GRAPHICS FLUTTER ANALYSIS METHODS

An Interactive Computing System at Lockheed-California Company

Nick A. Radovcich

Lockheed-California Company

SUMMARY

An interactive computer graphics system, Graphics Flutter Analysis Methods (GFAM) was developed by Lockheed-California Company to complement FAMAS, Lockheed's matrix-oriented batch computing system, and other computer programs in performing complex numerical calculations using a fully integrated data management system. GFAM has many of the matrix operation capabilities found in FAMAS, but on a smaller scale, and is utilized when the analysis requires a high degree of interaction between the engineer and computer, and schedule constraints exclude the use of batch entry programs. Applications of GFAM to a variety of preliminary design, development design, and project modification programs suggest that interactive flutter analysis using matrix representations is a feasible and cost effective computing tool.

INTRODUCTION

The calendar time span required to transform aerodynamic data, airframe inertia, and structural design criteria into the form necessary for the structural design and sizing of the airframe has always occupied a large share of the total available time between concept and hardware. Releasing drawings to manufacturing before final definition of design loads and structural sizing are complete can lead to costly design and manufacturing changes.

The structural analysis process requires the interaction of seven major engineering functions: Loft, Design, Aerodynamics, Weights, Structural Analysis, Loads and Criteria, and Flutter. A sample of the organizational interfaces is shown in Figure 1.

Figure 2 illustrates Lockheed's present status and approach to computer-aided design for Structures and Aeromechanics. The Common Matrix Data is a data storage and access system common to FAMAS, NASTRAN, GFAM and auxiliary programs, such as aerodynamics transformation and safety margin. The data format is in the form of 2-dimensional arrays (matrices) which are identified by two 4-digit numbers, and, if necessary, the date the matrices were created. The computer automatically catalogs the identification when a new matrix is

entered into Common Matrix Data System; thus the computer is able to locate the matrix when requested by any one of the programs having access to the matrix data storage.

FAMAS (Flutter And Matrix Algebra System), a Lockheed-developed computing system, performs static and dynamic loads analysis, flutter analysis, plotting, response analysis, and a complete set of linear algebraic operations, such as matrix inverse, matrix eigenvalue problems, matrix multiplication, etc., in addition to standard non-matrix-algebraic operations such as element by element multiplication, store, extract, etc.

NASTRAN is a structural modeling and analysis program which performs vibration analysis and certain matrix operations found also in FAMAS.

Computer graphics application at Lockheed-California Company is divided into two functions: drafting package and analytics. The primary drafting package program is CADAM (Computer Augmented Design and Manufacturing), a powerful design tool used in lofting, drafting, and numerical control tasks. CADAM has its own data base and data management system and drives 7 or more scopes per 130K core partition.

Under analytics, a variety of user application programs are executed, each, when running, requiring a separate 126K core allocation. The ICSMP (Interactive Continuous System Model Program) package is used extensively in the simulations of the airplane for maneuver, landing, and impact analysis, by integrating a set of ordinary differential equations. A number of graphics programs assist aerodynamics and other disciplines. Interactive flutter analysis, formerly done using the Network Analyzer, a direct analog computer, is now available as graphics analytic program, GFAM.

This paper reports on the development and use of GFAM to accelerate the design process in satisfying the flutter requirements, especially when the design exhibits flutter deficiencies.

DESCRIPTION

General

GFAM, using a matrix data base generated for batch flutter analysis in the FAMAS system, performs interactive flutter analysis, structural optimization to satisfy flutter requirements, control synthesis for CCV (Control Configured Vehicle) applications, general matrix algebra operations, and the matching of structural dynamics analysis to ground vibration test data. GFAM supports test data correlation, flutter methods development, and quick analysis of a design for flutter and structural dynamic characteristics during preliminary and point design phases. GFAM applications will expand into a more extensive graphics generation capability of the stiffness and mass characteristics of the design from coordinate and physical property descriptions. Future discipline additions may include limited loads and

stress analyses. Terminal requirements and other pertinent data are tabulated in Table 1; GFAM's general computing capabilities are summarized in Figure 3, and the computing system location in the IBM 360 computer tree is shown in Figure 4.

GFAM Control Display

The engineer controls the operation of GFAM through a graphics display shown in Figure 5. Column one lists members containing 80-column data cards (image format) e.g., program code as listed in Appendix A, which are moved to the card input file, FT05, when the member name is detected by the light pen. The data in the input file can be modified, copied, merged, deleted, or saved as another member. The card image input file is accessible by programs listed in column three. Each of these programs which operates in the interactive mode is identified by an 8-character name and can be individually loaded and executed using the SUBMIT option.

Column two in the GFAM control display lists users' disc files which store matrix data accessible by programs submitted for execution. These data storage files are available while the problem solution is actively pursued. Matrix data may be moved by the user to tape or disc in the FAMAS system for long term storage or for access by the FAMAS system.

The following is a description of GFAM control display menu:

FETCH: Causes the cards in the detected member to be shown on the scope for viewing and editing.

SAVE: Saves a modified card data set as a new or old member in the FT05 column.

PRINT: Prints the active FT05 data set member.

PURGE: Purges a card data set from the FT05 column.

SUBMIT: Transfers control to one of the programs in the PROGRAMS column.

OUTPUT: Permits the review of print and punch output of the last SUBMIT. The user has the option to send the print output to the printer and to save the punch data set as another FT05 member.

LEAVE: Terminates a session at the scope.

Program MATRIX

MATRIX is a general matrix algebra computing program which is used to generate, condition, and manage matrix data requirements of the technology modules which will be discussed subsequently.

Each matrix algebra operation is performed interactively, and the average wait time at the scope for each operation is 10 seconds, with the maximum wait time designed to be not more than 2 minutes.

The following are examples of user-supplied coding that performs the more common matrix operations:

(1) Create new stiffness matrix after changing the (3,3) element of the compliance matrix (see equation (8)).

```
C @ 3001              identifies matrix 3001 as C
CM @ 2001             identifies matrix 2001 as CM
C(3,3) = .006         modifies the (3,3) element of C
CI = INV(C)           generates the inverse of C
K = CM*CI*CM'         forms stiffness matrix
PRNT K,C              prints matrices K and C
RNAM K as 4000        reidentifies variable K as matrix 4000
KEEP 4000             moves matrix 4000 to user file
```

(2) Perform static reduction to 100 degrees of freedom on a 200 x 200 $[K]$ matrix partitioned such that $[K]=\begin{bmatrix} K11 & K12 \\ K21 & K22 \end{bmatrix}$. The reduced stiffness matrix is defined by:

$$KR = K11 - K12 * K22^{-1} * K21 \qquad (1)$$

```
K22 = XTRC 100 BY 100 FROM K (101,101)
K12 = XTRC 100 BY 100 FROM K(1,101)
K11 = XTRC 100 BY 100 FROM K(1,1)
K22I = INV (K22)
KMD = K12 * K22I * K12'
KR = K11 - KMD
```

or by a macro instruction,

```
KR = GYAN (K) BY 100.
```

With the macro instruction,

```
KR = GYAN (K) BY V SAVE G
```

the program will shuffle [K] according to the vector V and output the Guyan transformation matrix [G] for use on a consistent mass matrix [M] where

$$[MG] = [G]^T [M] [G] . \qquad (2)$$

(3) Perform a vibration analysis:

```
M @ 1002                identifies matrix 1002 as M

K @ 3003                identifies matrix 3003 as K

MI = INV(M)             forms inverse of  M

A = MI*K

ROOT = EIG(A)           extracts eigenvalues by QR

V = VECT(ROOT, A)
FROM 1 to 30            extracts eigenvectors for the first
                        30 roots

RNAM V AS 5000          reidentifies variable V as matrix 5000

PRNT ROOT, V            Prints matrices ROOT and V

KEEP 5000               move matrix 5000 to user file
```

Appendix A includes another MATRIX program example with line by line explanation of the code.

The first MATRIX display with which the user interacts is a restart option from a previous checkpoint. The next display is the INPUT DISPLAY shown in Figure 6. From here the user may read matrix data from tape or disc/DATA READ/, write matrix data on tape or disc/DATA WRITE/, edit the user file/MATRIX TABLE DISPLAY/, return to the user program/PROGRAM/, and move the user program cards from the FT05 active file into MATRIX program file/CARDS/.

The user-supplied coding to be used by program MATRIX may be accessed with the/CARD/option, from the FT05 members saved previously, or the user may generate the coding directly in the PROGRAM DISPLAY shown in Figure 7. The user also controls the execution of the coding from this display and has the choice of executing the code singularly or in designated blocks. If the coding is incompatible with the indicated matrix operation requirements, an error code is given the user indicating the type of error and suggesting the required action for its correction.

Another function of MATRIX is data management of the matrices in the user files. Figures 8 to 11 illustrate displays used for reading data from tape (FAMAS) or disc into GFAM users' file. Figure 12 is a display that the user sees after a read attempt. Matrices may be easily moved from one user file to another.

Data write function is controlled by DATA WRITE DISPLAY shown in Figure 13. The user may write a FAMAS tape to save his matrix data indefinitely and make it available for batch programs using the FAMAS matrix format.

Matrix management and the associated identifiers of the user file is possible in the MATRIX TABLE DISPLAY shown in Figure 14, where matrices may be purged and the file rebuilt. The MATRIX TABLE DISPLAY lists the number of matrices, KOUNT, on the user file; the matrix number, e.g., 9090; the number of rows of the matrix; number of columns; the type (real or complex); the logical file on which the matrix is stored, FILE; the relative position on the file, POS; and three identifiers, FREQ, DAMP, and SET#. FREQ and DAMP are two numeric values associated with each matrix on the file. For example, all aerodynamic matrices have a value of reduced frequency associated with the matrix. By convention, the value of the reduced frequency is stored under the FREQ identifier, e.g., matrix 1863 has a value of .760 for reduced frequency in Figure 14. Values of FREQ and DAMP can represent almost anything the programmer/engineer decides is helpful when the matrix is being used in a technology module. The identifier, SET#, is used to group matrices together as one set. In Figure 14, matrices 1863 to 1873 all have the set number 71. In the FLUTTER program, the user needs only to identify the aerodynamics by the SET#, i.e., 71. The program scans the user file and pulls into the program all matrices with a SET# equal to 71. In addition, program FLUTTER scans the FREQ identifier of matrices with SET# equal to 71 and uses FREQ values as reduced frequency. Once the values of the identifiers are part of the MATRIX TABLE they are available to programs such as FLUTTER, OPTX50TH, F.VEL, etc. (Figure 3), without any additional action on the part of the user. Since these identifiers are an integral part of GFAM's matrix data set, program MATRIX provides three ways to generate identifier values for the MATRIX TABLE. First, the READ WITH LIST display, Figure 9, generates the identifiers by indexing DELTA FREQ and DELTA DAMP for the matrices in the read list. Second, if the list of identifiers is used many times, a matrix of these identifiers generated previously and stored under a separate matrix number can be referenced in the FREQ/DAMP GROUP ID. Third, in MATRIX TABLE display, Figure 14, the identifiers can be generated or changed directly.

Once the user has built the MATRIX TABLE, the information need never be entered into the computer again. When the user wishes to save his file on tape, he has the option to save the MATRIX TABLE by assigning a matrix number to #FOR MAT MATRIX in Figure 13. When the matrices are written out on tape, a separate matrix is generated that saves all the identifiers, as well as the matrix numbers, that the user identified to be saved. To load the file with the data saved on tape, the user enters the MATRIX TABLE#, light pen detects /READ/or/READ & KEEP/ in Figure 10, and all the matrices together with all identifiers will be loaded into the user file. This permits the use of one

file by many part-time GFAM users with this simple data storage and retrieval system.

Flutter Programs

There are four basic flutter programs, FLUTTER, F.VEL, FLUTFEED, and OPTX50TH, to assist the engineer in performing interactive flutter analysis once the matrix data describing the mass, stiffness, aerodynamics, and CCV properties are available.

The flutter matrix equation,

$$\left[D\ (p,\ V,\ \bar{M})\right]\ \{\bar{z}\} = \{0\} \tag{3}$$

represents the linearized equations of motion for a flexible structure with unsteady aerodynamics. Program FLUTTER solves the characteristic equation

$$\left|\left[D(p,\ V,\ \bar{M})\right]\right| = 0 \tag{4}$$

for a non-dimensional root, p, when the airplane velocity, V, and the Mach number, $\bar{M}$, are specified. If the air density replaces the Mach number as an independent variable, solutions are made at different $\bar{M}$, but the aerodynamic matrices are not adjusted for Mach number effects. Equation (3) was formed by substituting $\{z\} = \{\bar{z}\}\ e^{pt^*}$ into the equations of motion, where $t^* = \frac{tV}{\bar{c}}$, $\bar{c}$ is the characteristic length, usually the semi-chord of the airplane, and $\{\bar{z}\}$, the eigenvector, is proportional to the initial conditions required to excite this mode only. The eigenvalue, p, has the form, $p = \gamma + ik$, where $i = \sqrt{-1}$, k is the reduced frequency, and γ is a non-dimensional real part indicating the mode stability - stable if γ is negative, unstable if γ is positive, and neutrally stable (flutter point) if γ is zero. The exponential form of stability or instability results from $\{z\} = \{\bar{z}\}\ e^{\gamma t^*}\ e^{ikt^*}$.

Another form of solution to the equations of motion is the standard k-method, where g is the structural damping required for neutral stability. The relationship $g = 2\gamma$ provides the representation of p-type solutions of equation (3) in the g format of the standard k-method solution.

FLUTTER computes f-V-g (frequency, velocity, damping) as shown in Figure 15 for any in-flight mode. The user, however, controls the computations through the OUTPUT display so that only the modes and velocities of interest are computed. Damping versus velocity for each mode is plotted in the OUTPUT display as shown in Figure 16. The user can purge roots from the solution /POUT/, review purged roots /ROLL/, store solution roots of interest /STORE/, checkpoint the computations up to that time /CKPT/, view past solutions numerically /PASTSOL/, and select the best input trials for new root computations from the past solutions /TRIAL/.

Program F.VEL directly computes the flutter speed (damping = 0) and the associated frequency.

Program FLUTFEED computes flutter roots when CCV and autopilot equations are added to the basic flutter equation. The data for the transfer functions and matrix numbers of the associated sensor matrices and the force distribution matrix are stored under one matrix number. Figure 17 is the INPUT display, where the transfer function data may be entered together with matrix information for mass, stiffness, etc., if this is not already available through an FT05 member data set. If the transfer functions need to be changed, the engineer enters the index of the matrix to be modified and light pen detects /MODIFY/. The display shown in Figure 18 gives the transfer function data in the form of polynomials,

$$\prod_{i=1}^{4} \left(\frac{a_0 + a_1 s + a_2 s^2}{b_0 + b_1 s + b_2 s^2} \right)_i \tag{5}$$

where s is the Laplace operator. If elements of a force distribution matrix require new values, the user need not transfer out of FLUTFEED and enter MATRIX, but simply light pen detect /MODIFY DELTA/, and, as shown in Figure 19, change the matrix elements.

FLUTFEED OUTPUT display is similar to FLUTTER, except for additional displays with which the engineer determines the phase and gain between a control surface and a sensor which satisfies a prescribed damping and frequency requirement.

Finally, program OPTX50TH in GFAM resizes the structure to satisfy the flutter constraints and to keep the weight increase at a minimum. The program assumes the mass and stiffness matrices to have the form

$$[M] = [M_o] + [\Delta M_i (\beta_i)] \tag{6}$$

$$[K] = [K_o] + [\Delta K_i (\beta_i)] \tag{7}$$

where $[M_o]$ and $[K_o]$ are the base mass and stiffness matrices, and $[\Delta M_i]$ and $[\Delta K_i]$ are delta mass and stiffness matrices associated with the i^{th} design variable β_i. The delta stiffness matrix is a function of β_i up to the third power, while the delta mass matrix is only linear in β_i.

VEHICLE DESIGN FOR FLUTTER

General

When a design is deficient in satisfying the flutter requirements, the flutter analyst in conjunction with representatives from other disciplines defines candidate areas for design changes. The magnitude of design changes

required to satisfy flutter constraints is determined by analysis, and the design with flutter changes is recycled through stress and possibly loads, as shown in Figure 20. The loads-stress-flutter cycle is an integral part of preliminary design and project design. The simplified structural models for preliminary design permit much shorter analysis cycle time than is possible in project design.

Structural Resizing For Flutter

The general analysis flow for structural resizing and/or ballasting of a flutter deficient design for minimum weight addition is shown in Figure 21. An overview of GFAM's role during this task follows:

Aerodynamic matrices $[\hat{A}]$ are usually available from the baseline flutter analysis. The computation of $[\hat{A}]$ is presently and, in the forseeable future, will remain a batch function. However, the generation of the batch program input data may become in the near future a graphics function in GFAM.

Transformation matrices, $[D_x]$ and $[D_z]$, used to transform structural deflections into local angles of attack associated with the aerodynamic loads points are presently generated in batch and are part of the aerodynamic matrices $[A]$. Because the baseline flutter analysis must precede any resizing task, the goal is to make generation of $[D_x]$ and $[D_z]$ a GFAM task to reduce the overall elapsed time.

Structural model data generation in GFAM at present is restricted to a simple structural model representation. It includes the formulation of the connecting matrix $[CM]$ and the diagonal compliance matrix $[C]$, which are used to form the stiffness matrix,

$$[K] = [CM] * [C]^{-1} * [CM]^T \tag{8}$$

Equation (8) is computationally efficient when the sparseness of $[CM]$ and the diagonality of $[C]$ are taken into account in the triple matrix multiplication. Stiffness derivative matrices required by the structural optimization program are readily generated from equation (8). Program COMPLY in GFAM computes the diagonal compliance matrix from physical properties of the structure.

When the stiffness matrix cannot be represented by equation (8), the stiffness matrix computation must be performed in batch using NASTRAN. A NASTRAN pre-processor using the scope interface is currently under study.

There are presently no program modules to assist the engineer in forming the mass matrix from distributive data. However, the mass matrix is often formulated as

$$[M] = [DM]^T * [ME] * [DM] \tag{9}$$

where $[ME]$ is a diagonal elemental mass matrix. This formulation offers the required flexibility to generate design variable mass derivative matrices.

Vibration analysis is usually preceded by a static reduction of the stiffness matrix (equation (1)) and a Guyan reduction of the mass matrix (equation (2)) to reduce the problem size down to 100 degrees of freedom for GFAM or down to 130 to 200 degrees of freedom for batch processing using FAMAS. If the matrices are a function of both static reduction (equation (1)) and vibration modes, then updating the matrices for these two effects is a necessary part of the structural resizing procedure as the structure mass and stiffness characteristics are being modified.

Baseline flutter f-V-g curves are usually computed in batch. When the design is altered and the engineer wants to determine the effect of the design change on some f-V-g curves, program FLUTTER in GFAM solves equation (4) for p and plots the roots as a function of velocity. The program tracks the solution of a single mode through the velocity range requested by the engineer.

Structural resizing occurs in two parts. First, is the initial structural resizing to satisfy all flutter requirements. Second, is the resizing for minimum weight while explicitly satisfying the flutter requirements and not violating strength requirements. The engineer performs both resizings using program OPTX50TH in GFAM. Details of the first resizing may be found in Chapter 8 of Reference 1, while details of the optimization are reported in References 2 and 3.

Active Control Technology

The current application of ACT methods is maneuver loads relief, gust loads alleviation, fatigue life increase, and ride quality control. The engineer synthesizes a feedback or feedforward control system by prescribing in-flight damping for certain modes as a function of velocity, as shown in Figure 21, and/or flight regime.

The prescribed damping curves result from studies in which the sensitivities of gust loads, etc. to changes of the in-flight damping of different modes are determined. Different combinations of control surfaces and sensor locations are evaluated with respect to the prescribed damping curves. The control synthesis process is optimized to derive the greatest benefits to the ACT goals in terms of matching the prescribed damping curves. The basic tool in this evaluation is program FLUTFEED in GFAM. Its main task is the computing of exact gain/phase relationships required between a control surface and a sensor for the prescribed damping and frequency of a given mode.

PRELIMINARY DESIGN EFFORT

A typical preliminary design effort involves the formulation of a structural model using a simple Russell Beam or 2-dimensional (2-D) representation with 300 or less structural elements. Typically, the preliminary design group initiates a new design from which the stress group sizes the proposed aircraft configuration using strength criteria. A structural model is

formulated, together with the aerodynamics and inertia matrices, and a standard vibration and flutter analysis is performed. A full flutter f-V-g plot is determined, and the design is subjected to Mach number and weight configuration variations. Typically, the design will be flutter deficient, because the sizing is primarily based on strength. The task is then to size the structure for flutter and determine the sizing changes that must be made to the design to satisfy the flutter constraints. The matrices associated with the mathematical model are loaded into the GFAM files by accessing the FAMAS tapes (disc). The engineer then defines the candidate sections of the structure which would be most effective in removing the flutter deficiencies. Because the structural model will typically have many more structural elements than needed for independent design variables, the practice has been to assemble a number of sizing elements which may be uniformly changed into specific groups. Since the best groupings of sizing elements are not known a priori, the engineer starts with design regions of arbitrary size and interrupts the solution process if the data indicates that groupings of structural elements are too coarse or too fine. The weight penalty due to stiffness requirements of the structure must be given to the group evaluating the proposed design.

An alternative to changing structural sizing to alter the design flutter characteristics is to add mass ballasting. In structural optimization, mass ballasting is simply another design variable with $[\Delta K_i (\beta_i)]$ in equation (7) equal to zero.

During the resizing task, the engineer periodically checks dynamic characteristics of the design by taking the current mass and stiffness matrices and initiating a flutter analysis to verify that the resizing analysis is tracking the most critical flutter mode(s).

PROJECT DESIGN EFFORT

In contrast to the approach taken in the preliminary design phase, project design usually requires a more elaborate structural model than the simple Russell Beam or 2-D representation and consequently the use of large structural programs such as FAMAS or NASTRAN.

At this stage, the flutter characteristics of the design and the areas of the structure that are most amenable to correcting a flutter problem are well known. However, static reduction, equation (1), and further reduction of the dynamic equation to 50 degrees of freedom using natural vibration modes make the stiffness changes in equation (7) nonlinear in the design variables β_i, as well as a function of cross terms $\beta_i \beta_j$, etc. Because of large computing costs, the engineer attempts to extract the maximum design changes before either updating the structural vibration modes or updating the stiffness matrices. GFAM keeps the optimization process highly visible and permits the engineer to incorporate experience and judgement directly into the optimization process.

The updating of either vibration modes and/or the stiffness matrices requires the transfer back to the batch computing mode for the large order matrix computations.

APPLICATIONS

GFAM applications, listed in Table 2, include flutter assessments for F-104 (Stores), L-1011, S-3A, NASA Arrow-Wing AST and preliminary design configurations. Table 2 also includes time history evaluations provided by ICSMP which are integrated in GFAM studies to include pilot flying characteristics requirements in synthesizing Control Configured Vehicle (CCV) systems.

The optimization for flutter performed during NASA AST Arrow-Wing study (Contract NAS1-12288) is an example of point design application of GFAM. Figure 23 defines the design regions for the outer section of the wing which were used to increase the flutter speed of the symmetric wing bending and torsion mode to 468 KEAS (see Figures 24 and 25). The wing sizing increments generated by GFAM's OPTX50TH program reflected sizing changes in the NASTRAN model. Because some of these sizings could not be translated directly into the physical model, the designers had to approximate the structure stiffness properties. The final NASTRAN model with the new sizings, however, produced a flutter speed of 512 KEAS (Figure 26). This indicates that further weight reduction is possible.

Some factors that determine whether GFAM will be used during a flutter analysis effort are

1) schedule constraints,
2) problem size,
3) analyst-in-the-loop requirements, and
4) batch cost comparison

GFAM computer runs are about 1.5 times the cost of the same task in batch. However, this may be compensated directly by showing that using GFAM will reduce the numerical effort by 35%. The dollar value of meeting schedule constraints is a variable that management determines on an individual basis.

CONCLUDING REMARKS

Management responsible for flutter assessment and design modifications to satisfy flutter constraints views GFAM as an important computing system in fulfilling its charter in the design process. Successful GFAM users have been engineers with considerable interactive computing console time as well as engineers whose first interactive system exposure was GFAM. Cost, scope accessibility, and overall computer reliability are the primary factors affecting GFAM performance from the user viewpoint.

A true beneficiery of GFAM's architecture is methods development, which include Incremented Flutter Analysis (Reference 4), structural optimization

for flutter constraints (References 2 and 3), Flutter Modules Contract NAS1-12121 (Reference 1), Control Configured Vehicle - State Space model (Reference 5), and Control Configured Vehicle - Flutter model (IRAD effort).

GFAM is presently approaching the near-term goals defined by interactive flutter analysis requirements. Expansion to data preparation of large batch programs such as NASTRAN, aerodynamics programs, etc., and integration of the CADAM data base to go directly from loft data to flutter analysis with minimum data handling by the engineer, is presently under study.

ACKNOWLEDGEMENT

The author wishes to express his gratitude to Judy Skender, of Scientific Computing Division, for the successful development of GFAM's software, especially program MATRIX, and for many valuable discussions during the course of this work. Many thanks are also due Jim McIntosh, also of Scientific Computing Division, for his general assistance, in particular with the development of GFAM's control program.

REFERENCES

1. O'Connell, R. F., Hassig, H. J., and Radovcich, N. A.: Study of Flutter Related Computational Procedures for Minimum Weight Structural Sizing of Advanced Aircraft. NASA CR-2607, 1975.

2. Radovcich, N. A.: Structural Optimization with Flutter Speed and Minimum Gage Constraints. Report 26405, Lockheed-California Co., Apr. 1974.

3. O'Connell, R. F., Radovcich, N. A., and Hassig, H. J.: Structural Optimization with Flutter Speed Constraints Using Maximized Step Size. AIAA paper no. 75-778, 1975.

4. O'Connell, R. F.,: Incremented Flutter Analysis. Journal of Aircraft, Vol. 11, No. 4, April 1974, pp. 236-240.

5. Davis, W. J., Chiodi, O. A., and Rambin, R. L.: Practical Application of Optimal Control Theory. AIAA paper no. 75-1030, Aug. 1975.

APPENDIX

MATRIX LANGUAGE EXAMPLE

Code	Card Number
STRT	(1)
PRNT DECK	(2)
*	(3)
* CCV Test Case Setup	(4)
* 40th order problem reduced to 20th	(5)
$	(6)
M @ 2644	(7)
K @ 3002	(8)
* VIB Analysis	(9)
MI = INV(M)	(10)
A = MI * K	(11)
ROOT = EIG(A)	(12)
V = VECT(A,ROOT)	(13)
* Generate Matrices	(14)
S = -1.	(15)
UN = UNIT(1)	(16)
NROW = NULL(1,40)	(17)
NCOL = NULL(40,1)	(18)
NC20 = NULL(20,1)	(19)
NR20 = NULL(1,20)	(20)
NUN = S * UN	(21)
* DELTA	(22)
DELT = STOR UN AT (35,1) INTO NCOL	(23)
DELA = STOR NUN AT (36,1) INTO DELT	(24)
* MU	(25)
MU = STOR UN AT (1,19) INTO NROW	(26)
* GAM	(27)
GAM = STOR UN AT (20,1) INTO NC20	(28)
* LAM	(29)
LAM = STOR UN AT (1,19) INTO NROW	(30)

```
                              Code                           Card Number
*                                                                  (31)
* GENERATE T MATRIX                                                (32)
ATR = STOR NR20 AT (39,1) INTO V                                   (33)
BT = STOR NCOL AT (1,20) INTO ATR                                  (34)
T = STOR UN AT (39,20) INTO BT                                     (35)
* GENERATE MODAL M,K,A                                             (36)
MM = T'*M*T                                                        (37)
KM = T' * K * T                                                    (38)
A @ 1864 TO 1873                                                   (39)
SET (,,9)                                                          (40)
AM = T' * A * T                                                    (41)
$END                                                               (42)
*                                                                  (43)
RNAM GAM AS 7999                                                   (44)
RNAM LAM AS 8000                                                   (45)
RNAM MM AS 8001                                                    (46)
RNAM KM AS 8002                                                    (47)
RNAM MU AS 8003                                                    (48)
RNAM AM AS 8004                                                    (49)
*                                                                  (50)
KEEP                                                               (51)
*                                                                  (52)
PRNT ROOT,V,DELA,T                                                 (53)
PRNT 7999 TO 8013                                                  (54)
```

Card Number	Comments
(1)	Initialize variable table
(2)	Print this set of code
(6)	Turn on automatic processing
(7) - (8)	Set up variables M and K to represent matrices 2644 and 3002

Card Number	Comments
(10)	Compute inverse of M
(11)	Create matrix A = M^{-1} * K
(12)	Compute the eigenvalues of A
(13)	Compute the eigenvectors of A corresponding to the eigenvalues computed in card 12
(16)	Create a 1 by 1 unit matrix
(17) - (20)	Create matrices of zeroes
(21)	Create a 1 by 1 matrix with value -1
(23) - (24)	Create a 10 by 1 matrix, DELA, which is zero everywhere except DELA(35,1) = 1 and DELA(36,1) = -1.
(26)	Create a 1 by 40 matrix, MU, which is zero everywhere except MU(1,19) = 1
(28)	Create a 20 x 1 matrix, GAM, which is zero everywhere except GAM (20,1) = 1
(30)	Create a 1 by 40 matrix, LAM, which is zero everywhere except GAM(20,1) = 1
(33) - (35)	Create a 40 by 40 matrix which is the eigenvalue matrix V modified as follows: T(39,J) = 0 for j = 1,19 T(I,20) = 0 for I = 1,19 and for I = 21,40 T(39,20) = 1
(37) - (38)	Create matrices MM and KM by performing pre- and post-multiply operations on the matrix T
(39)	Set up the multiple-entry variable A to represent matrices 1864 to 1873
(40)	Set the set number in the Input Display to card 9
(41)	Create the matrix AM by performing a pre- and post-multiply operation on matrix T.
(42)	Turn off the automatic processing
(44) - (49)	Reassign matrices GAM,LAM,MM,KM,MJ, and AM with matrix numbers 7999 to 8013.
(51)	Keep as permanent matrices all temporary matrices with matrix numbers in the range 1000 to 9999, i.e., matrices 7999 to 8013
(53) - (54)	Print matrices ROOT,V,DELA,T and 7999 to 8013

TABLE 1. GFAM HARDWARE AND CAPABILITY SUMMARY

Terminal Requirements:	Vector graphics scope, IBM 2250 or Vector General with NOVA for interface
Computer Resources:	IBM 360-91, dedicated core, 126K bytes (31K single precision words), shared CPU with batch, CADAM, etc.
Data Base:	Matrix data stored in FAMAS system Special card input list - card image data sets
Problem size constraints:	
o Matrix module	Small order matrices 100 x 100 Large order matrices 300 x 300 (under development)
o Flutter modules	50 x 50 complex flutter matrix
Operational Data:	
o Wait times at scope	100 x 100 eigenvalue problem — 90 seconds 50 x 50 flutter matrix root extraction — 15 seconds 50 x 50 optimization matrix root extraction — 20 seconds 100 x 100 matrix inversion (real) — 40 seconds

TABLE 2.- GFAM APPLICATIONS

NR: Not Required

MODELS	GFAM										ICSMP	
	Structure Model	Aerodynamics Model	Aero-Structure Interface	Guyan Reduction	Vibration	Modalization	Flutter	Structural Opt. For Flutter	CCV	Parametric Variations	Standard	CCV
F-104 Stores	20	X		NR	X	NR	20			X		
NASA Arrow-Wing							20/50	1		X		
ASW (S-3A)							50	2		X		
L-1011 - Full Size Model							20/50		X	X		
Preliminary Design Fighter	38			NR	X	NR	38	1		X		
L-1011 State Space Model					80				X			
L-1011 Model A	67		X	X	40	20	20/40			X		
L-1011 Demonstration Model	50			NR	X	NR	50	2		X		
L-1011 (Rigid) Equations of Motion Model	Number of structural elements				Problem order		Problem order	Number of simultaneous flutter constraints			X	X

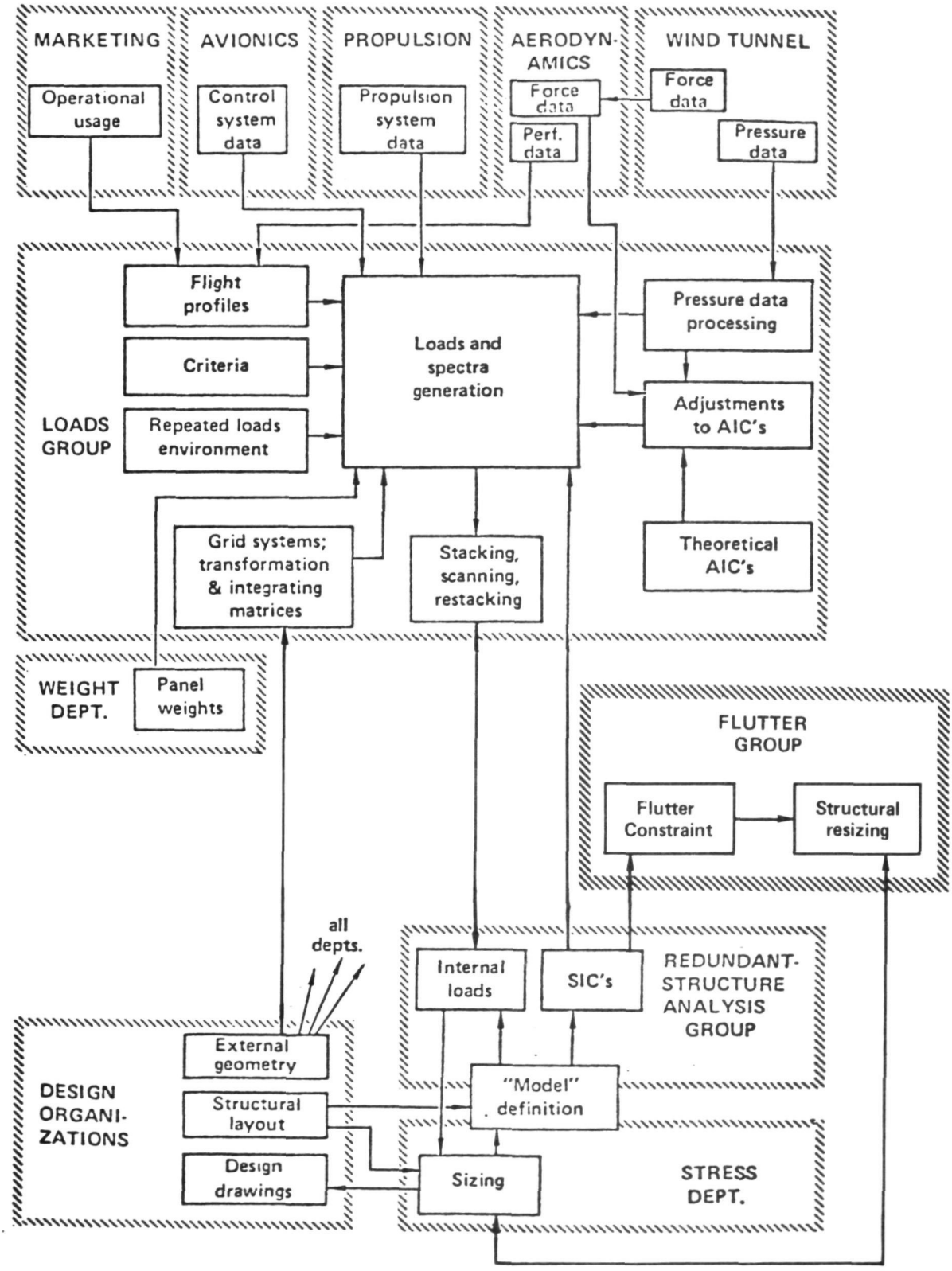

Figure 1.- Organizational interfaces and flow of data.

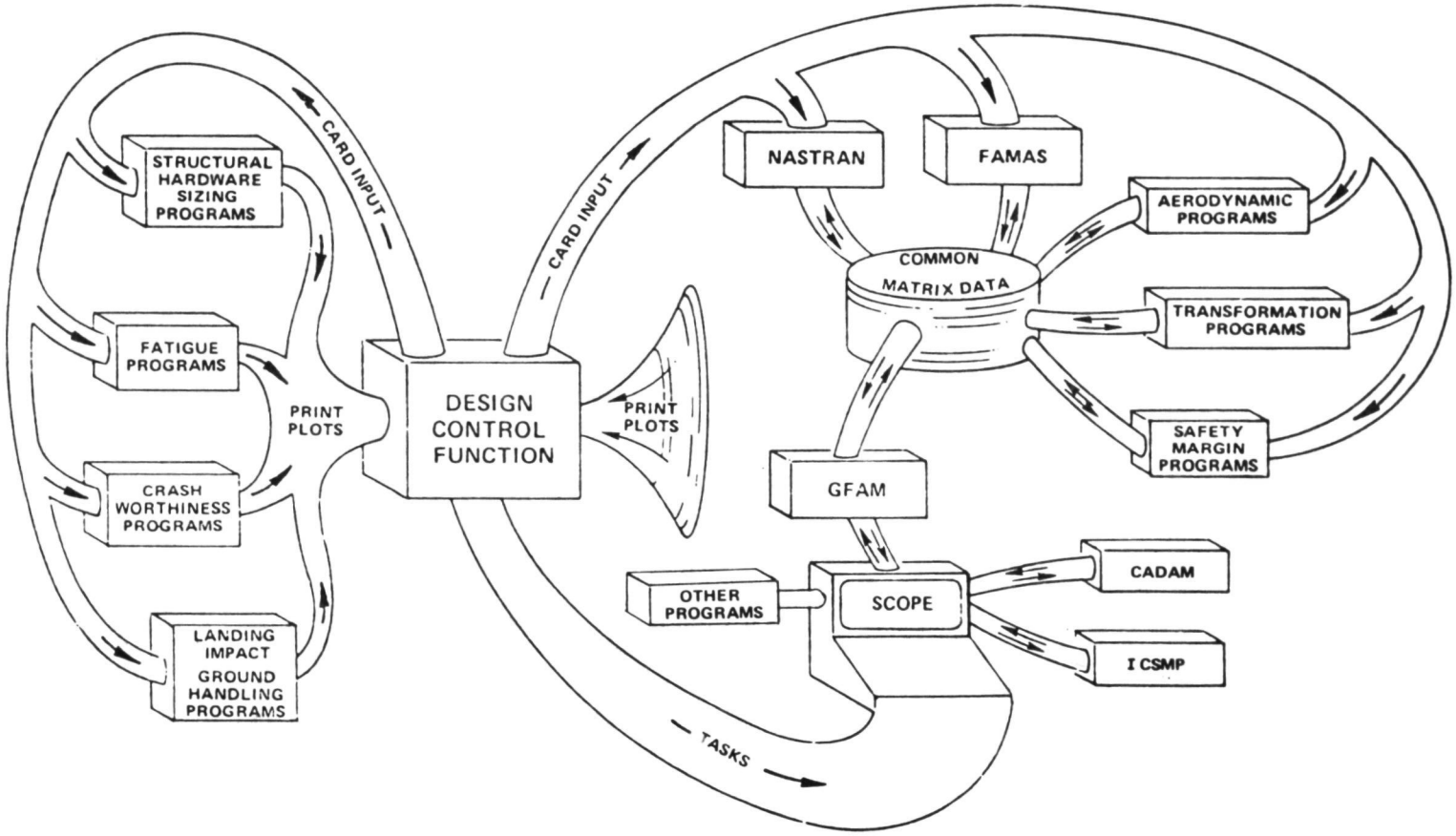

Figure 2.- Point design - Structures / Aeromechanics.

INTERACTIVE GRAPHICS COMPUTING SYSTEM · COMPATIBLE WITH FAMAS MATRIX DATA STORAGE SYSTEM

GFAM/MATRIX

- INPUT/OUTPUT/PRINT. FAMAS MATRIX DATA FROM TAPES, DISCS, AND CARDS. DATA MANAGEMENT SYSTEM
- PLOTTING/DATA SCREENING/TEST · THEORETICAL MATCHING
- MATRIX ALGEBRA. ADD/SUBTRACT, MULTIPLY, TRANSPOSE, INVERSE, EIGENVECTOR, EIGENVALUES
- MATRIX OPERATIONS. EXTRACT, STORE, UNIT AND NULL MATRIX GENERATORS.
- MATRIX ALGEBRA LANGUAGE. A = B + C, A = B' * C * B

  A = INV(B)...

  INTERACTIVE COMPILER/CHECKER/MATRIX DATA EDITOR

  INLINE CODING, GO TO ....

GFAM/TECHNOLOGY MODULES

FLUTTER · SOLVES THE FLUTTER EQUATION FOR DAMPING AND FREQ. TRACKS A GIVEN MODE. f-V-g PLOTS ON SCOPE

FLUTTER VEL · COMPUTES FLUTTER VELOCITY DIRECTLY

OPTX50TH · STRUCTURAL OPTIMIZATION FOR FLUTTER VELOCITY AND MINIMUM GAGE CONSTRAINTS

FLUTTER FEED · COMPUTES FEEDBACK AMPLITUDE AND PHASE VS FREQUENCY REQUIRED FOR PRESCRIBED DAMPING VS VELOCITY

BLOWOUT · COMPUTES PRESSURE TIME HISTORIES, AIRPLANE DECOMPRESSION ANALYSIS

Figure 3.- Graphics Flutter Analysis Methods.

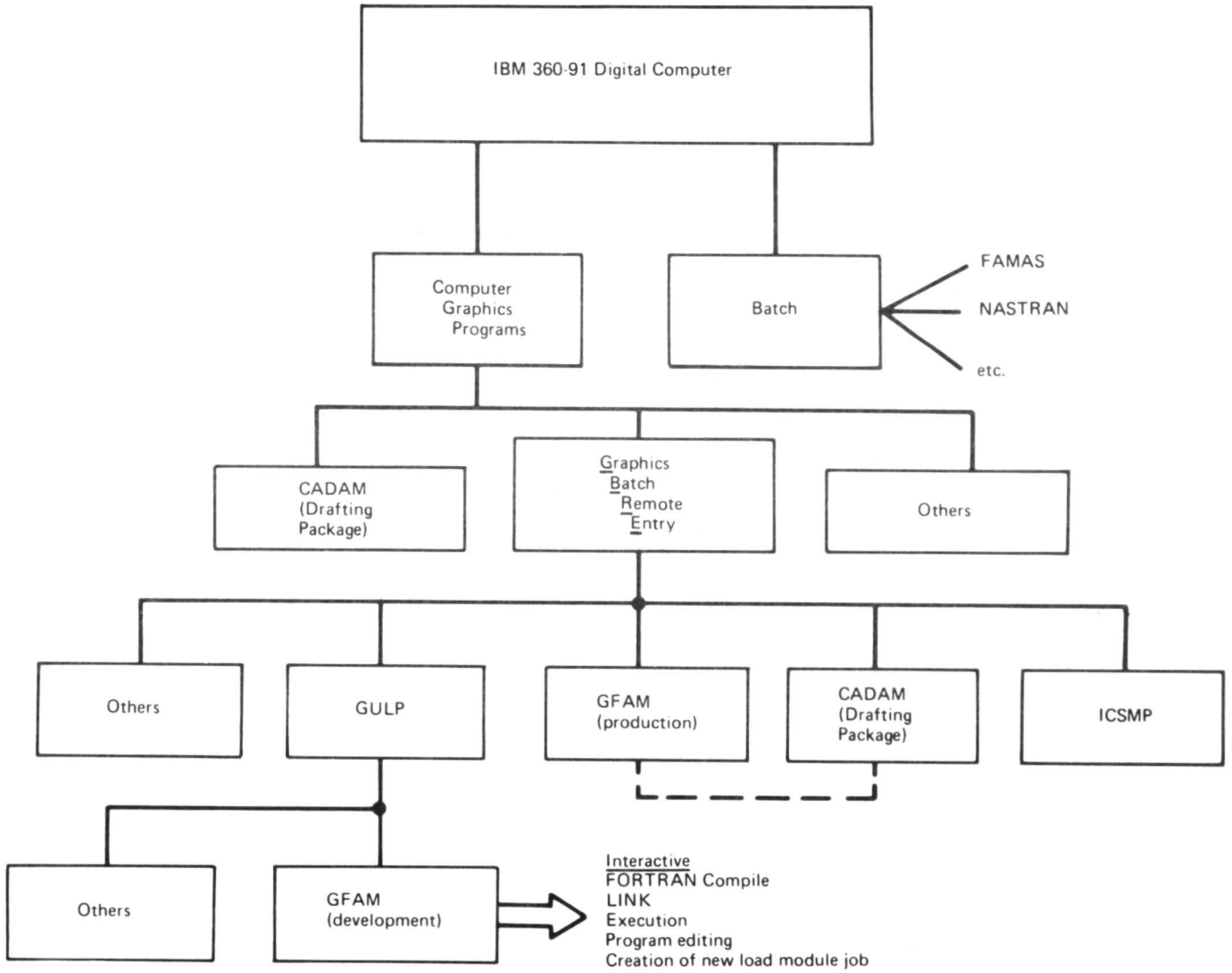

Figure 4.- GFAM in the computer tree.

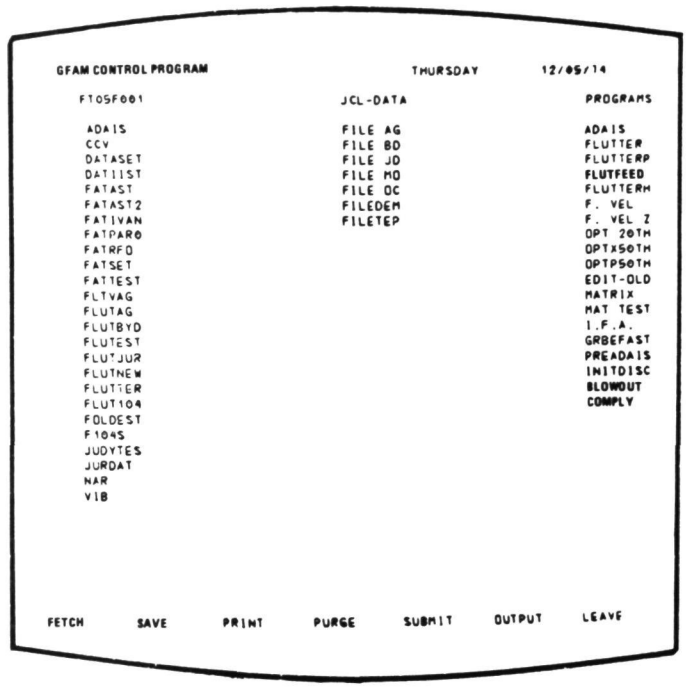

Figure 5.- GFAM control display.

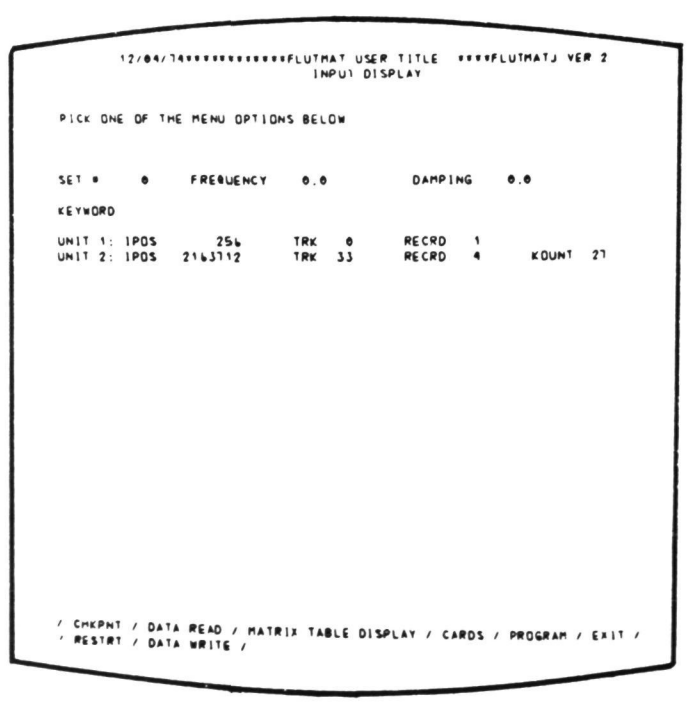

Figure 6.- MATRIX- input display.

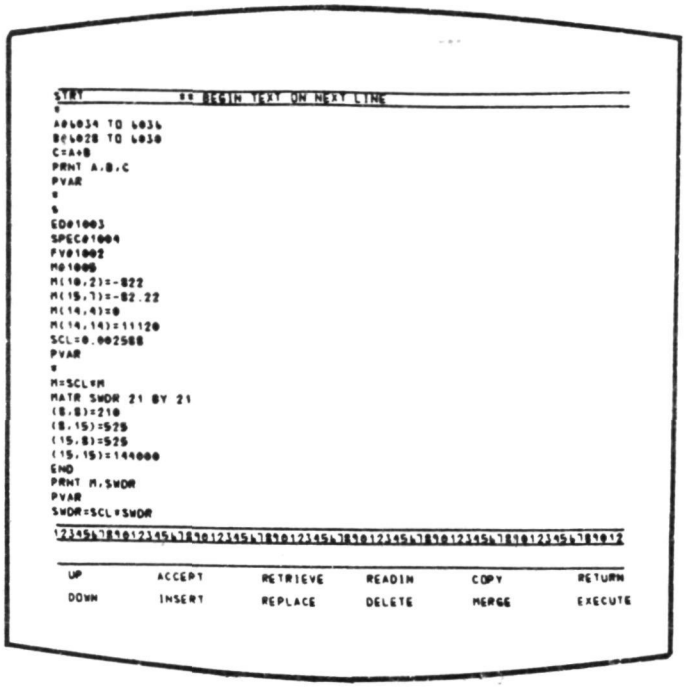

Figure 7.- MATRIX- program display.

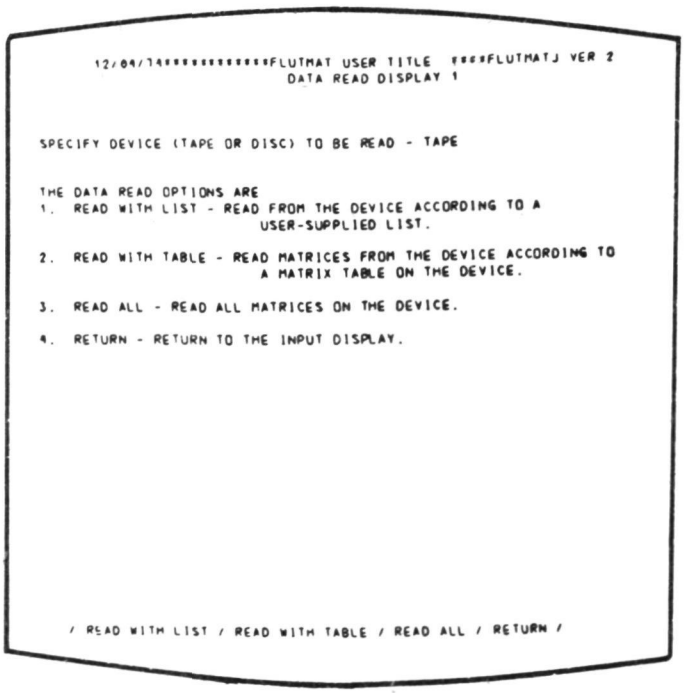

Figure 8.- MATRIX- data read display 1.

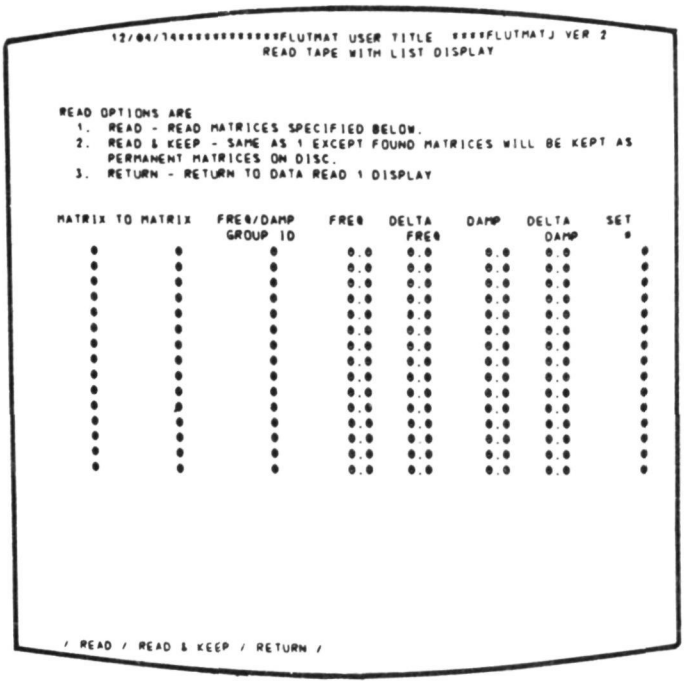

Figure 9.- MATRIX- read with list display.

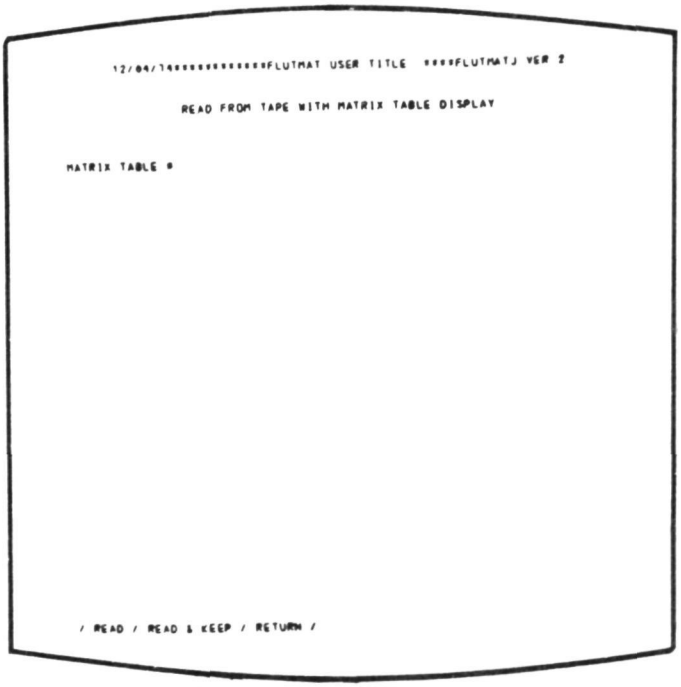

Figure 10.- MATRIX- read with matrix table display

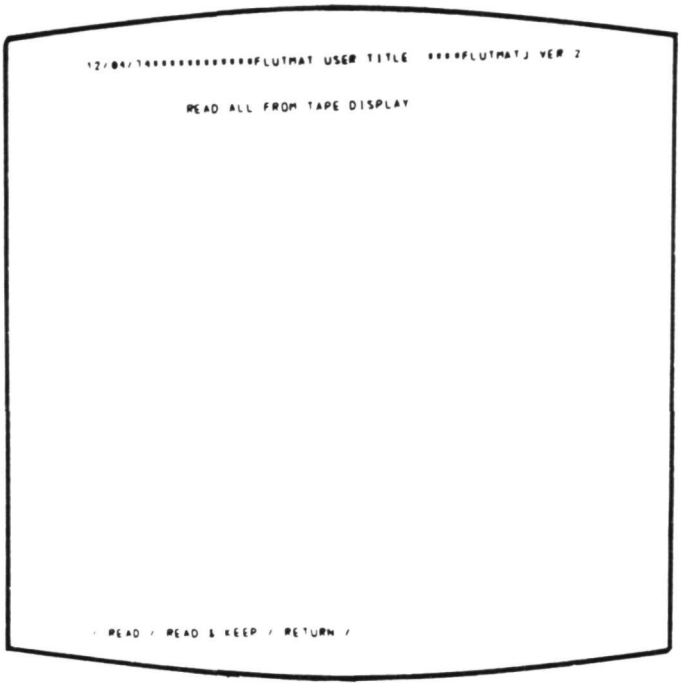

Figure 11.- MATRIX- read all display.

Figure 12.- MATRIX- read display 2.

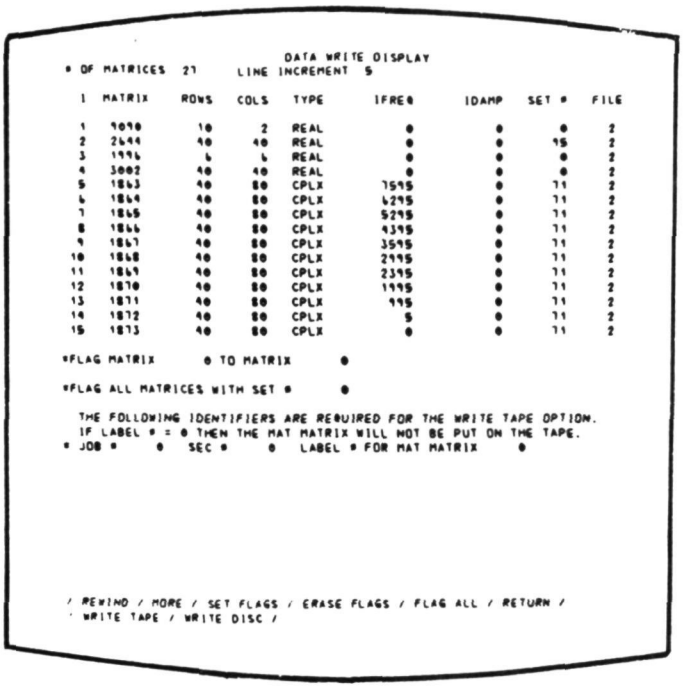

Figure 13.- MATRIX- data write display.

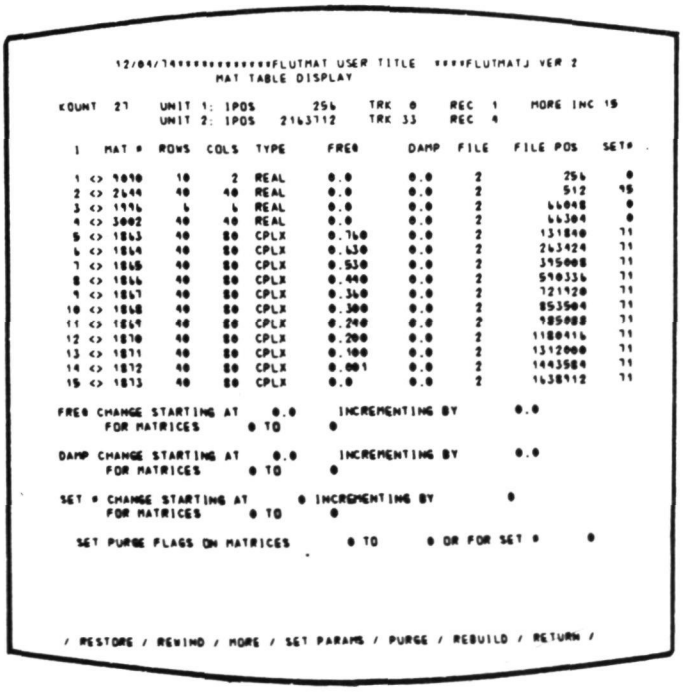

Figure 14.- MATRIX- matrix table display.

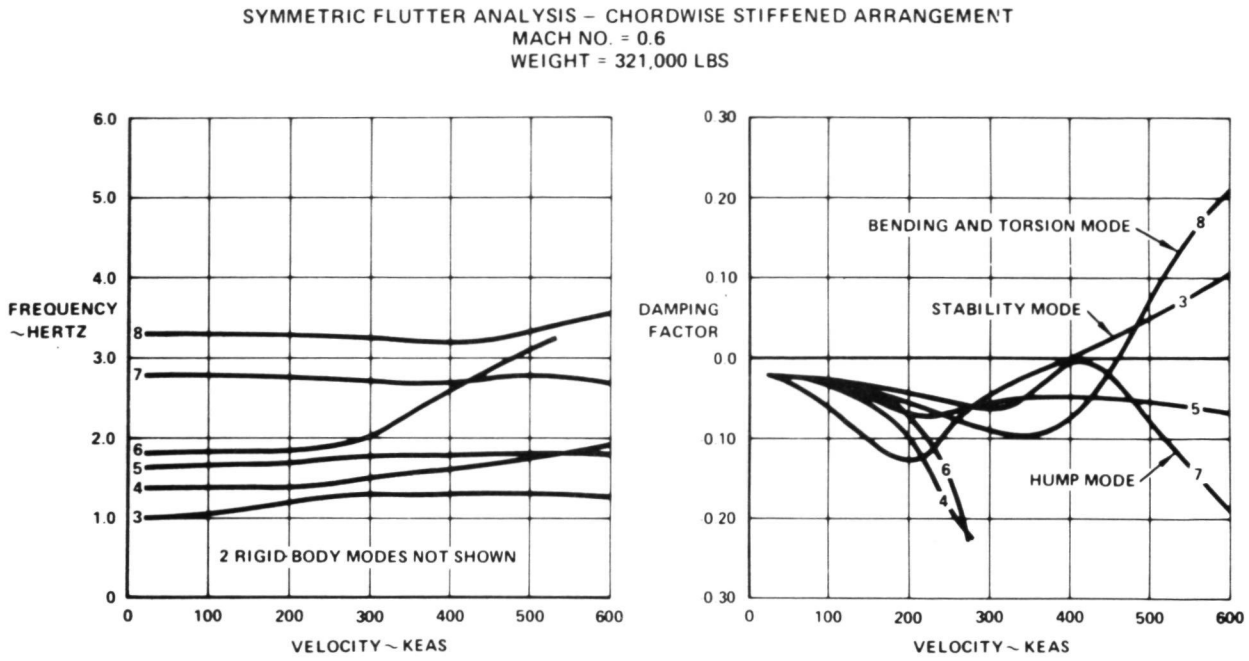

Figure 15.- Typical flutter f-V-g plots. 321 000 lb = 1427.8 kN.

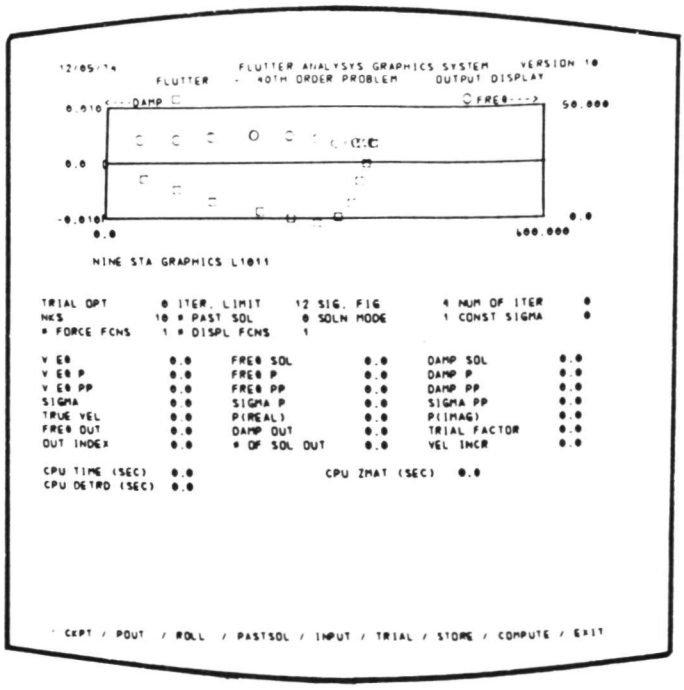

Figure 16.- FLUTTER- output display.

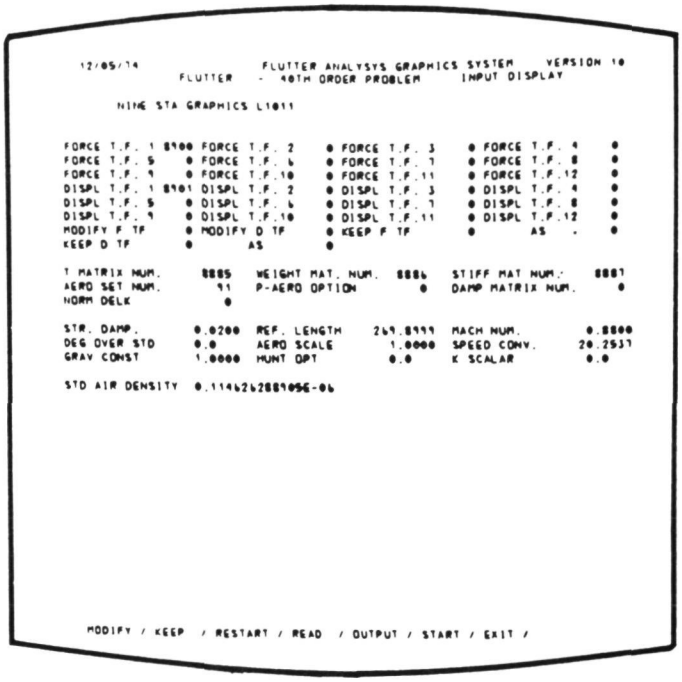

Figure 17.- FLUTFEED- input display.

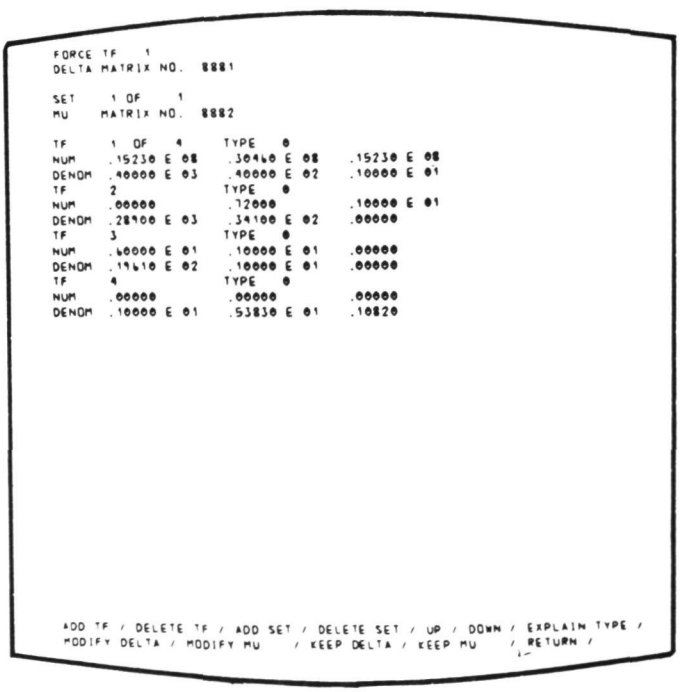

Figure 18.- FLUTFEED- transfer function display.

Figure 19.- FLUTFEED- matrix modify display.

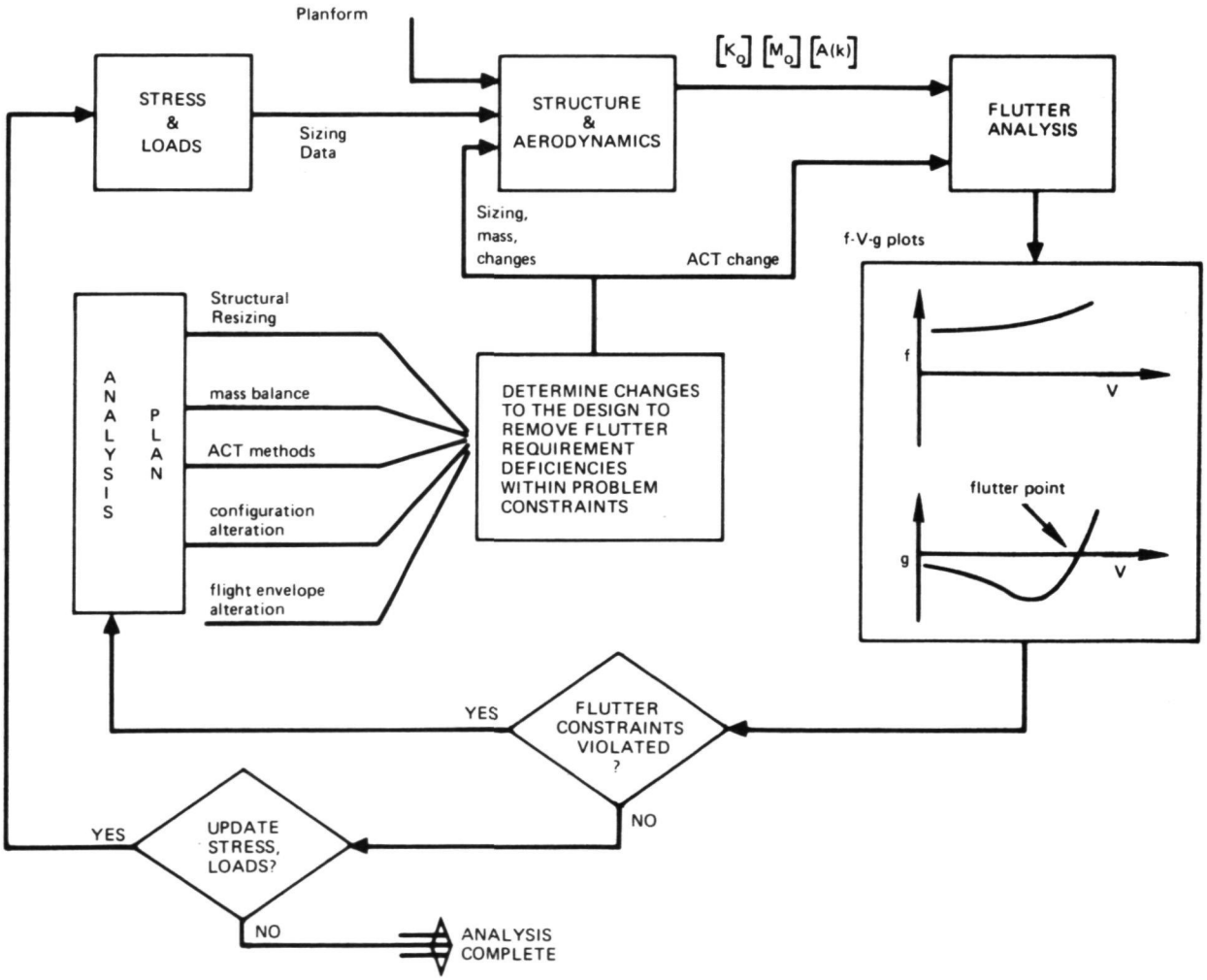

Figure 20.- Vehicle design cycle for flutter, stress, and loads.

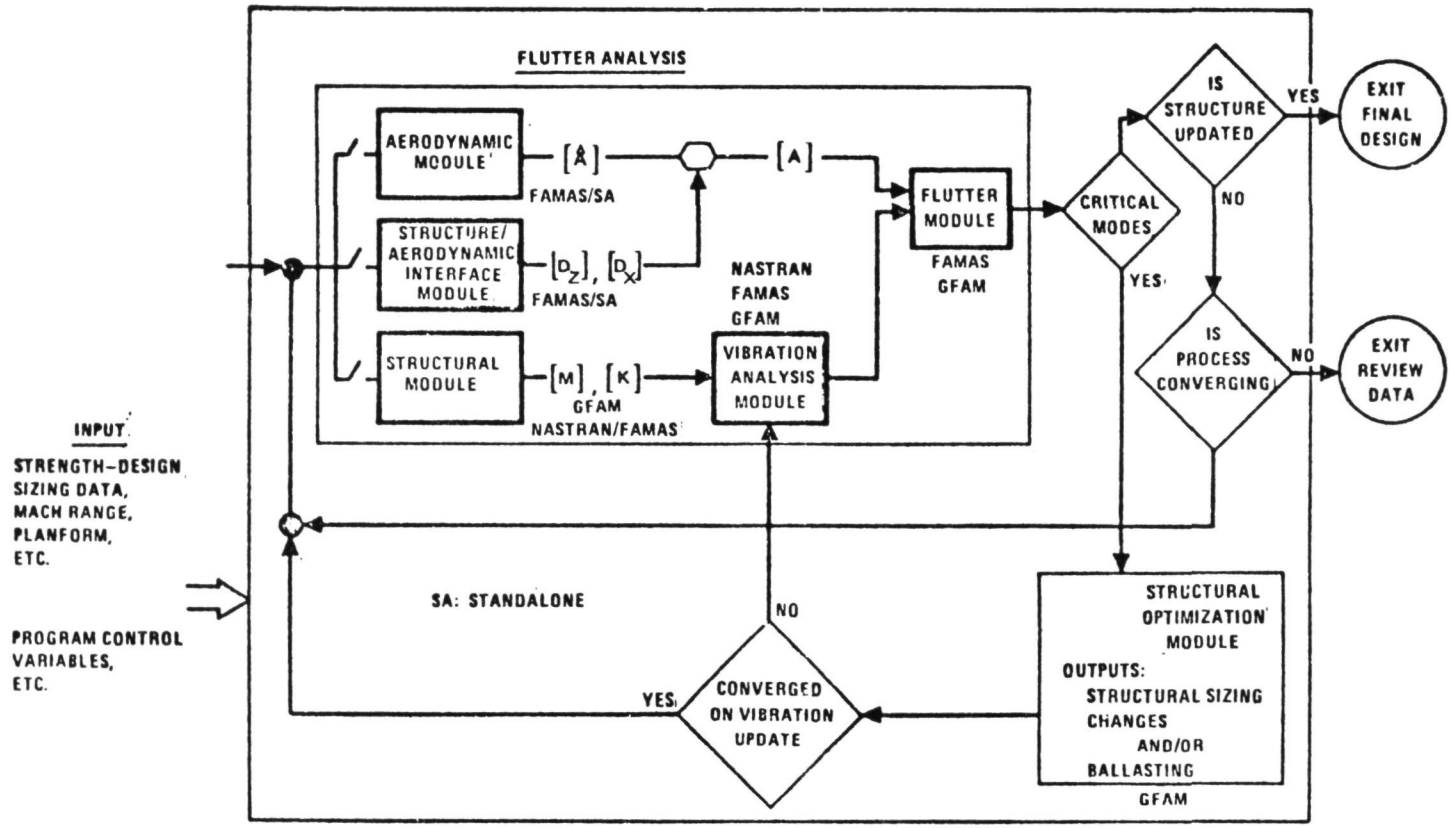

Figure 21.- Structural design for flutter.

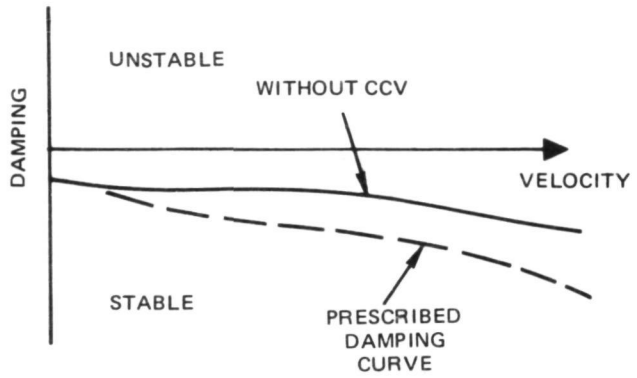

Figure 22.- Prescribed damping versus velocity.

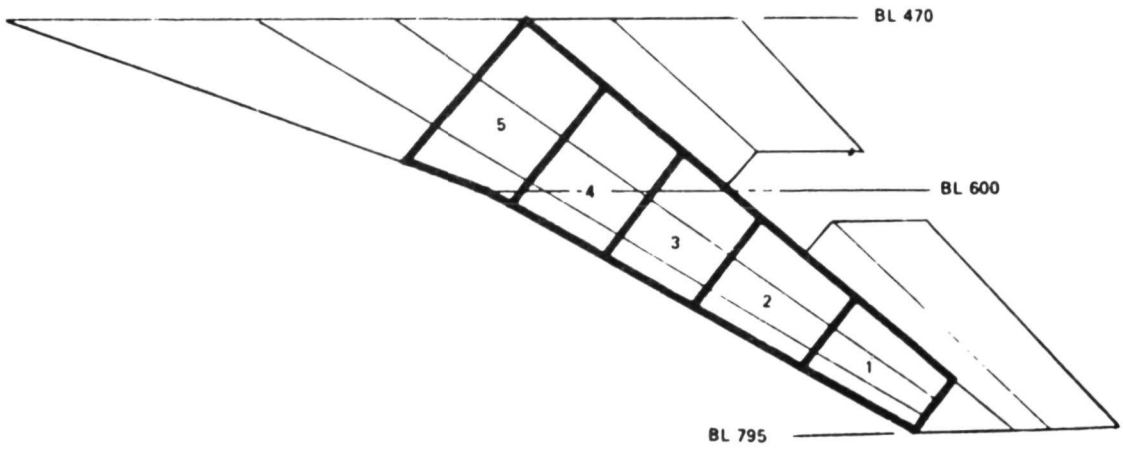

Figure 23.- Flutter optimization design regions.

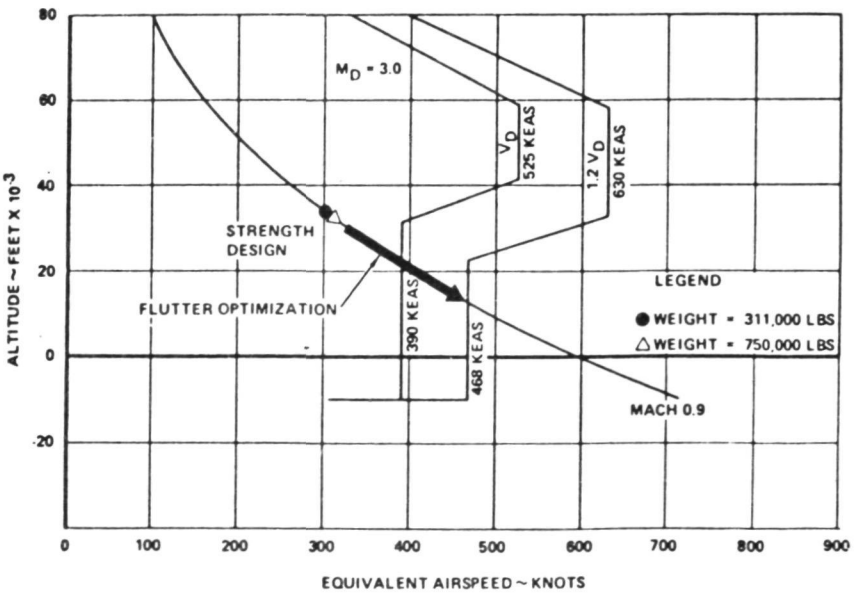

Figure 24.- Flutter speeds for symmetric bending and torsion mode. 311 000 lb = 1383.3 kN; 750 000 lb = 3336 kN.

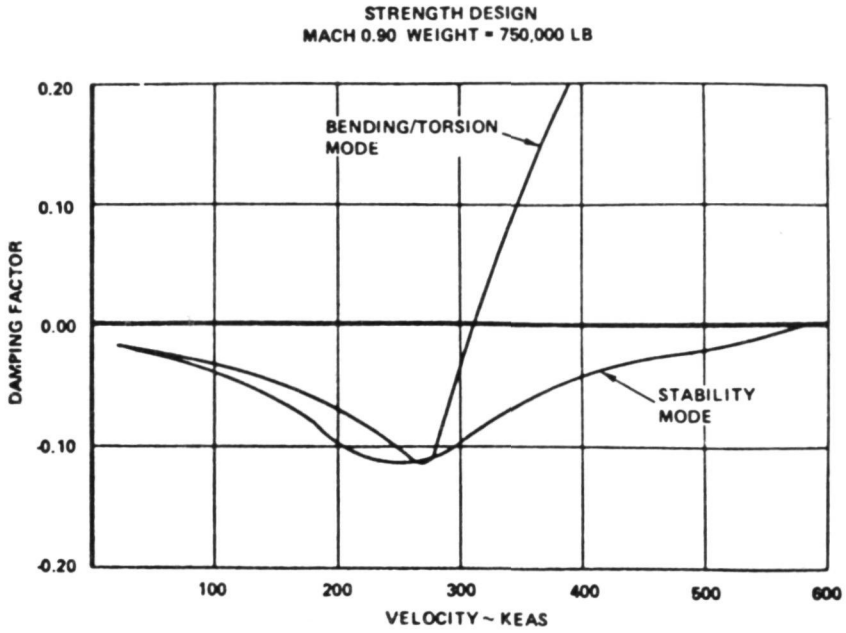

Figure 25.- Symmetric flutter analysis - Mach 0.9 - FFFP. KEAS denotes knots equivalent airspeed. 750 000 lb = 3336 kN.

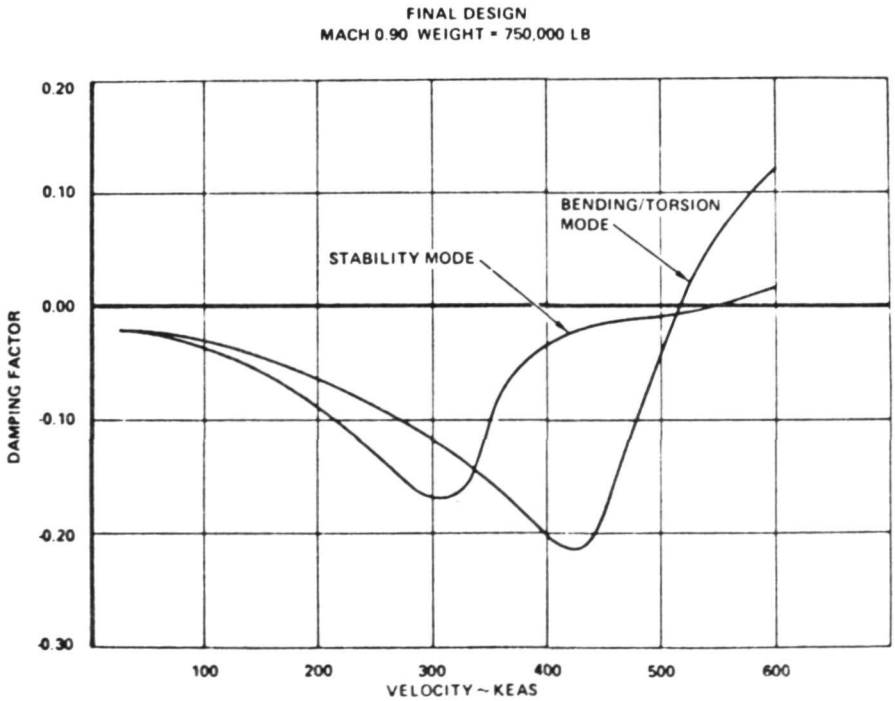

Figure 26.- Symmetric flutter analysis - Mach 0.9 - FFFP. KEAS denotes knots equivalent airspeed. 750 000 lb = 3336 kN.