(A,VECID,VECRD,N,N)
      CALL PROG4(N,VECRD,VECID,A,Z,II,PP,DD,FV1,W,WT)
      RETURN
      END
C
C
C.....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
C
C.....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
C
C
C.....B(M,N)=S * A(M,N)
      SUBROUTINE SCAMAT(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
```

```
C.....V(N,M)= A(M,N) TRANSPOSED
      SUBROUTINE MATTRA(A,V,M,N)
      DIMENSION A(M,N),V(N,M)
      DO 14 I=1,M
      DO 14 J=1,N
      V(J,I)=A(I,J)
   14 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE PROG4(N,A,X,Q,XN,E,P,DX,WA,D,DT)
      DIMENSION A(N,N),X(N,N),Q(N,N),XN(N,N),E(N,N),P(N,N),DX(N,N),
     +WA(N),D(N,1),DT(1,N)
      WRITE(6,10)
   10 FORMAT(1X,'INPUT THE INTENSITY OF THE GUST NOISE, R.')
      READ(5,*)R
      CALL MATTRA(D,DT,N,1)
      CALL SCAMAT(R,D,D,N,1)
      CALL MATMUL(D,DT,Q,N,1,N)
      CALL CAL(A,Q,XN,N,15,2,X,E,P,DX,WA)
      WRITE(6,35)
      NN=N-9
      ZZ=57.3*SQRT(XN(NN+1,NN+1))
      WRITE(6,38)ZZ
      ZZ=57.3*SQRT(XN(NN+2,NN+2))
      WRITE(6,39)ZZ
      ZZ=57.3*SQRT(XN(NN+4,NN+4))
      WRITE(6,40)ZZ
       ZZ=57.3*SQRT(XN(NN+5,NN+5))
       WRITE(6,41)ZZ
   38 FORMAT(1X,'OUTBOARD DEFLECTION =  ,E15.4)
   39 FORMAT(1X,'OUTBOARD RATE = ',E15.4)
   40 FORMAT(1X,'INBOARD DEFLECTION = ',E15.4)
   41 FORMAT(1X,'INBOARD RATE = ',E15.4)
   35 FORMAT(1X,'RMS OF CONTROL RESPONSE'//)
      DO 24 I=1,N
      DO 24 J=1,N
      WRITE(2,164)XN(I,J)
  164 FORMAT(E16.8)
   24 CONTINUE
      CALL REPLACE(5HTAPE2,5HCOVST,0,0)
      RETURN
      END
C
C
      SUBROUTINE CAL(A,Q,XN,N,IMAX,IT,X,E,P,DX,WA)
C
```

```
      DIMENSION A(N,N),D(N,N),XN(N,N)
      DIMENSION X(N,N),E(N,N),P(N,N),DX(N,N),WA(N)
      TR=0.
      DO 300 I=1,N
  300 TR=TR+A(I,I)
      FN=N
      IF(TR)301,302,301
  301 ALF=ABS(TR)/FN
      GOTO 303
  302 ALF=1.
  303 CONTINUE
      EE=.01
      NC=N*(N+1)
      NC=NC/2
      DO 60 I=1,N
      DO 63 J=1,N
      GOTO(61,62),IT
   61 P(I,J)=A(I,J)
      X(I,J)=A(I,J)
      GOTO 63
   62 P(I,J)=A(J,I)
      X(I,J)=A(J,I)
   63 CONTINUE
      P(I,I)=P(I,I)-ALF
      X(I,I)=X(I,I)-ALF
   60 CONTINUE
C
C     FIND INVERSE(P)
C
      CALL MXLNEQ(P,N,N,DET,N,1.E-14,WA)
      DO 4 I=1,N
      DO 4 J=1,N
      E(I,J)=0.
      DO 4 K=1,N
    4 E(I,J)=E(I,J)+P(K,I)*D(K,J)*2.*ALF
      DO 5 I=1,N
      DO 5 J=1,N
      XN(I,J)=0.
      DO 5 K=1,N
    5 XN(I,J)=XN(I,J)+E(I,K)*P(K,J)
      DO 7 I=1,N
      DO 8 J=1,N
    8 P(I,J)=P(I,J)*2.*ALF
    7 P(I,I)=P(I,I)+1.
      ITER=0
  100 CONTINUE
      DO 9 I=1,N
      DO 9 J=1,N
      E(I,J)=0.
```

```
    9 E(I,J)=E(I,J)+P(K,I)*XN(K,J)
      DO 10 I=1,N
      DO 10 J=I,N
      DX(I,J)=0.
      DO 10 K=1,N
   10 DX(I,J)=DX(I,J)+E(I,K)*P(K,J)
      DO 20 I=1,N
      DO 20 J=1,N
      E(I,J)=P(I,J)
   20 X(I,J)=P(I,J)
      DO 12 I=1,N
      DO 12 J=I,N
      XN(I,J)=XN(I,J)+DX(I,J)
   12 XN(J,I)=XN(I,J)
      ICOT=0
      DO 15 I=1,N
      DO 15 J=I,N
      IF(XN(I,J))201,14,201
  201 RAT=ABS(DX(I,J)/XN(I,J))
      IF(RAT-EE)14,14,70
   14 ICOT=ICOT+1
   15 CONTINUE
   70 CONTINUE
      IF(ICOT-NC)16,18,16
   16 DO 17 I=1,N
      DO 17 J=1,N
      P(I,J)=0.
      DO 17 K=1,N
   17 P(I,J)=P(I,J)+E(I,K)*X(K,J)
   18 ITER=ITER+1
      IF(ICOT-NC)40,50,40
   40 IF(ITER-IMAX)100,50,50
   50 CONTINUE
      PRINT 600,ITER
  600 FORMAT(10X,'INNER LOOP TERMINATED AT ITER = ',I2)
      RETURN
      END
```

82/12/15
MNFTS PROGRAM LOADS

```
      PROGRAM LOADS(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,COVST,
+TAPE3=COVST,LMAT,TAPE2=LMAT)
C*****************************************************************
C     THIS PROGRAM INPUTS CLOSED LOOP STATE COVARIANCE MATRIX
C     (OUTPUT FROM COV), LOADS INFLUENCE MATRIX AND OUTPUTS
C     THE RMS SHEAR, RMS BENDING MOMENT, AND RMS TORQUE AT
C     VARIOUS STATIONS ALONG THE WING SPAN.
C****************************************************************
      REAL XN(42,42),T(10,42),LT(9,42),LTT(42,9),TEMP(42,9)
      DIMENSION KRET(8)
      WRITE(6,10)
   10 FORMAT(1X,'INPUT WHICH STATES WHERE RETAINED(1) OR DELETED(0)?')
      READ(5,*)(KRET(J),J=1,8)
      KSUM=0
      DO 20 I=1,8
      KSUM=KSUM+KRET(I)
   20 CONTINUE
      WRITE(6,11)
   11 FORMAT(1X,'INPUT THE NUMBER OF LAG STATES?')
      READ(5,*)M
      KST=KSUM*(2+M)
      KKSUM=KST+9
      CALL PROG1(KST,KKSUM,KRET,XN,T,LT,LTT,TEMP)
      STOP
      END
C
C
      SUBROUTINE PROG1(KST,KKSUM,KRET,XN,T,LT,LTT,TEMP)
      REAL LBM(9,10),LSH(9,10),LTO(9,10),XN(KKSUM,KKSUM),T(10,KKSUM),
+LT(9,KKSUM),LTT(KKSUM,9),TEMP(KKSUM,9),COV(9,9),BM(9),SH(9),TO(9)
      DIMENSION KRET(8)
      CALL GETPF(5HTAPE2,4HLMAT,0,0)
      DO 10 I=1,10
      DO 10 J=1,9
C
C     READ IN LOADS INFLUENCE MATRIX
C
      READ(2,15)LBM(J,I),LSH(J,I),LTO(J,I)
   15 FORMAT(3E16.8)
   10 CONTINUE
      CALL GETPF(5HTAPE3,5HCOVST,0,0)
      DO 20 I=1,KKSUM
      DO 20 J=1,KKSUM
```

```
C
      READ(3,15)XN(I,J)
   20 CONTINUE
      DO 40 I=1,10
      DO 40 J=1,KKSUM
      T(I,J)=0.
   40 CONTINUE
      J=0
      DO 50 I=1,8
      IF(KRET(I).EQ.0)GO TO 50
      J=J+1
      T(I,J)=1.
   50 CONTINUE
      T(9,KST+1)=.514
      T(10,KST+4)=.518
      CALL MATMUL(LBM,T,LT,9,10,KKSUM)
      CALL MATTRA(LT,LTT,9,KKSUM)
      CALL MATMUL(XN,LTT,TEMP,KKSUM,KKSUM,9)
      CALL MATMUL(LT,TEMP,COV,9,KKSUM,9)
      DO 60 I=1,9
      BM(I)=SQRT(COV(I,I))
   60 CONTINUE
      CALL MATMUL(LSH,T,LT,9,10,KKSUM)
      CALL MATTRA(LT,LTT,9,KKSUM)
      CALL MATMUL(XN,LTT,TEMP,KKSUM,KKSUM,9)
      CALL MATMUL(LT,TEMP,COV,9,KKSUM,9)
      DO 70 I=1,9
      SH(I)=SQRT(COV(I,I))
   70 CONTINUE
      CALL MATMUL(LTQ,T,LT,9,10,KKSUM)
      CALL MATTRA(LT,LTT,9,KKSUM)
      CALL MATMUL(XN,LTT,TEMP,KKSUM,KKSUM,9)
      CALL MATMUL(LT,TEMP,COV,9,KKSUM,9)
      DO 80 I=1,9
      TQ(I)=SQRT(COV(I,I))
   80 CONTINUE
      WRITE(6,90)
   90 FORMAT(35X,'RMS LOADS'/1X,'BEND MOM',27X,'SHEAR',27X,'TORQUE'//)
      DO 100 I=1,9
  110 FORMAT(5X,E16.4,10X,E16.4,10X,E16.4)
      WRITE(6,110)BM(I),SH(I),TQ(I)
  100 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE MATMUL(A,U,T,M,N,P)
      INTEGER P
      DIMENSION A(M,N),U(N,P),T(M,P)
```

```
      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
C
      SUBROUTINE MATTRA(A,V,M,N)
      DIMENSION A(M,N),V(N,M)
      DO 14 I=1,M
      DO 14 J=1,N
      V(J,I)=A(I,J)
   14 CONTINUE
      RETURN
      END
```

82/12/15

MNETS PROGRAM FEEDBKE

```
      PROGRAM FEEDBKE(INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT,ONELEE,
+TAPE2=ONELEE,MASS,TAPE1=MASS,GAINL,TAPE3=GAINL,TAPE4)
C*****************************************************************
C     THIS PROGRAM INPUTS THE STRUCTURAL MASS,STRUCTURAL STIFFNESS,
C     STRUCTURAL DAMPING, AERODYNAMIC INFLUENCE MATRICES, FLIGHT
C     CONDITION, FEEDBACK GAINS(OUTPUT OF MODAL) AND OUTPUTS THE
C     CLOSED LOOP EIGENVALUES.  THIS IS USEFUL TO STUDY OFF DESIGN
C     PERFORMANCE.
C************************************************************
      REAL MXX(8,8),CS(8,8),KS(8,8),CT(8,8),KT(8,8),CC(8,8),
+KK(8,8),A2X(8,8),A1X(8,8),A0X(8,8),MINV(8,8),A2X0(8,8)
+,A1X0(8,8),A0X0(8,8),A1(16,16),A2(32,32),00(8,2),DUM1(8,7),
+DUM2(8,7),DUM3(8,2),DUM4(8,7),DUM5(8,2),DUM6(8,2),DUM7(8,1)
      REAL DUM8(8,1),A0U(8,2),A1U(8,2),EI(8,2,4),R(8,2),
+ETJG(8,7,4),FI(8,4),A3(40,40),B3(40,2),B2(32,2),II(8,8),
+DI(8,8,4),KT(4),A1D(8,1),A0D(8,1),T(8,1),U(8,1),A4(42,42),
+B4(42,2),W(42,1),B1(16,2),A5(24,24),B5(24,2)
      WRITE(6,8)
    8 FORMAT(1X,'INPUT THE ORDER OF THE STRUCTURAL-RIGID BODY STATE VECTO
+R(N)'/)
      READ(5,*)N
      WRITE(6,10)
   10 FORMAT(1X,'INPUT VELOCITY(V), CHORD(C), DYNAMIC PRESSURE(Q), NUMBE
+R OF LAG STATES(L)'/)
C
C     INPUT FLIGHT CONDITIONS
C
      READ(5,*)V,C,Q,L
      DO 20 I=1,L
      WRITE(6,25)I
C
C     INPUT AERODYNAMIC LAG FREQUENCIES
C
      READ(5,*)KI(I)
   20 CONTINUE
   25 FORMAT(1X,'INPUT KT(',I1,')'/)
      N2=2*N
      KEND=N*(L+2)
      KENDD=KEND+7
      KEND2=KENDD+2
      K5=2*N+7
      CALL PROG1(N,N2,KEND,KENDD,KEND2,K5,V,C,Q,L,MXX,CS,KS,CT,KT,
+CC,KK,A2X,A1X,A0X,MINV,A2X0,A1X0,A0X0,A1,A2,00,DUM1,DUM2,DUM3,DUM4
+,DUM5,DUM6,DUM7,DUM8,A0U,A1U,EI,R,ETJG,II,A3,B3,B2,FI,DI,KT,A1D,
```

```
      STOP
      END
C
C
      SUBROUTINE PROG1(N,N2,KEND,KENDD,KEND2,K5,V,C,Q,L,MXX,CS,KS,CT,KT,
+CC,KK,A2X,A1X,A0X,MINV,A2XQ,A1XQ,A0XQ,A1,A2,QQ,DUM1,DUM2,DUM3,DUM4
+,DUM5,DUM6,DUM7,DUM8,A0U,A1U,EI,R,EIJG,II,A3,B3,B2,FI,DI,KI,A1D,
+A0D,T,U,A4,B1,B4,W,A5,B5)
      REAL LAMDA,LL
      INTEGER FLAG
      COMMON KRET(42)
      REAL MXX(N,N),CS(N,N),KS(N,N),CT(N,N),KT(N,N),CC(N,N),KK(N,N),
+A2X(N,N),A1X(N,N),A0X(N,N),MINV(N,N),A2XQ(N,N),A1XQ(N,N),A0XQ(N,N)
+,A1(N2,N2),A2(KEND,KEND),QQ(N,2),DUM1(N,7),DUM2(N,7),DUM3(N,2),
+DUM4(N,7),DUM5(N,2),DUM6(N,2),DUM7(N,1),DUM8(N,1),A0U(N,2)
      REAL A1U(N,2),
+EI(N,2,L),G(7,7),H(7,2),R(N,2),JG(2,7),EIJG(N,7,L),FI(N,L),
+JH(2,2),A3(KENDD,KENDD),B3(KENDD,2),B2(KEND,2),II(N,N),DI(N,N,L)
      REAL KI(L),A1D(N,1),A0D(N,1),T(N,1),U(N,1),A4(KEND2,KEND2),
+B1(N2,2),B4(KEND2,2),W(KEND2,1),
+A5(K5,K5),B5(K5,2),JJ(2,7)
      REAL AA(42,42),BB(42,2)
      CALL GETPF(5HTAPE1,4HMASS,0,0)
      DO 1 I=1,N
      DO 1 J=1,N
C
C     READ IN STRUCTURAL DATA
C
      READ(1,101)MXX(I,J),CS(I,J),KS(I,J)
    1 CONTINUE
      DO 3 I=1,7
      DO 3 J=1,7
C
C     READ IN ACTUATOR DATA
C
      READ(1,101)G(I,J)
    3 CONTINUE
      DO 4 I=1,7
      READ(1,101)(H(I,J),J=1,2)
    4 CONTINUE
      DO 5 J=1,7
      READ(1,101)(JJ(I,J),I=1,2)
    5 CONTINUE
      CALL GETPF(5HTAPE2,6HONELEE,0,0)
      DO 430 K=1,L
      DO 410 J=1,N
      DO 410 I=1,N
C
C     READ IN AERODYNAMIC INFLUENCE DATA
C
```

```
  410 CONTINUE
      DO 420 I=1,2
      DO 420 I=1,N
      READ(2,101)A0U(I,J),A1U(I,J),ET(I,J,K)
  420 CONTINUE
      DO 430 I=1,N
      READ(2,101)A0D(I,K),A1D(I,K),FI(I,K)
      FI(I,K)=FI(I,K)/V
  430 CONTINUE
  101 FORMAT(3E16.8)
  109 FORMAT(4E16.8)
      EPS=1.E-10
      TEMP=Q*(C/2./V)**2
      TEMP2=Q*C/2./V
      CALL SCAMAT(TEMP,A2X,A2XQ,N,N)
      CALL MATADD(MXX,A2XQ,MINV,N,N)
C
C     FIND INVERSE(MINV)
C
      CALL MXLNEQ(MINV,N,N,DET,JRANK,EPS,A2XQ,0)
      CALL SCAMAT(TEMP2,A1X,A1XQ,N,N)
      CALL MATADD(CS,A1XQ,CT,N,N)
      CALL SCAMAT(Q,A0X,A0XQ,N,N)
      CALL MATADD(KS,A0XQ,KT,N,N)
      CALL SCAMAT(-1.,MINV,MINV,N,N)
      CALL MATMUL(MINV,KT,KK,N,N,N)
      CALL MATMUL(MINV,CT,CC,N,N,N)
      DO 15 I=1,N
      DO 14 J=1,N
      II(I,J)=0.
   14 CONTINUE
      II(I,I)=1.0
   15 CONTINUE
      N21=N2+1
      DO 18 I=1,N
      DO 18 J=1,N
      A1(I,J)=0.
      A1(I,J+N)=II(I,J)
      A1(I+N,J)=KK(I,J)
      A1(I+N,J+N)=CC(I,J)
   18 CONTINUE
      IF(L.EQ.0)GO TO 102
      DO 20 I=1,KEND
      DO 20 J=1,KEND
      A2(I,J)=0.
   20 CONTINUE
      DO 25 I=1,N
      DO 25 J=1,N
      A2(I,J)=A1(I,J)
      A2(I+N,J)=A1(I+N,J)
```

```
      DO 25 K=1,L
      KN=K*N
      A2(I+N,KN+N+J)=MINV(I,J)
      A2(KN+N+I,J+N)=DI(I,J,K)
      A2(KN+N+I,KN+N+I)=-2.*V*KI(K)/C
   25 CONTINUE
      SCA=Q*C/2./V
      CALL SCAMAT(SCA,A1U,QQ,N,2)
      CALL SCAMAT(Q,A0U,R,N,2)
      DO 36 I=1,KENDD
      DO 36 J=1,KENDD
      A3(I,J)=0.
   36 CONTINUE
      DO 37 I=1,KEND
      DO 37 J=1,KEND
      A3(I,J)=A2(I,J)
   37 CONTINUE
      DO 38 I=1,7
      DO 38 J=1,7
      A3(KEND+I,KEND+J)=G(I,J)
   38 CONTINUE
      CALL MATMUL(JJ,G,JG,2,7,7)
      CALL MATMUL(QQ,JG,DUM1,N,2,7)
      CALL MATMUL(R,JJ,DUM2,N,2,7)
      CALL MATADD(DUM2,DUM1,DUM2,N,7)
      CALL MATMUL(MINV,DUM2,DUM1,N,N,7)
      DO 40 K=1,L
      DO 39 I=1,N
      DO 39 J=1,2
      DUM3(I,J)=EI(I,J,K)
   39 CONTINUE
      CALL MATMUL(DUM3,JG,DUM4,N,2,7)
      DO 40 I=1,N
      DO 40 J=1,7
      EIJG(I,J,K)=DUM4(I,J)
   40 CONTINUE
      DO 45 I=1,N
      DO 45 J=1,7
      A3(I+N,J+KEND)=DUM1(I,J)
      DO 45 K=1,L
      KN=K*N
      A3(KN+N+I,J+KEND)=EIJG(I,J,K)
   45 CONTINUE
      CALL MATMUL(JJ,H,JH,2,7,2)
      CALL MATMUL(QQ,JH,DUM3,N,2,2)
      CALL MATMUL(MINV,DUM3,DUM5,N,N,2)
      DO 47 K=1,L
      DO 46 I=1,N
      DO 46 J=1,2
```

```
      CALL MATMUL(DUM3,JH,DUM6,N,2,2)
      DO 47 I=1,N
      DO 47 J=1,2
      ETJG(I,I,K)=DUM6(I,J)
   47 CONTINUE
      DO 48 I=1,N
      DO 48 I=1,2
      B3(I,J)=0.
      B3(I+N,J)=DUM5(I,J)
      DO 48 K=1,L
      KN=K*N
      B3(I+KN+N,J)=ETJG(I,J,K)
   48 CONTINUE
      DO 49 I=1,7
      DO 49 J=1,2
      B3(I+KEND,J)=H(I,J)
   49 CONTINUE
      WRITE(6,56)
   56 FORMAT(1X,'INPUT CHARACTERISTIC LENGTH(LL) OF NOISE'/)
      READ(5,*)LL
      TEMP3=TEMP2/V
      CALL SCAMAT(TEMP3,A1D,T,N,1)
      Q1=Q/V
      CALL SCAMAT(Q1,A0D,U,N,1)
      CALL MATMUL(MINV,U,DUM7,N,N,1)
      CALL MATMUL(MINV,T,DUM8,N,N,1)
      DO 58 I=1,KENDD
      DO 58 J=1,2
      B4(I,J)=B3(I,J)
      B4(KENDD+1,J)=0.
      B4(KENDD+2,J)=0.
   58 CONTINUE
      DO 59 I=1,KEND2
      DO 59 J=1,KEND2
      A4(I,J)=0.
   59 CONTINUE
      DO 60 I=1,KENDD
      DO 60 J=1,KENDD
      A4(I,J)=A3(I,J)
   60 CONTINUE
      A4(KENDD+1,KENDD+2)=1.
      A4(KENDD+2,KENDD+1)=-1.001*(V/LL)**2
      A4(KENDD+2,KENDD+2)=-2.001*V/LL
      DO 62 I=1,N
      A4(I+N,KENDD+1)=DUM7(I,1)
      A4(I+N,KENDD+2)=DUM8(I,1)
      DO 62 K=1,L
      KN=K*N
      A4(I+KN+N,KENDD+2)=ET(I,K)
```

```
      DO 64 I=1,KENDD
      W(I,1)=0.
   64 CONTINUE
      XX=1.732*V/LL
      W(KENDD+1,1)=XX
      W(KENDD+2,1)=-2.464*(V/LL)**2
      CALL SCAMAT(XX,DUM8,DUM8,N,1)
      DO 66 I=1,N
      W(I+N,1)=DUM8(I,1)
      DO 66 K=1,L
      KN=K*N
      W(I+N+KN,1)=XX*FI(I,K)
   66 CONTINUE
      FLAG=4
      GO TO 200
  102 SCA=D*C/2./V
      CALL SCAMAT(SCA,AIU,DD,N,2)
      CALL SCAMAT(D,AOU,R,N,2)
      DO 106 I=1,K5
      DO 106 J=1,K5
      A5(I,J)=0.
  106 CONTINUE
      CALL MATMUL(JJ,H,JH,2,7,2)
      CALL MATMUL(DD,JH,DUM3,N,2,2)
      CALL MATMUL(MINV,DUM3,DUM5,N,N,2)
      DO 108 I=1,N
      DO 108 J=1,2
      B5(I,J)=0.
      B5(I+N,J)=DUM5(I,J)
  108 CONTINUE
      DO 110 I=1,7
      DO 110 J=1,2
      B5(I+N2,J)=H(I,J)
  110 CONTINUE
      DO 112 I=1,N2
      DO 112 J=1,N2
      A5(I,J)=A1(I,J)
  112 CONTINUE
      CALL MATMUL(JJ,G,JG,2,7,7)
      CALL MATMUL(DD,JG,DUM1,N,2,7)
      CALL MATMUL(R,JJ,DUM2,N,2,7)
      CALL MATADD(DUM2,DUM1,DUM2,N,7)
      CALL MATMUL(MINV,DUM2,DUM1,N,N,7)
      DO 114 I=1,N
      DO 114 J=1,7
      A5(I+N,J+N2)=DUM1(I,J)
  114 CONTINUE
      DO 116 I=1,7
      DO 116 J=1,7
      A5(I+N2,J+N2)=G(I,J)
```

```
      FLAG=1
  200 WRITE(6,201)
  201 FORMAT(1X,'INPUT RETAIN(1) OR DELETE(0) FOR EACH STATE MEMBER'/)
      KSTATE=N
      KSUM=0
      READ(5,*)(KRET(J),J=1,KSTATE)
      DO 205 I=1,KSTATE
      KRET(I+N)=KRET(I)
      IF(KRET(I).EQ.1)KSUM=KSUM+1
  205 CONTINUE
      IF(L.EQ.0)GO TO 207
      DO 206 K=1,L
      DO 206 I=1,N
      KN=N*(K+1)+I
      KRET(KN)=KRET(I)
  206 CONTINUE
      DO 209 I=1,9
      KRET(KEND+I)=1
  209 CONTINUE
  207 GO TO(310,320,330,340,350),FLAG
  310 KSUM=2*KSUM
      CALL PROG2(FLAG,KSUM,A1,B1,N2,AA,BB)
      GO TO 900
  320 KSUM=(L+2)*KSUM
      CALL PROG2(FLAG,KSUM,A2,B2,KEND,AA,BB)
      GO TO 900
  330 KSUM=7+(L+2)*KSUM
      CALL PROG2(FLAG,KSUM,A3,B3,KENDD,AA,BB)
      GO TO 900
  340 KSUM=9+(L+2)*KSUM
      CALL PROG2(FLAG,KSUM,A4,B4,KEND2,AA,BB)
      GO TO 900
  350 KSUM=7+2*KSUM
      DO 351 I=N21,K5
      KRET(I)=1
  351 CONTINUE
      CALL PROG2(FLAG,KSUM,A5,B5,K5,AA,BB)
  900 RETURN
      END
C
C
      SUBROUTINE MATDEL(A,N,AA,KSUM)
C
C     THIS ROUTINE DELETES DESIRED STATES FROM THE A MATRIX
C
      DIMENSION A(N,N),AA(KSUM,KSUM),JRET(40)
      COMMON KRET(42)
      J=0
      DO 100 I=1,N
      IF(KRET(I).EQ.0)GO TO 100
```

```
  100 CONTINUE
      DO 200 I=1,KSUM
      DO 200 J=1,KSUM
      AA(I,J)=A(JRET(I),JRET(J))
  200 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE MATDELB(B,N,BB,KSUM)
C
C     THIS ROUTINE DELETES DESIRED STATES FROM THE B MATRIX
C
      DIMENSION B(N,2),BB(KSUM,2),JRET(40)
      COMMON KRET(42)
      J=0
      DO 100 I=1,N
      IF(KRET(I).EQ.0)GO TO 100
      J=J+1
      JRET(J)=I
  100 CONTINUE
      DO 200 I=1,KSUM
      DO 200 J=1,2
      BB(I,J)=B(JRET(I),J)
  200 CONTINUE
      RETURN
      END
C
C
C.....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
C
C.....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 J=1,N
      C(I,J)=A(I,J)+B(I,J)
    3 CONTINUE
      RETURN
      END
C
C
C.....B(M,N)=S * A(M,N)
      SUBROUTINE SCAMAT(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
C
C
C.....V(N,M)= A(M,N) TRANSPOSED
      SUBROUTINE MATTRA(A,V,M,N)
      DIMENSION A(M,N),V(N,M)
      DO 14 I=1,M
      DO 14 J=1,N
      V(J,I)=A(I,J)
   14 CONTINUE
      RETURN
      END
C
C
      SUBROUTINE PROG2(FLAG,KSUM,A,B,IP,AA,BB)
      INTEGER FLAG,IV1(42)
      REAL AA(KSUM,KSUM),BB(KSUM,2),A(IP,IP),B(IP,2),FV1(42)
      REAL WR(42),WI(42),Z(42,42),G(2,42)
      GO TO(310,320,330,340,350),FLAG
  100 FORMAT(3E16.8)
  310 CALL MATDEL(A,IP,AA,KSUM)
      DO 311 I=1,KSUM
      DO 311 J=1,KSUM
      WRITE(4,100)AA(I,J)
  311 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1,BB,G)
      GO TO 900
  320 CALL MATDEL(A,IP,AA,KSUM)
      DO 321 I=1,KSUM
      DO 321 J=1,KSUM
      WRITE(4,100)AA(I,J)
  321 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1,BB,G)
      GO TO 900
  330 CALL MATDEL(A,IP,AA,KSUM)
```

```
      DO 331 I=1,KSUM
      DO 331 J=1,KSUM
      WRITE(4,100)AA(I,J)
  331 CONTINUE
      DO 332 I=1,KSUM
      WRITE(4,100)BB(I,1),BB(I,2)
  332 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1,BB,G)
      GO TO 900
  340 CALL MATDEL(A,IP,AA,KSUM)
      CALL MATDELB(B,IP,BB,KSUM)
      DO 341 I=1,KSUM
      DO 341 J=1,KSUM
      WRITE(4,100)AA(I,J)
  341 CONTINUE
      DO 342 I=1,KSUM
      WRITE(4,100)BB(I,1),BB(I,2)
  342 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1,BB,G)
      GO TO 900
  350 CALL MATDEL(A,IP,AA,KSUM)
      CALL MATDELB(B,IP,BB,KSUM)
      DO 351 I=1,KSUM
      DO 351 J=1,KSUM
      WRITE(4,100)AA(I,J)
  351 CONTINUE
      DO 352 I=1,KSUM
      WRITE(4,100)BB(I,1),BB(I,2)
  352 CONTINUE
      CALL PROG3(KSUM,AA,WR,WI,Z,IV1,FV1,BB,G)
  900 RETURN
      END
C
C
      SUBROUTINE PROG3(N,AA,WR,WI,Z,IV1,FV1,BB,G)
      COMMON KRET(42)
      DIMENSION AA(N,N),WR(N),WI(N),Z(N,N),IV1(N),FV1(N),G(2,N),BB(N,2)
C
C     READ IN FEEDBACK GAINS OUTPUT FROM PROGRAM: MODAL
C
      CALL GETPF(5HTAPE3,5HGAINL,0,0)
      DO 24 I=1,N
      READ(3,*)(G(J,I),J=1,2)
   24 CONTINUE
      CALL MATMUL(BB,G,Z,N,2,N)
      CALL MATADD(AA,Z,AA,N,N)
C
C     CALCULATE CLOSED LOOP EIGENVALUES
C
      CALL RG(N,N,AA,WR,WI,1,Z,IV1,FV1,IERR)
```

-106-

```
[illegible]
     +'IMAGINARY PART')
      DO 30 I=1,N
      WRITE(6,25)I,WR(I),WI(I)
   25 FORMAT(20X,I2,'.',2X,E15.4,15X,E15.4)
   30 CONTINUE
      RETURN
      END
```