NASA CR-111918

15 copies

A DESIGN STUDY FOR THE INCORPORATION OF AEROELASTIC CAPABILITY INTO NASTRAN

by

Robert L. Harder
Richard H. MacNeal
William P. Rodden

May 1971

Prepared under Contract No. NAS1-9801 by
THE MACNEAL-SCHWENDLER CORPORATION
7442 North Figueroa Street
Los Angeles, California 90041
for
NATIONAL AERONAUTICS AND SPACE ADMINISTRATION

A DESIGN STUDY FOR THE INCORPORATION OF AEROELASTIC CAPABILITY INTO NASTRAN

by

Robert L. Harder
Richard H. MacNeal
William P. Rodden

ABSTRACT

The basic computational tasks required for solution of both static and dynamic aeroelastic problems are discussed. The modifications and additions to NASTRAN that are required to add an aeroelastic capability are presented. New functional modules and rigid formats are outlined. Special emphasis is placed on flutter analysis.

TABLE OF CONTENTS

TABLE OF CONTENTS

(Cont'd.)

TABLE OF CONTENTS

(Cont'd.)

A DESIGN STUDY FOR THE INCORPORATION OF AEROELASTIC CAPABILITY INTO NASTRAN

1. INTRODUCTION

1.1 Present NASTRAN Capabilities

The NASTRAN* digital computer program for structural analysis is a finite element program with an extremely broad range of applications. Its problem solving capabilities are, at present, separated into the following twelve "rigid formats," each of which entails a predetermined sequence of calculations that are performed by "functional modules" under the control of a flexible executive system.

Static Analysis

1. (Basic) Static Analysis
2. Static Analysis with Inertia Relief
3. Static Analysis with Differential Stiffness
4. Piecewise Linear Analysis

Elastic Stability

5. Buckling

Dynamics

6. Vibration Mode Analysis
7. Direct Complex Eigenvalue Analysis
8. Direct Frequency and Random Response Analysis

* NASTRAN is a mnemonic for NASA STRuctural ANalysis program. The NASTRAN Theoretical Manual (NASA SP-221, Sept. 1970) contains a general introduction to its capabilities.

9. Direct Transient Response Analysis

10. Modal Complex Eigenvalue Analysis

11. Modal Frequency and Random Response Analysis

12. Modal Transient Response Analysis

It will be noted that dynamic response and complex eigenvalue problems may be solved either by a "direct" method or by a "modal" method. In a direct method the degrees of freedom are physical components of displacement. In a modal method the degrees of freedom are generalized modal coordinates. Like all other general purpose structural programs, NASTRAN employs the finite element approach. The structural model that is analyzed consists of "elements" such as beams, plates, and concentrated springs, that are connected together at a finite number of "gridpoints." The equilibrium equations of the model are expressed and solved in terms of the components of motion at gridpoints. The solution of large problems, with hundreds or thousands of structural elements, is emphasized in the design of NASTRAN. It is, in addition, a highly user-oriented program with a large number of convenience features including automatic load generation, plotting, and restart capabilities. It also includes provisions for including nonstructural components, such as control systems, in the structural model. At present, provision is made in NASTRAN for the solution of aeroelastic problems by means of "direct input" matrices which may be real or complex and which may be added to the structural mass, damping, or stiffness matrices and included in the solution of any of the seven dynamic rigid formats. This provision is regarded as inadequate for the following reasons:

a. It requires the use of separate computer programs for the generation of the aerodynamic input matrices, thereby causing delays and increasing the frequency of errors.

b. It places the burden of achieving geometric compatibility between the aerodynamic forces and the structure entirely on the user of the program.

c. It provides very little static aeroelastic capability and it automates none of the specialized features of aeroelastic analysis, such as the selection of control surface deflections for trimmed flight, and the plotting of V-g flutter curves.

1.2 Objectives

The purpose of the present study is to define the modifications that will be required to make NASTRAN an effective tool for aeroelastic analysis. Due to its modular character, the addition of new capability to NASTRAN is not inherently difficult. The modifications required to add a particular new capability may involve changes in existing functional modules, the generation of new functional modules, or the compilation of new rigid formats. All of these measures will be required to add aeroelastic capability.

One of the major objectives for the NASTRAN aeroelastic modification should be to provide a means for solving aeroelastic problems which has a degree of automation and a range of application comparable to that which presently exists for the solution of purely structural problems. Advanced aerodynamic configurations, such as the Space Shuttle, indicate a need for a more sophisticated approach to aeroelasticity in order to cope with the attendant structural and aerodynamic complexities. Our study of these matters has led to the following list of requirements for a general purpose aeroelastic program:

1. Broad Application
    a. Many types of structures
    b. Many aerodynamic configurations
    c. Subsonic to hypersonic flow regimes
    d. Many classes of static and dynamic analysis
    e. Compatibility with other aspects of structural analysis, such as stress analysis, and control system interaction.
2. Sophistication
    a. Ability to use advanced aerodynamic theories
    b. Accurate and efficient computational procedures
    c. Many degrees of freedom
3. Automation
    a. Minimum user effort, for both clerical and intellectual tasks
    b. Checkpoint and restart capabilities.
4. Ease of modification to include new or improved features

The requirement for broad application is important from an economic viewpoint, because it reduces the number of separate computer programs and, therefore, reduces development cost. It also eliminates the time that the user would spend in learning to apply several separate programs, and in preparing several different sets of input data.

The qualities listed under Sophistication and Automation are the qualities that are usually associated with large user-oriented computer programs, and they are now largely taken for granted in structural analysis, if not yet in aeroelastic analysis. Ease of modification to include new or improved features is particularly important due to the long length of the development cycle for large scale computer programs, and the rapidity of recent technical progress.

The heart of an aeroelastic analysis is the theory used to calculate aerodynamic forces, and the requirements listed above for the complete computer program can also, very largely, be applied to the selection of aerodynamic theories. Since no existing theory has the desired range of application, the ability to include several different aerodynamic theories, perhaps even in the same problem, is recognized as a requirement. Advanced theories should be included, and also simple theories which are usually better from the standpoint of computational efficiency. One of the tasks undertaken in the study has been to investigate candidate unsteady aerodynamic theories with respect to their range of application, accuracy, efficiency, user convenience, and general compatibility with a finite element approach to aeroelastic analysis. The results of the investigation are reported in Appendix G and in Section 5.5.2.

1.3 Classification of Aeroelastic Problems

It is expected that aeroelastic capability will be implemented in NASTRAN by a number of different rigid formats with names such as "Divergence," "Flutter," and "Frequency Response" that correspond to different types of analysis. A first task, before discussing the proposed measures for including aeroelasticity in NASTRAN, is to classify aeroelastic problems according to type of solution. The classification will be separated, for convenience, into static aeroelastic problems, flutter, and dynamic response problems.

Static Aeroelastic Problems

A static aeroelastic problem may be defined as a problem involving the response of a flexible structure to aerodynamic loading in which terms proportional to the velocities and accelerations of the structure are assumed to be independent of time. Thus, inertia forces, if they enter at all, are assumed to be constant in time. The three common types of static aeroelastic problems are

a. Calculation of static response, including loads and stresses in the structure.

b. Calculation of stability and control derivatives, i.e., the calculation of the changes in the aerodynamic loading (and, more particularly, of the changes in its resultants) due to small changes in the motions of the vehicle and of control surface deflections.

c. Divergence, which is an idealized stability problem in which the determinant of the stiffness matrix, including both structural and aerodynamic terms, vanishes.

Each of the static aeroelastic problems may be further classified as to whether the structure is supported or free to move. If it is free to move, the inertia forces due to (steady) accelerations must be taken into account.

An additional distinction occurs in the determination of the static response of a freely moving structure, with respect to the manner in which the velocity components and the control surface deflections of the structure are determined. In some cases the analyst specifies all of the components of velocity and the control surface deflections and accepts whatever accelerations they produce. He may, on the other hand, request that some of the velocity components and control surface deflections be evaluated so as to put the forces on the vehicle into equilibrium, i.e., into a trimmed flight condition.

The general task of formulating and classifying static aeroelastic problems for solution in NASTRAN is addressed in Appendix B. It is shown there that all of the problem types mentioned above may be solved with the following three rigid formats:

-- Aeroelastic Divergence

-- Untrimmed Static Aeroelastic Response

-- Trimmed Static Aeroelastic Response

In particular, the calculation of stability and control derivatives is treated as a subcase of aeroelastic response.

Flutter

Flutter is the dynamic aeroelastic stability problem just as divergence is the static stability problem, and it is appropriately analyzed as an eigenvalue problem. It is, however, a multiple eigenvalue problem because the frequency of the oscillations and the speed and altitude at which they become unstable are all desired items of information. The unsteady aerodynamic forces are functions of reduced frequency (or Strouhal number), Mach number, air density, and perhaps of other parameters. A standard general approach to flutter analysis is to evaluate the frequency of the oscillations and their damping (or equivalently the structural damping required to make them neutrally stable) for different values of the parameters, and then to cross-plot the results to find stability boundaries in the parameter space. Several variations of the general approach are explored in Appendix F, resulting in the recommendation that two different methods of flutter-eigenvalue analysis be implemented in NASTRAN. These methods, called the k method and the p-k method, can be implemented with the same rigid formats.

The ability to have both "direct" and "modal" methods of solution, noted earlier as an existing NASTRAN capability, is also important for flutter analysis, although the modal method is usually more efficient. Separate "direct" and "modal" rigid formats are proposed for flutter analysis.

Dynamic Response

The steady state response of an aeroelastic system to sinusoidal excitation can provide the basis for the calculation of deflections and stresses due to loads caused by oscillatory vibration, random vibration, and transient excitation. The frequency response calculation requires the solution of a system of simultaneous complex algebraic equations. The transfer functions that result from the calculation can be used to compute the statistical properties of the response of linear systems to stationary random excitation. In addition, Fourier integral transform theory provides the basis for using frequency response data to compute the response of linear systems to transient excitation.

The latter ability is particularly important for aeroelastic analysis because all advanced unsteady aerodynamic theories are formulated in the frequency domain. Their conversion to the time domain in order to numerically integrate the equations of motion appears to be prohibitively awkward unless gross approximations are made. This matter is explored a bit further in Appendix D where the conclusion is reached that transient aeroelastic response calculation should be implemented by the Fourier Integral Method only, at least initially.

Dynamic aeroelastic response, therefore, results in four NASTRAN rigid formats corresponding to frequency response analysis and transient response analysis (by the Fourier Integral Method) using either a direct or a modal formulation. Random analysis is treated, as in the present NASTRAN rigid formats, as a subcase of frequency response.

1.4 The Major Tasks

The process of solving a linear aeroelastic problem with the aid of a structural analysis program is conceived as consisting of the following steps:

1. Define the properties of the structure in terms of matrices of stiffness, mass and (perhaps) damping coefficients that refer to the components of motion at structural grid points (or to generalized modal coordinates in a modal formulation).

2. Compute a matrix of aerodynamic influence coefficients, relating forces to motions, at a set of aerodynamic control points.

3. Transfer the aerodynamic influence coefficients to the structural grid points (or to modal coordinates) and combine them with the structural matrices.

4. Solve the resulting matrix equations by whatever methods are appropriate for the type of results desired, e.g., by eigenvalue extraction for a flutter analysis.

5. Recover stresses, aerodynamic forces, etc., by means of equations relating these quantities to the degrees of freedom evaluated in step 4.

In the above list of steps,

-- Step 1 requires no modification of the structural analysis program;

-- Steps 2 and 3 are wholly new;

-- Step 4 requires some new solution techniques to accommodate special features of aeroelastic analysis;

-- Step 5 requires modification of the program only to the extent of adding routines to recover aerodynamic quantities.

Major emphasis has been placed, during the study, on formulating an approach for the implementation of steps 2 and 3. The foremost objectives have been to provide a broad range of options with respect to configuration parameters and available theories, and to minimize user effort. Step 2 has been divided into two tasks: first, subdivision of aerodynamic surfaces into subregions (called aerodynamic elements) and the definition of their geometric properties; and second, the calculation of aerodynamic matrices for the selected aerodynamic theories. By restricting the range of permitted aerodynamic element shapes and orientations, the design of a module to perform the first task has been made independent of the second task. The work required to add new aerodynamic theories will, thereby, be minimized for theories that conform geometrically to the restrictions that have been imposed. For this reason considerable attention has been paid to the geometric requirements of current aerodynamic theories, see Section 5.2.

Step 3 has been divided into four tasks as follows, each performed by a different functional module.

1. Generate a transformation matrix that linearly relates displacements at aerodynamic control points to displacements at structural grid points. Calculation of the transformation matrix involves the use of interpolation procedures, for which a variety of options are proposed, see Section 5.4 and Appendix E.

2. Compute aerodynamic matrices as seen at the structural gridpoints (or modal coordinates) for a selected list of aerodynamic parameter sets (reduced frequency and Mach number), using the transformation matrix from task 1 of step 3, the aerodynamic matrices generated in step 2, and, if needed, the matrix of structural eigenvectors.

3. Interpolate the aerodynamic matrices to other values of the aerodynamic parameters; see Section 5.7 and Appendix J.

4. Combine the aerodynamic and structural matrices in the functional module that generates each particular type of solution.

The accomplishment of step 4, solution of matrix equations, will require new routines for flutter analysis, divergence and static aeroelastic response. In our judgement, the existing NASTRAN procedures are not adequate to accommodate the special features of these analysis types, although in all cases the new routines resemble existing routines.

The effort required to formulate computational procedures for the above tasks has resulted in some innovations that may be worthy of consideration in other contexts. These include the use of elastic plates for surface interpolation (Appendices E and J), a formulation of the static aeroelastic problem that preserves rigid body accelerations as degrees of freedom (Appendix B), and a procedure for minimizing the number of cases required to obtain a flutter boundary (Appendix F). In addition, a means for extending the subsonic aerodynamic theory known as the Doublet-Lattice Method to supersonic speeds has been explored. This result will be reported in the near future.

1.5 Summary

The remaining sections of the main text present the content of the proposed NASTRAN modifications. The text is organized so that it can be used both as a specification, and as a preliminary design description, of the proposed modifications. Section 2 presents simplified flow diagrams of the proposed computational procedures for aeroelastic analysis and lists the new Rigid Formats and Functional Modules. Sections 3 and 4 summarize the formal matrix algebra used in accomplishing each of the steps in the calculation. Section 5 includes descriptions of the new functional modules and Section 6 describes the new types of input data, the new types of output data, and the options that will be available to the user. Special topics are discussed in the appendices.

The degree of detail included in the functional module descriptions varies from complete mathematical designs of the algorithms to brief sketches. More detail is included for those modules required in dynamic aeroelastic analysis than for those used in static analysis, and flutter analysis receives more emphasis than dynamic response analysis. The relative emphasis reflects the priorities that have been set by the Technical Monitor.

2. PROPOSED PROBLEM FLOW

A computer program consists of a set of instructions that are organized to solve a variety of mathematical problems. The problems themselves are described by parameter values supplied by the user of the program. The computational task is subdivided into steps that are performed by discrete blocks of code called "Functional Modules." In NASTRAN, calls to functional modules are controlled by the "Executive System." A number of sequences of module calls, called "Rigid Formats," are stored in the program. Each rigid format is designed to solve a particular class of problems such as "Basic Static Analysis," "Buckling," "Modal Transient Response Analysis," etc.

The subdivision of computational steps into Functional Modules is arbitrary at least to the degree that the dividing points that they insert in the sequence of calculations can be placed arbitrarily. In some cases it is also possible to rearrange the order of calculations. Computational efficiency (and common sense) dictates the insertion of division points at places where the data transmitted across the interfaces are minimal (NASTRAN requires that all data blocks transferred from one module to another be first placed in peripheral storage). Other factors, such as the requirement that restarts can only be scheduled at the beginning of a module, or that certain calculations are useful in several different types of problems, influence the subdivision into Functional Modules.

The proposed subdivision of aeroelastic analysis into NASTRAN Functional Modules and Rigid Formats is summarized in the following exhibits:

Table 2-1: Simplified Flow Diagram for Dynamic Aeroelastic Analysis

Table 2-1 presents a sequence of functional module calls for all forms of dynamic aeroelastic analysis. It also provides a very brief statement of the operations performed by each module and indicates whether the module is an existing module or a new module.

Table 2-2: Simplified Flow Diagram for Static Aeroelastic Analysis

Table 2-2 is analogous to Table 2-1.

Table 2-3: List of New Functional Modules

Table 2-3 lists the names of the new functional modules contained in Table 2-1 and 2-2 in three classifications:

A. Modules used in both static and dynamic analysis.

B. Modules used only in dynamic analysis.

C. Modules used only in static analysis.

Table 2-4: List of New Rigid Formats

Table 2-4 lists nine proposed new rigid formats for the solution of aeroelastic problems. Each rigid format corresponds to a particular type of aeroelastic analysis and, in the case of dynamic aeroelasticity, to a particular method of analysis. Additional rigid formats will be required for transient analysis with aerodynamic forces represented directly in the time domain, but this topic has not been researched sufficiently. The different rigid formats correspond to alternate paths in the flow diagrams of Tables 2-1 and 2-2, or to the deletion of some modules, or, in some cases, simply to alternate paths within modules. Separate rigid formats are provided for direct and modal dynamic analyses, which is consistent with existing NASTRAN practice for structural dynamics. A separate rigid format does not exist for the calculation of stability derivatives, because this task can be accomplished with Format No. 8A, Untrimmed Static Aeroelastic Response, see Appendix B.

The proposed new functional modules are described in detail in Sections 5.1 to 5.17. The major algebraic tasks performed in the solution of aero-elastic problems are summarized in Sections 3 and 4.

TABLE 2-1. Simplified Flow Diagram for Dynamic Aeroelastic Analysis

Step	Module	Status	Functions
1.	Static Part of NASTRAN	(Existing)	1. Forms structural mass, damping, and stiffness matrices. 2. Generates geometric data for structure.
2.	Dynamic Pool Distributor	(Existing)	Organizes tabular data for structure, control systems and loads.
3.	Real Eigenvalue Analysis, READ	(Existing)	Finds structural modes. (Used for modal approach only.)
4.	Aerodynamic Pool Distributor	(New)	Forms tables of aerodynamic data.
5.	Aerodynamic Element Generator	(New)	1. Defines boundaries of aerodynamic elements. 2. Locates and orients displacement components at aerodynamic control points.
6.	Aerodynamic Plotter	(New)	Plots aerodynamic elements and control point displacement directions in 3-D projection.
7.	Geometry Interpolator	(New)	Forms the matrix relating displacements at aerodynamic control points to structural displacements.
8.	Aerodynamic Matrix Generator	(New)	Forms the basic aerodynamic matrices according to each aerodynamic theory.
9.	Direct Dynamic Matrix Assembler	(Existing)	1. Decodes control system input. 2. Reduces direct matrix input. 3. Assembles complete dynamic matrices (excluding aerodynamic terms) for direct approach.
10.	Modal Dynamic Matrix Assembler	(Existing with minor modification)	Forms dynamic matrices for modal approach (excluding aerodynamic terms).
11.	Aerodynamic Matrix Processor	(New)	Forms composite aerodynamic matrices.
	Continue 2		

TABLE 2-1. Simplified Flow Diagram for Dynamic Aeroelastic Analysis (Cont.)

Step	Module	Status	Functions
	Continue 2		
12.	Aerodynamic Matrix Interpolator	(New)	Interpolates composite aerodynamic matrices for different Mach numbers and reduced frequencies, as required.
13.	Flutter Analysis	(New)	1. Combines matrices as required. 2. Finds roots of flutter determinant.
14.	Dynamic Aeroelastic Load Generator	(New)	1. Generates downwashes due to gusts and reduces them to structural grid point loads. 2. Combines gust loads with other loads in either time domain or frequency domain. 3. Finds modal excitation (for modal approach).
15.	Frequency Response	(Existing with modifications)	Finds $\{u_d\}$ or $\{u_h\}$ at discrete frequencies.
16.	Dynamic Data Recovery	(Existing)	1. Finds physical displacements (modal approach) 2. Uses mode acceleration method to improve displacements (optional).
17.	Aerodynamic Data Recovery	(New)	Recovers aerodynamic displacements, downwashes, pressures, and forces at control points (all optional).
18.	Static Data Recovery Modules	(Existing)	1. Recovers dependent displacements (optional) 2. Finds internal loads and stresses. (optional)
19.	Random Analysis	(Existing)	Finds psd, rms value and/or autocorrelation function for any response quantity. (Optional)
20.	Inverse Fourier Transform	(New)	Finds time history of any response quantity for Fourier transform method of transient analysis.
21.	Output File Processor	(Existing)	Organizes output data for printing and plotting.
	Continue 3		

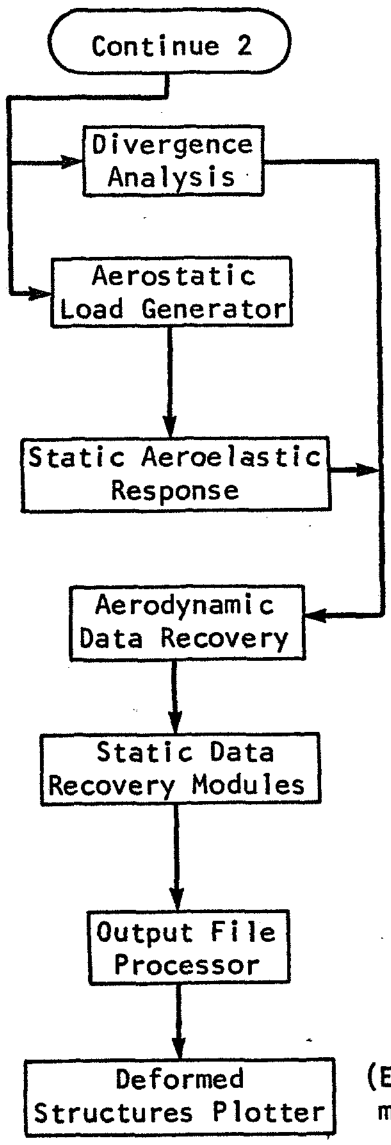
TABLE 2-1. Simplified Flow Diagram for Dynamic Aeroelastic Analysis (Cont.)

Step	Module	Status	Functions
	Continue 3		
22.	X-Y Plotter	(Existing with modifications)	1. Plots time histories (optional) 2. Makes Bode plots of frequency response output. (Optional) 3. Makes V-g and V-f plots of flutter roots. (New feature, optional)
23.	Deformed Structures Plotter	(Existing with modifications)	1. Plots structural modes in 3-D projection. (Optional) 2. Plots real and imaginary parts of flutter modes in 3-D projection (new feature, optional)

TABLE 2-2. Simplified Flow Diagram for Static Aeroelastic Analysis

Step	Module	Status	Functions
1.	Static Part of NASTRAN	(Existing)	Forms structural mass and stiffness matrices, rigid body properties and nonaerodynamic loads.
2.	Aerostatic Pool Distributor	(New)	Forms tables of aerodynamics data
3.	Aerodynamic Element Generator	(New)	1. Defines boundaries of aero elements. 2. Locates and orients displacement components at aero control points.
4.	Aerodynamic Plotter	(New)	Plots aerodynamic elements and control point displacement directions in 3-D projection.
5.	Geometry Interpolator	(New)	Forms the matrix relating displacements at aerodynamic control points to structural displacements.
6.	Aerostatic Matrix Generator	(New)	Forms the basic aerodynamic matrices and the steady angle of attack vector for each aerodynamic theory.
7.	Aerodynamic Matrix Processor	(New)	Forms composite aerodynamic matrices.
8.	Aerodynamic Matrix Interpolator	(New)	Interpolates the composite aerodynamic matrices for different Mach numbers as required.
9.	Aerostatic Matrix Assembler	(New)	Assembles matrices required for divergence analysis, untrimmed loads analysis or trimmed loads analysis.
	Continue 2		

TABLE 2-2. Simplified Flow Diagram for Static Aeroelastic Analysis (Cont.)



Step	Module	Status	Functions
10.	Divergence Analysis	(New)	Finds divergence speed(s) and eigenvectors.
11.	Aerostatic Load Generator	(New)	1. Reduces angle of attack distribution to structural gridpoint loads. 2. Combines air loads with other loads.
12.	Static Aeroelastic Response	(New)	Solves for displacements in either trimmed or untrimmed analysis.
13.	Aerodynamic Data Recovery	(New)	1. Recovers aerodynamic displacements, downwashes, pressures and forces at control points (all optional). 2. Recovers net forces and moments on vehicle for untrimmed condition (optional).
14.	Static Data Recovery Modules	(Existing)	1. Recovers dependent displacements. 2. Finds internal loads and stresses. (optional)
15.	Output File Processor	(Existing)	Organizes output data for printing and plotting.
16.	Deformed Structures Plotter	(Existing with modifications)	1. Plots divergence mode(s) in 3-D projection. (Optional) 2. Plots deformed structure due to loads. (Optional)

TABLE 2-3

NEW FUNCTIONAL MODULES

A. Modules used in both static and dynamic aeroelastic analysis

1. Aerodynamic Element Generator
2. Aerodynamic Plotter
3. Geometry Interpolator
4. Aerodynamic Matrix Processor
5. Aerodynamic Matrix Interpolator
6. Aerodynamic Data Recovery

B. Modules used only in dynamic aeroelastic analysis

7. Aerodynamic Pool Distributor
8. Aerodynamic Matrix Generator
9. Flutter Analysis
10. Dynamic Aeroelastic Load Generator
11. Inverse Fourier Transform

C. Modules used only in static aeroelastic analysis

12. Aerostatic Pool Distributor
13. Aerostatic Matrix Generator
14. Aerostatic Matrix Assembler
15. Divergence Analysis
16. Aerostatic Load Generator
17. Static Aeroelastic Response

TABLE 2-4

NEW RIGID FORMATS

1A. Direct Flutter Analysis

2A. Direct Aeroelastic Frequency and Random Response

3A. Direct Aeroelastic Transient Response by Fourier Integral Method

4A. Modal Flutter Analysis

5A. Modal Aeroelastic Frequency and Random Response

6A. Modal Aeroelastic Transient Response by Fourier Integral Method

7A. Aeroelastic Divergence

8A. Untrimmed Static Aeroelastic Response

9A. Trimmed Static Aeroelastic Response

Note: There are twelve NASTRAN rigid formats at the present time. The (A) after each rigid format number in the above list is intended to symbolize aeroelastic analysis.

3. FORMAL MATRIX ALGEBRA FOR NASTRAN DYNAMIC AEROELASTICITY

The formal matrix algebra for dynamic aeroelasticity will be summarized using the NASTRAN matrix terminology described in Appendix A. The development parallels the flow diagram in Table 2-1 of Section 2 and it is divided according to functional modules. The functional modules are described in detail in Section 5.

<u>Step 1: Static Part of NASTRAN</u> (existing)

Form: $[M_{aa}]$, $[K_{aa}]$, $[K^4_{aa}]$ and $[B_{aa}]$

Reference: NASTRAN Theoretical Manual, Section 9.3.3

<u>Step 3: Real Eigenvalue Analysis, READ</u> (existing)

a. Solve: $[K_{aa} - \lambda M_{aa}]\{u_a\}$
   for eigenvalues $\lambda_i = \omega_i^2$ and eigenvectors $\{\phi_{ai}\}$.

b. Normalize eigenvectors and form modal masses

$$m_i = \{\phi_{ai}\}^T[M_{aa}]\{\phi_{ai}\}$$

Reference: NASTRAN Theoretical Manual, Section 9.2.1

<u>Step 7: Geometry Interpolator</u> (new)

a. Form $[G_{ka}]$ directly, <u>or</u>

b. Form $[G_{kg}]$ and

   partition: $[G_{kg}] = [G_{km} \mid \overline{G}_{kn}]$

   form: $[G_{kn}] = [\overline{G}_{kn}] + [G_{km}][G_m]$

   partition: $[G_{kn}] = [G_{ks} \mid G_{ko} \mid \overline{G}_{ka}]$

   form: $[G_{ka}] = [\overline{G}_{ka}] + [G_{ko}][G_o]$

References: 1. Section 5.4
            2. Appendix E

Step 8. Aerodynamic Matrix Generator (New)

Form, according to the aerodynamic theory selected by the user:

$[D_{jk}]$, $[S_{kj}]$, and $[A_{jj}]$ or $[A_{jj}]^{-1}$

Reference: Section 5.5

Step 9. Direct Dynamic Matrix Assembler (Existing)

Form:
$$[K_{dd}] = (1 + ig)[K^1_{dd}] + [K^2_{dd}] + i[K^4_{dd}]$$
$$[B_{dd}] = [B^1_{dd}] + [B^2_{dd}]$$
$$[M_{dd}] = [M^1_{dd}] + [M^2_{dd}]$$

Reference: NASTRAN Theoretical Manual, Section 9.3.3

Step 10. Modal Dynamic Matrix Assembler (Existing with minor modification)

Form: $[M_{hh}]$, $[B_{hh}]$ and $[K_{hh}]$

modification: set $b_i = 0$ and $k_i = (1 + ig(\omega_i))\omega_i^2 m_i$

Reference: NASTRAN Theoretical Manual, Section 9.3.4

Step 11. Aerodynamic Matrix Processor (New)

a. For direct flutter or frequency response analysis, form:

$$[Q_{dd}] = \left[\begin{array}{c|c} Q_{aa} & Q_{ae} \\ \hline 0 & 0 \end{array}\right]$$

where:

$$[Q_{aa}] = [G_{ka}]^T[S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{ka}]$$
$$[Q_{ae}] = [G_{ka}]^T[S_{kj}][A_{jj}]^{-1}[D_{je}]$$

notes: (i) $[D_{je}]$ which relates downwash, $\{w_j\}$, to extra-point displacements, $\{u_e\}$, is supplied by the user.

(ii) If $[A_{jj}]$ rather than $[A_{jj}]^{-1}$ is formed in the aerodynamic matrix generator, operations with the triangular factors of $[A_{jj}]$ replace multiplication by $[A_{jj}]^{-1}$.

b. For modal flutter or frequency response analysis, form

$$[Q_{hh}] = \left[\begin{array}{c|c} Q_{ii} & Q_{ie} \\ \hline 0 & 0 \end{array}\right]$$

where:

$$[Q_{ii}] = [\phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{ka}][\phi_{ai}]$$

$$[Q_{ie}] = [\phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1}[D_{je}]$$

note: see notes (i) and (ii) above

c. For frequency response or for transient response also form

$$[Q_{aj}] = [G_{ka}]^T[S_{kj}][A_{jj}]^{-1}$$

for direct analysis, or

$$[Q_{ij}] = [\phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1}$$

for modal analysis.

Reference: Section 5.6

Step 12. Aerodynamic Matrix Interpolator (New)

Interpolate the matrices generated in the Aerodynamic Matrix Processor with respect to Mach number, m, and reduced frequency, k. Values are specified for a rectangular array of m and k (imbedded blanks are permitted). Also interpolate $[A_{jj}]^{-1}$ for use in aerodynamic data recovery.

References: 1. Section 5.7.
2. Appendix J.

Step 13. Flutter Analysis (New)

a. For k method, direct approach:

(i) form: $[M^a_{dd}] = \frac{\rho b^2}{2k^2}[Q_{dd}]$

(ii) find eigenvalues $p_j = i\omega_j\left(1 - \frac{i\bar{g}}{2}\right)$ and eigenvectors $\{\phi_{dj}\}$ for the eigenproblem

$$[(M_{dd} + M^a_{dd})p^2 + B_{dd}p + K_{dd}]\{u_d\} = 0$$

note: for the method of eigenvalue extraction described in Appendix H, it is required to set $[B_{dd}] = 0$.

(iii) repeat for selected values of ρ, k and m.

b. For k method, modal approach:

(i) form $[M^a_{hh}] = \frac{\rho b^2}{2k^2}[Q_{hh}]$

(ii) find eigenvalues $p_j = i\omega_j\left(1 - \frac{i\bar{g}}{2}\right)$ and eigenvectors $\{\phi_{hj}\}$ for the eigenproblem

$$[(M_{hh} + M^a_{hh})p^2 + B_{hh}p + K_{hh}]\{u_h\} = 0$$

note: for the method of eigenvalue extraction described in Appendix H, it is required to set $[B_{hh}] = 0$.

(iii) repeat for selected values of ρ, k and m.

c. For p-k method:

(i) direct approach, form $[K^a_{dd}] = -\frac{1}{2}\rho V^2[Q_{dd}]$

(ii) modal approach, form $[K^a_{hh}] = -\frac{1}{2}\rho V^2[Q_{hh}]$

(iii) direct approach, find eigenvalues p_j and eigenvectors $\{\phi_{dj}\}$ for the eigenproblem

$$[M_{dd}p^2 + B_{dd}p + (K_{dd} + K^a_{dd})]\{u_d\} = 0$$

by the inverse power method described in Appendix J.

(iv) modal approach, substitute subscript h for subscript j in (iii)

References: Section 5.8
Appendices F, H, I

Step 14. Dynamic Aeroelastic Load Generator (New)

a. Direct approach, form the load vector

$$\{P_d\} = \left\{\begin{array}{c} P^s_a \\ \hdashline P^s_e \end{array}\right\} + \left\{\begin{array}{c} P^a_a \\ \hdashline 0 \end{array}\right\}$$

where $\{P^s_a\}$ and $\{P^s_e\}$ are structural loads and

$$\{P^a_a\} = q[G_{ka}]^T[S_{kj}][A_{jj}]^{-1}\{w^g_j\} = q[Q_{aj}]\{w^g_j\}$$

b. Modal approach, form the load vector

$$\{P_h\} = \left\{\begin{array}{c} P^s_i \\ \hdashline P^s_e \end{array}\right\} + \left\{\begin{array}{c} P^a_i \\ \hdashline 0 \end{array}\right\}$$

where $\{P^s_i\}$ and $\{P^s_e\}$ are structural loads and

$$\{P^a_i\} = q[\phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1}\{w^g_j\} = q\,[Q_{ij}]\{w^g_j\}$$

notes:

(i) see NASTRAN Theoretical Manual, Section 12.1 for generation of structural loads.

(ii) see Section 5.9 and Appendix D for transformation of transient loads into frequency response loads.

(iii) a routine for automatic generation of the downwash due to gust, $\{w_j^g\}$, including time delays, is provided.

(iv) the matrices $[Q_{aj}]$ or $[Q_{ij}]$ are interpolated to the required frequencies in the Aerodynamic Matrix Interpolator.

References: Section 5.9
Appendix D

Step 15. Frequency Response (Existing, with modifications)

a. Direct approach, form $[K^a_{dd}] = -\frac{1}{2}\rho V^2[Q_{dd}]$

b. Modal approach, form $[K^a_{hh}] = -\frac{1}{2}\rho V^2[Q_{hh}]$

c. Direct approach, solve for $\{u_d\}$ for various values of ω:

$$[-\omega^2 M_{dd} + i\omega B_{dd} + K_{dd} + K^a_{dd}]\{u_d\} = \{P_d\}$$

d. Modal approach, replace subscript d by subscript h.

Note: $[Q_{dd}]$ and $[Q_{hh}]$ are functions of $k = \frac{b\omega}{V}$. They are interpolated by the aerodynamic matrix interpolator.

References: NASTRAN Theoretical Manual Section 12.1

Step 16. Dynamic Data Recovery (Existing)

a. For modal approach, calculate

$$\{u_d\} = [\phi_{dh}]\{u_h\}$$

b. Partition

$$\{u_d\} = \left\{\begin{array}{c} u_a \\ \hdashline u_e \end{array}\right\}$$

c. Use mode acceleration method to improve $\{u_a\}$. (Optional)

Reference: NASTRAN Theoretical Manual, Section 9.4

Step 17. Aerodynamic Data Recovery (New)

For frequency response analysis, or for flutter modes using the imaginary part of $p = \alpha + i\omega$, form:

$$\{u_k\} = [G_{ka}]\{u_a\}$$

$$\{w_j\} = [D_{jk}]\{u_k\} + [D_{je}]\{u_e\} + \{w_j^g\}$$

$$\{f_j^a\} = q[A_{jj}]^{-1}\{w_j\}$$

$$\{F_k^a\} = [S_{kj}]\{f_j^a\}$$

Note: $[A_{jj}]^{-1}$ is interpolated to the required frequency by the Aerodynamic Matrix Interpolator.

Reference: Section 5.10.

Step 20. Inverse Fourier Transform (New)

For the vector of response quantities, $\{u(\omega)\}$, form

$$\{u(t)\} = \frac{1}{\pi} \int_0^\infty R\ell \left[e^{i\omega t}\{u(\omega)\}\right] d\omega$$

by approximate methods.

References: Section 5.11
Appendix D

4. FORMAL MATRIX ALGEBRA FOR NASTRAN STATIC AEROELASTICITY

The formal matrix algebra for static aeroelasticity will be summarized using the NASTRAN matrix terminology described in Appendix A. The development parallels the flow diagram in Table 2-2 of Section 2, and it is divided according to functional modules. Detailed descriptions of the functional modules are given in Section 5.

<u>Step 1. Static Part of NASTRAN</u> (existing)

Form: $[K_{\ell\ell}]$, $\{P_\ell^s\}$, $[M_{\ell\ell}D + M_{\ell r}]$, $[m_r]$, $[D]$, $\{P_r^s\}$

Reference: NASTRAN Theoretical Manual, Sections 3.5.5 and 3.6.3

<u>Step 5. Geometry Interpolator</u> (new)

a. Form $[G_{ka}]$ directly, <u>or</u>

b. Form $[G_{kg}]$ and

partition: $[G_{kg}] = [G_{km} \mid \bar{G}_{kn}]$

form: $[G_{kn}] = [\bar{G}_{kn}] + [G_{km}][G_m]$

partition: $[G_{kn}] = [G_{ks} \mid G_{ko} \mid \bar{G}_{ka}]$

form: $[G_{ka}] = [\bar{G}_{ka}] + [G_{ko}][G_o]$

c. partition: $[G_{ka}] = [G_{k\ell} \mid G_{kr}]$

Note: Except for the last partition, the function of the Geometry Interpolator is the same in statics and dynamics.

References: 1. Section 5.4
2. Appendix E

Step 6. Aerostatic Matrix Generator (new)

a. Form $[D_{jk}]$, $[S_{kj}]$ and $[A_{jj}]$ or $[A_{jj}]^{-1}$. These tasks are analogous to those for dynamic analysis.

b. Form $\{w_j^g\}$, the static angle of attack distribution.

c. Form $[D_{je}]$ which relates downwash (and perhaps other aerodynamic variables) to perturbation velocity components of the vehicle and control surface rotations.

Reference: Section 5.13

Step 7. Aerodynamic Matrix Processor (new)

a. Form

$$[Q_{dd}] = \begin{array}{c} \begin{array}{ccc} u_\ell & u_r & u_e^a \end{array} \\ \left[\begin{array}{c|c|c} Q_{\ell\ell} & 0 & Q_{\ell e} \\ \hline Q_{r\ell} & 0 & Q_{re} \\ \hline 0 & 0 & 0 \end{array}\right] \end{array}$$

where

$$[Q_{\ell\ell}] = [G_{k\ell}]^T[S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{k\ell}]$$

$$[Q_{r\ell}] = [G_{kr}]^T[S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{kr}]$$

$$[Q_{\ell e}] = [G_{k\ell}]^T[S_{kj}][A_{jj}]^{-1}[D_{je}]$$

$$[Q_{re}] = [G_{kr}]^T[S_{kj}][A_{jj}]^{-1}[D_{je}]$$

b. For use in generating aerostatic loads, form

$$[Q_{\ell j}] = [G_{k\ell}]^T[S_{kj}][A_{jj}]^{-1}$$

and

$$[Q_{rj}] = [G_{kr}]^T[S_{kj}][A_{jj}]^{-1}$$

Notes: 1. $[D_{je}]$ is automatically generated in Step 6.

2. If $[A_{jj}]$ rather than $[A_{jj}]^{-1}$ is formed in the Aerodynamic Matrix Generator, operations with the triangular factors $[A_{jj}]$ replace multiplication by $[A_{jj}]^{-1}$.

Reference: Section 5.6

Step 8. Aerodynamic Matrix Interpolator (new)

Interpolate the matrices formed in the Aerodynamic Matrix Processor vs. Mach number.

Reference: Section 5.7.

Step 9. Aerostatic Matrix Assembler (new)

a. Form

$$[K^a_{\ell\ell}] = -q[Q_{\ell\ell}]$$

$$[K^a_{\ell e}] = -q[Q_{\ell e}]$$

$$[K^a_{r\ell}] = -q[Q_{r\ell}]$$

$$[K^a_{re}] = -q[Q_{re}]$$

b. For untrimmed static loads and divergence, partition

$$[K^a_{\ell e}] = [K^a_{\ell a} \mid 0]$$

$$[K^a_{re}] = [K^a_{ra} \mid 0]$$

where $[K^a_{\ell a}]$ and $[K^a_{ra}]$ refer to velocity components and control surface rotations included in $\{u^a_e\}$.

c. For untrimmed static loads, direct solution option,

Form $[K_{dd}]$ in $[K_{dd}]\{u_d\} = \{P_d\}$, which is written in expanded form as

$$\left[\begin{array}{c|c|c|c} K_{\ell\ell} + K^2_{\ell\ell} + K^a_{\ell\ell} & M_{\ell r} + M_{\ell\ell}D & K^a_{\ell a} + K^2_{\ell a} & K^2_{\ell o} \\ \hline D^T(K^2_{\ell\ell} + K^a_{\ell\ell}) + K^2_{r\ell} + K^a_{r\ell} & m_r & D^T[K^a_{\ell a} + K^2_{\ell a}] + K^a_{ra} + K^2_{ra} & D^T K^2_{\ell o} + K^2_{ro} \\ \hline K^2_{a\ell} & 0 & I & K^2_{ao} \\ \hline K^2_{o\ell} & 0 & K^2_{oa} & K^2_{oo} \end{array}\right] \left\{\begin{array}{c} u_\ell \\ \hline \ddot{u}_r \\ \hline u^a_e \\ \hline u^o_e \end{array}\right\} = \{P_d\}$$

notes: 1. $\{\ddot{u}_r\}$ and $\{u^a_e\}$ are automatically generated extra points.
$\{u^o_e\}$ is user-generated.

2. All matrices with superscript $(^2)$ are user-supplied.

d. For divergence and iterative solution option, form

$$[K^1_{dd} + K^2_{dd}] = \left[\begin{array}{c|c|c|c} K_{\ell\ell} + K^2_{\ell\ell} & M_{\ell r} + M_{\ell\ell}D & 0 & K^2_{\ell o} \\ \hline D^T K^2_{\ell\ell} + K^2_{r\ell} & m_r & 0 & D^T K^2_{\ell o} + K^2_{ro} \\ \hline K^2_{a\ell} & 0 & I & K^2_{ao} \\ \hline K^2_{o\ell} & 0 & K^2_{oa} & K^2_{oo} \end{array}\right]$$

and

$$[K^a_{dd}] = \left[\begin{array}{c|c|c|c} K^a_{\ell\ell} & 0 & K^a_{\ell a} & 0 \\ \hline D^T(K^a_{\ell\ell}) + K^a_{r\ell} & 0 & D^T K^a_{\ell a} + K^a_{ra} & 0 \\ \hline 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 \end{array}\right]$$

e. For trimmed static loads

(i) partition $[K^a_{\ell e}] = [K^a_{\ell t} \mid K^a_{\ell u}]$

$$[K^a_{re}] = [K^a_{rt} \mid K^a_{ru}]$$

where $[K^a_{\ell t}]$ and $[K^a_{rt}]$ refer to the subset of velocity components and control surface rotations that are used to trim the vehicle and $[K^a_{\ell u}]$ and $[K^a_{ru}]$ refer to all other extra points.

(ii) For the direct solution option

Form $[K_{dd}]$ in $[K_{dd}]\{u_d\} = \{P_d\}$, which may be written in expanded form as

$$\left[\begin{array}{c|c|c} K_{\ell\ell} + K^2_{\ell\ell} + K^a_{\ell\ell} & K^a_{\ell t} & K^a_{\ell u} + K^2_{\ell u} \\ \hline D^T(K^2_{\ell\ell} + K^a_{\ell\ell}) & D^T K^a_{\ell t} & D^T(K^a_{\ell u} + K^2_{\ell u}) \\ + K^2_{r\ell} + K^a_{r\ell} & + K^a_{rt} & + K^a_{ru} + K^2_{ru} \\ \hline K^2_{u\ell} & K^2_{ut} & K^2_{uu} \end{array}\right] \left\{\begin{array}{c} u_\ell \\ \hline u^t_e \\ \hline u^u_e \end{array}\right\} = \{P_d\}$$

(iii) For the iterative solution option, separate $[K^a]$ terms from $[K + K^2]$ terms.

Notes: 1. $\{u_e^t\}$ is selected by the user from $\{u_e^a\}$; $\{u_e^u\}$ contains the remaining members of $\{u_e^a\}$ plus all members of $\{u_e^o\}$.

2. The partition of $[K_{uu}^2]$ corresponding to the remaining members of $\{u_a\}$ will be set equal to an identity matrix if no values are supplied by the user.

Reference: Appendix B.

Step 10. Divergence Analysis (new)

Calculate eigenvalues and eigenvectors of

$$[K_{dd}^1 + K_{dd}^2 + \lambda K_{dd}^a]\{u_d\} = 0$$

References: 1. Section 5.15
2. Appendix C.

Step 11. Aerostatic Load Generator (new)

a. Form

$$\{P_\ell^a\} = q[Q_{\ell j}]\{w_j^g\}$$

and

$$\{P_r^a\} = q[Q_{rj}]\{w_j^g\}$$

The downwash vector $\{w_j^g\}$ is calculated automatically by theory-dependent subroutines in the Aerostatic Matrix Generator. $[Q_{\ell j}]$ and $[Q_{rj}]$ are formed in the Aerodynamic Matrix Interpolator.

b. For untrimmed load cases form

$$\{P_d\} = \begin{Bmatrix} P_\ell \\ P_e^r \\ P_e^a \\ P_e^o \end{Bmatrix} = \begin{Bmatrix} P_\ell^s + P_\ell^a \\ \hline P_r^s + D^T(P_\ell^s + P_\ell^a) + P_r^a \\ \hline P_e^a \\ \hline P_e^o \end{Bmatrix}$$

where $\{P_\ell^s\}$ and $\{P_r^s\}$ are structural loads, $\{P_e^a\}$ and $\{P_e^o\}$ are loads on extra points specified directly by the user.

c. For trimmed load cases form

$$\{P_d\} = \begin{Bmatrix} P_\ell \\ P_e^t \\ P_e^u \end{Bmatrix} = \begin{Bmatrix} P_\ell^s + P_\ell^a \\ \hline P_r^s + D^T(P_\ell^s + P_\ell^a) + P_r^a \\ \hline P_e^u \end{Bmatrix}$$

$\{P_e^u\}$ are loads on extra points specified directly by the user.

Reference: Section 5.16.

Step 12. Static Aeroelastic Response (new)

a. For direct solution option, solve

$$[K_{dd}]\{u_d\} = \{P_d\}$$

b. For iterative solution option, solve iteratively until convergence

$$[K_{dd}^1 + K_{dd}^2]\{u_d^n\} = \{P_d\} - [K_{dd}^a]\{u_d^{n-1}\}$$

References: Section 5.17
Appendix B

Step 13, Aerodynamic Data Recovery (new)

Form
$$\{u_k\} = [G_{k\ell}]\{u_\ell\} \qquad (1)$$

$$\{w_j\} = [D_{jk}]\{u_k\} + [D_{je}]\{u_e\} + \{w_j^g\} \qquad (2)$$

$$\{f_j^a\} = q[A_{jj}]^{-1}\{w_j\} \qquad (3)$$

$$\{F_k^a\} = [S_{kj}]\{f_j^a\} \qquad (4)$$

$$\{F_r\} = [m_r]\{\ddot{u}_r\} \qquad (5)$$

Notes: 1. Equations 1, 2, 3, and 4 are identical to those for dynamic analysis.

2. Equation 5 gives resultant forces on the vehicle.

3. $[A_{jj}]^{-1}$ is interpolated vs. Mach number in the Aerodynamic Matrix Interpolator.

Reference : Section 5.10.

5. DESCRIPTIONS OF NEW FUNCTIONAL MODULES

5.1 Aerodynamic Pool Distributor

The purpose of the Aerodynamic Pool Distributor is to reduce the number of files required to be opened during setup of structural matrices, and to preprocess the aerodynamic data. All aerodynamic data is transferred by the input file processor onto one data block (called AEDECK for AErodynamic DECK). This data is pre-processed by the Aerodynamic Pool Distributor before any modules requiring this data are executed, but after completion of the structural tasks.

One of the module tasks is to supply default values for all input data cards. The default values are listed on the card descriptions, see Section 6.2.

Another task is to sort the data for the aerodynamic theories. It will do this by computing lists of entries for each CAERØ and PAERØ data card. The items on the lists are:

1.	For CAERØ cards:	CID (CAERØ ID)
		PID (PAERØ ID)
		TID (theory ID, to be found)
2.	For PAERØ cards:	PID (PAERØ ID)
		TID (theory ID)
		USED $\begin{cases} 0 \rightarrow \text{card not referenced} \\ +1 \rightarrow \text{card is referenced} \end{cases}$

a. The first step is to compile the lists, getting the first two entries from the data cards* The third entries are equal to zero. The lists are sorted on the first entry (CID for CAERØ cards and PID for PAERØ cards). If any two PAERØ cards have the same ID, a fatal message occurs.

* The input cards are listed in Section 6.

b. Scan the PID's on the CAERØ card list, and see if the corresponding PAERØ card can be found. If it can't, prepare a warning message. If a PAERØ reference is found, find the TID and put it in the CAERØ list in third entry; also set "USED" for the PAERØ = +1.

c. Scan the "USED" entries on the PAERØ card list, and prepare a warning message if "USED" = 0.

d. Form a revised CAERØ card list. First discard all entries for which TID = 0. Then sort on the contents of the TID entries.

This list gives a reference to all aero-cells in each theory, and will be needed by the Aerodynamic Element Generator, phase 2. Note that cards not referenced are not used, but produce warning messages.

The other task performed by the Aerodynamic Pool Distributor is to pass on the images for each card to the module, or modules, which require them:

Data List	Name	Module using it
connections (includes theory table)	ADTCN	Ae Elem Gen
properties (includes m-k values)	ADTPR	Ae Matrix Gen
splines	ADTSP	Geom Int
loads	ADTLD	Dyn Ae Load Gen
methods	ADTMD	Flutter Anal, Inv Fourier Trns

5.2 Aerodynamic Element Generator

The Aerodynamic Element Generator processes the user supplied information about aerodynamic cells. The job is divided into two tasks. The first, called Aerodynamic Element Generator-1 does only the geometry (found on CAERØ data cards) associated with the cell. The second part, called Aerodynamic Element Generator-2, performs those tasks which refer to a specific theory, such as choosing the number of degrees of freedom per cell, and locating them (using data found on the PAERØ data cards).

5.2.1 Aerodynamic Element Generator, Part 1

The purpose of the Aerodynamic Element Generator-1 is to process the user supplied information about the aerodynamic cell geometry, producing a data list which is useful for the Aerodynamic Matrix Generator , the Geometry Interpolator and the Aerodynamic Model Plotter. The aerodynamic element is similar to the NASTRAN structural element connection in that it provides geometric information. The chief innovation is the concept of a macro-element which defines many geometrically similar aerodynamic elements.

The input to the Aerodynamic Element Generator Module comes from the NASTRAN bulk data deck and from the output of the previously executed Geometry Processor modules. The input data cards have been designed for user convenience. There are two types of aerodynamic elements, the quadrilateral and the cylinder. There are three ways allowed to describe quadrilaterals and cylinders on bulk data cards:

A. Quadrilateral

1. Give grid point numbers at four corners.
2. Give the location of the four corners in defined coordinate systems.
3. Give the location of the two leading edge corners and the edge chord lengths.

B. Cylinder

1. Give grid point numbers at the two ends of the axis.
2. Give the location of the two ends of the axis in defined coordinate systems.
3. Give one location and one length.

Sample cards are in Figures 6-1 through 6-4.

The grid point numbers refer to locations defined by GRID bulk data cards. A single aerodynamic coordinate system will be user supplied in which the flow is in the $+X_1$ direction, for aerodynamic calculations; see Figure 6-7.

The desired form of output is different for each of the several theories which may use the output from the Aerodynamic Element Generator module. The theories currently being considered are:

1. Doublet lattice (Subsonic and Supersonic)
2. Lifting cylinder (Subsonic)
3. Strip
4. Piston
5. Newtonian

Newtonian theory can accept any quadrilateral or cylinder which we will call format 1. Piston theory requires that the quadrilateral lie parallel to the flow (with a possible small initial angle of attack); this is format 2. Lattice theory requires, in addition, that the side chords be parallel to the flow, thus forming trapezoids; this is format 3. In all cases the quadrilateral aerodynamic elements will be made planar, even though the user may supply four points not on a plane. A basic decision is to output data in all three formats, plus another format useful for plotting. This implies that the aerodynamic element module need not know which theory is going to be used.

Images of the aerodynamic cell connection cards, plus the BGPDT and CSTM files are to be used by this module. The BGPDT (basic grid point data table) file contains a list of grid point coordinate systems, and the locations of the point in the basic coordinate system. The CSTM (coordinate system transformation matrix) file contains the coordinate system identification, the type, the system origin in basic, and the transformation matrix (the columns of this 3 x 3 are the $\bar{i}$, $\bar{j}$, and $\bar{k}$ unit vectors of the coordinate system referred to basic). Before beginning, locate the aerodynamic coordinate system and check that its type is rectangular. Then take the following steps for all elements.

Step 1. Locate the four corners or two ends of a macro-element in basic coordinates.

If CAERØ1 card: look up coordinates in BGPDT table.

If CAERØ2 card: compute basic coordinates using information in CSTM. This code should be similar to that used in NASTRAN GP1 to form the BGPDT.

If CAERØ3 card: compute coordinates of G_1 and G_4 using CSTM. Then

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_2 = \begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_1 + \Delta X_{12} \begin{Bmatrix} i \end{Bmatrix} \tag{1}$$

$$\begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_3 = \begin{Bmatrix} X \\ Y \\ Z \end{Bmatrix}_4 + \Delta X_{43} \begin{Bmatrix} i \end{Bmatrix} \tag{2}$$

where $\{i\}$ is the first column of the transformation matrix to aerodynamic coordinates.

If CAERØ4, CAERØ5, or CAERØ6; the task is analogous to the above for 1, 2, or 3, except that only 2 locations are defined.

Step 2. The quadrilateral of type 1 and 2 may be nonplanar. We replace the macro by an element projected on a plane half-way between the diagonals.

1. Let $\overline{r_i} = \begin{Bmatrix} X_i \\ Y_i \\ Z_i \end{Bmatrix}$ i=1, 2, 3, 4 in basic (3)

2. Find $\overline{V} = (\overline{r_3}-\overline{r_1}) \times (\overline{r_4}-\overline{r_2}) = \begin{Bmatrix} (y_3-y_1)(z_4-z_2)-(z_3-z_1)(y_4-y_2) \\ (z_3-z_1)(x_4-x_2)-(x_3-x_1)(z_4-z_2) \\ (x_3-x_1)(y_4-y_2)-(y_3-y_1)(x_4-x_2) \end{Bmatrix}$ (4)

3. Find $\overline{\delta r} = \dfrac{V\cdot(\overline{r_2}-\overline{r_1})}{2(\overline{V}\cdot\overline{V})}\ \overline{V}$ (5)

4. Print a warning message if

$$\frac{(\overline{\delta r})\cdot(\overline{\delta r})}{(\overline{V}\cdot\overline{V})^{1/2}} > .01 \qquad (6)$$

since the quadrilateral was not flat.

5. The corrected values are

$$\begin{array}{l} \overline{r_1} + \overline{\delta r} \\ \overline{r_2} - \overline{\delta r} \\ \overline{r_3} + \overline{\delta r} \\ \overline{r_4} - \overline{\delta r} \end{array} \qquad (7)$$

6. The area is given by

$$1/2 \quad (\overline{V}\cdot\overline{V})^{1/2} \qquad (8)$$

Skip the step for cylinders (i.e. type 4, 5, 6).

Step 3. Form the element data by cutting the macro-element. The input file processor will produce the following list of division points from the bulk data cards.

NSPAN (an integer)

$$\left.\begin{matrix}0\\ \vdots\\ 1.00\end{matrix}\right\}\ (\text{NSPAN}+1)\quad\text{entries}\ \left.\vphantom{\begin{matrix}0\\ \vdots\\ 1.00\end{matrix}}\right\}\ \text{CAERØ type 1, 2, 3}$$

NCHØRD

$$\left.\begin{matrix}0\\ \vdots\\ 1.00\end{matrix}\right\}\ (\text{NCHØRD}+1)\quad\text{entries}\ \left.\vphantom{\begin{matrix}0\\ \vdots\\ 1.00\end{matrix}}\right\}\ \text{all cards}$$

Let $R_i = \begin{Bmatrix} x \\ y \\ z \end{Bmatrix}_i$ be the location of the four corners of the macro-element.

Let f_m $m = 0, 1 \ldots$ (NSPAN)

and g_n $n = 0, 1 \ldots$ (NCHØRD) be the fractional distances.

Then compute for $m = 1 \ldots$ NSPAN

$n = 1 \ldots$ NCHØRD

1. The element identification

   EID.m.n (quad)
   EID.n (cylinder)

2. The corner locations in basic coordinates

$$r_i = \begin{Bmatrix} x \\ y \\ z \end{Bmatrix}_i \tag{9}$$

$$\begin{aligned} r_{1,m,n} = {} & (1-f_{m-1})(1-g_{n-1})\ R_1 + (f_{m-1})(1-g_{n-1})\ R_4 \\ & + (1-f_{m-1})(g_{n-1})\ R_2 + (f_{m-1})(g_{n-1})\ R_3 \end{aligned} \tag{10}$$

$$\begin{aligned} r_{4,m,n} = {} & (1-f_m)(1-g_{n-1})\ R_1 + (f_m)(1-g_{n-1})\ R_4 + (1-f_m)(g_{n-1})\ R_2 \\ & + (f_m)(g_{n-1})\ R_3 \end{aligned} \tag{11}$$

$$r_{2,m,n} = (1-f_m)(1-g_n)R_1 + (f_m)(1-g_n)R_4 + (1-f_m)(g_n)R_2 + (f_m)(g_n)R_3 \quad (12)$$

$$r_{3,m,n} = (1-f_{m-1})(1-g_n)R_1 + (f_{m-1})(1-g_n)R_4 + (1-f_{m-1})(g_n)\ R_2 + (f_{m-1})(g_n)R_3 \quad (13)$$

The area is computed for each sub-element using equations (4) and (8) of Step 2. The division for a cylinder is similar. The basic coordinate lists will be used for plotting.

Step 4. Transform to aero coordinates.

$$\{r_i\}_{aero} = [T]^T\left(\{r_i\}_{basic} - \{r_0\}\right) \quad (14)$$

where r_0 and T are the CSTM entries for the aerodynamic coordinate system. This is computed for every corner of every element, and the two ends of a cylinder.

Step 5. Transform to quadrilateral parallel to the flow. First compute the angle of attack. Let

$$\{r_i\} = \begin{Bmatrix} x \\ y \\ z \end{Bmatrix}_{i,\ aero} \quad \text{as computed in Step 4.} \quad (15)$$

Then compare(in case the quadrilateral has a side of zero length)

$$|r_2 - r_1| \text{ and } |r_3 - r_4| \quad .$$

Let k=2 if $|r_2 - r_1|$ is longer, otherwise k=3.

Then compute a unit normal vector in the aero coordinate system.

$$\frac{(r_k-r_1) \times (r_4-r_1)}{|(r_k-r_1) \times (r_4-r_1)|}$$

The first component is sin α_0. Then while projecting the element on a plane parallel to the flow, the values of x will not change. Let the vectors be two-dimensional for the rest of Steps 5 and 6, then in the cross-flow plane:

$$\overline{r}_i = \begin{Bmatrix} y \\ z \end{Bmatrix}_i \tag{16}$$

$$\text{Let } \overline{V} = 1/2\,(\overline{r}_3+\overline{r}_4) - 1/2(\overline{r}_1+\overline{r}_2) \tag{17}$$

$$\text{Let } \delta r_a = 1/2 \left[(r_2-r_1) - \frac{\overline{V}\cdot(\overline{r}_2-\overline{r}_1)}{\overline{V}\cdot\overline{V}}\;\overline{V} \right] \tag{18}$$

$$\delta r_b = 1/2 \left[(r_3-r_4) - \frac{\overline{V}\cdot(\overline{r}_3-\overline{r}_4)}{V\cdot V}\;\overline{V} \right] \tag{19}$$

then take

$$\begin{gathered} \overline{r_1} + \overline{\delta r_a} \\ \overline{r_2} - \overline{\delta r_a} \\ \overline{r_3} - \overline{\delta r_b} \\ \overline{r_4} + \overline{\delta r_b} \end{gathered} \tag{20}$$

This will be done for every corner of every element, which results in quadrilaterals parallel to the airflow.

For the cylinders (i.e. type 4,5,6)

$$\overline{\delta r} = 1/2(\overline{r_2}-\overline{r_1}) \tag{21}$$

and the corrected values are

$$\begin{gathered} \overline{r_1} + \overline{\delta r} \\ \overline{r_2} - \overline{\delta r} \end{gathered} \tag{22}$$

Step 6. Transform to a trapezoid. (quadrilaterals only)

$$\begin{Bmatrix} y \\ z \end{Bmatrix}_{1,2} = \tfrac{1}{2}\left(\begin{Bmatrix} y \\ z \end{Bmatrix}_1 + \begin{Bmatrix} y \\ z \end{Bmatrix}_2 \right) \tag{23}$$

$$\begin{Bmatrix} y \\ z \end{Bmatrix}_{3,4} = \frac{1}{2}\left(\begin{Bmatrix} y \\ z \end{Bmatrix}_3 + \begin{Bmatrix} y \\ z \end{Bmatrix}_4\right),$$

where $\begin{Bmatrix} y \\ z \end{Bmatrix}_i$ are defined in step 5.

The span is given by

$$\Delta\eta = [(y_1 - y_4)^2 + (z_1 - z_4)^2]^{1/2}$$

Step 7. Prepare the output list. The list will contain:

1. The macro-element ID
2. The element type

If type 1, 2, 3 (quadrilateral)

3.	The span number	
4.	the chord number	
5.-16.	$x_1, y_1, z_1, x_2, \ldots, z_4$	basic coordinates
17.-28.	$x_1, y_1, z_1, x_2, \ldots, z_4$	aero coordinates
29.-36.	$y_1, z_1, y_2, z_2, \ldots, y_4, z_4$	aero coordinates format 2
37.-40.	y_1, z_1, y_4, z_4	aero coordinates format 3
41.	$\sin\alpha_0$	the angle of attack
42.	A	the area
43.	$\Delta\eta$	the span

If type 4, 5, or 6 (cylinder)

3.	the subelement number	
4.-9.	$x_1, y_1, z_1, x_2, y_2, z_2$	basic coordinates
10.-15.	$x_1, y_1, z_1, x_2, y_2, z_2$	aero coordinates, format 1
16.-17.	$y_{1,2}, z_{1,2}$	aero coordinates, format 2, 3

5.2.2 Aerodynamic Element Generator, Part 2

The purpose of the Aerodynamic Element Generator-2 is to produce a list of the aerodynamic degrees of freedom, which are associated with the aerodynamic elements. These sets are called the k-set and the j-set. These are somewhat parallel to the other displacement sets defined in the structural NASTRAN solution algorithm, called g-set, a-set, etc. These sets are defined as:

Set k is a set of interpolated structural points. They are associated with local (i.e., global) coordinate systems with one to six degrees of freedom. The location may be specified by an aerodynamic theory as a specified location in a cell, or by the use of GRIDK bulk data cards. The directions for the global coordinate system will be internally computed for aero-cells, and will be user specified by referring to a coordinate system for the GRIDK method. Set k points may be plotted.

Set j is a set of aerodynamic points. A location and direction is specified solely for the purpose of plotting output. The physical interpretation of a displacement and force in j-set depends upon the aerodynamic theory, and may represent normal-wash and pressure, flap angle and hinge moment, camber and generalized force or others. The number of j-set points per aerodynamic cell may vary, but it will usually be 1.

For dynamic aeroelastic analysis, the Aerodynamic Element Generator-2 will provide the sets of j and k coordinates appropriate to the chosen theories. This will consist of a data block called USETKJ with the following information.

1. For k-set
    a. the external identification numbers: aero macro ID, span index, chord index, and the index in that cell (as there may be more than one). The GRIDK ID is used when the k-point is defined on a GRIDK card.
    b. location in basic coordinate system.
    c. code for the degrees of freedom (may include one thru six, and a typical code of 35 would indicate T3 and R2 motion in the global coordinate system).

2. For j-set
    a. the external identification numbers (similar to the k-set)
    b. the location in basic coordinate system for plotting displacements, and a unit vector for direction.
    c. the location in basic coordinate system for plotting forces, and a unit vector.

The order in which the entries appear will be as follows.

a. Sorted by aerodynamic theory number.

b. Sorted by external identification numbers.

The order of the entries on this list is the internal indexing scheme.

A header record will list the number of aerodynamic theories used, and the number of degrees of freedom in set k and set j in each aerodynamic theory. Also a list of all splines referenced, and a list of the number of degrees of freedom in k-set for each spline. These header records will be useful for indexing thru these lists in the Geometry Interpolator and Aerodynamic Matrix Modules.

5.3 Aerodynamic Plotter Module

The Aerodynamic Plotter Module serves two functions. It plots the aerodynamic elements for a visual display of geometry, which is similar to the role of the NASTRAN structure plotter. It also plots results, which is similar to the role of the NASTRAN deformed structure plotter. The aerodynamic plotter will extend the present plotting capability to include the following:

1. Aerodynamic elements
2. Aerodynamic element labels
3. k set point locations
4. k set point labels
5. j set point locations
6. j set point labels
7. k set deflections
8. j set deflections
9. j set forces

The present deformed structure capability will be extended to plot complex results. Thus plots can be made of the real part (in-phase) or the imaginary part (out-of-phase) of the structural and aerodynamic deflections. Magnitude can also be plotted. Provision will be made for linear combinations of the in-phase and out-of-phase plots in order to plot the results at any phase; this could be used to make the frames for a movie of flutter mode, if desired.

The plot package will be able to make overlay plots using the same viewing angles and scales in each plot. This is done to reduce the amount of clutter in the presentation. Overlay plots can also be made with the structures plotter in order to check congruence of the aerodynamic model and the structural model. For example, the plotting package can be used to visually check the accuracy of displacement interpolation by outputing the g set and the k set displacements for identically scaled plots.

The instructions for the plotter will be in the Case Control Deck. The selection of plotter, view angles, scaling, etc., will be the same as at present. No new data cards are needed. All that is required is that the interpretation of the PLØT card be extended to recognize aerodynamic quantities. Overlays are made with two plot calls using the same SCALE, ØRIGIN, VANTAGE PØINT, VIEW, and AXES. The desired PHASE will be an alternate to the existing RANGE/TIME of the PLØT card. Thus, no new data cards are needed.

5.4 Geometry Interpolator

The purpose of this module is to provide a transformation matrix which gives structural displacements at a set of interpolated (or extrapolated) locations in terms of deflections at structural grid points. The matrix coefficients are determined by using linear and surface splines, which give "structural-like" deformation patterns since they are beams and plates. The Geometry Interpolator performs the following two tasks:

1. It selects the structural grid points to be used. The spline is connected to a subset of the degrees of freedom of the structure, which are chosen to be in a region desired for interpolation. The user may specify the structural grid points in terms of aerodynamic elements, or in terms of a list.

2. It constructs interpolating functions (splines) which fit the structural deflections at the chosen structural degrees of freedom, and from which deflections at the interpolated (aerodynamic) points can be determined.

The module will be able to process several splines in one pass. Its primary use is for interpolation for providing a relationship between aerodynamic and structural degrees of freedom.

The Geometry Interpolator Module includes both linear and surface splines. The linear splines consist of uniform beam-torsion members, connected to the structure with rigid arms perpendicular to the axis of the spline (see Figure 5.4-2). In addition, scalar springs may be placed at the grid points. The surface splines consist of uniform plates which may be attached to the grid point through springs. The purpose of the springs is to provide smoothing of the interpolated points by not requiring the spline to go thru all points; see Appendix E for details of how to choose values for the springs. All splines are planar, i.e., the rigid arms to the linear splines all lie in the same plane, and the plates are flat. Within this plane, there are two types of motion. Inplane motion consists of displacements parallel to the plane and rotations about an axis normal to the plane. Out-of-plane motion consists of displacements normal to the plane, and rotations about axes parallel to the plane. Three types of splines are provided:

SO	=	surface, out-of-plane
LO	=	linear, out-of-plane
LI	=	linear, inplane

No inplane surface spline is provided. The only surface inplane interpolation with known solution is a rigid plate, and this gives the same results as a rigid linear spline. The in-plane interpolation is not needed for any currently proposed aerodynamic theories, but is included as a tool for implementation of theories involving motion in the airstream direction.

The desired result of the analysis is a set of interpolation coefficients. The interpolated dependent displacements, u_k, are determined at a set of points whose location is determined by the aerodynamic theory. The structural (independent) displacements, u_g, are grid point displacements in the global coordinate system. The goal of the module is to provide a matrix G_{kg} such that

$$\{u_k\} = [G_{kg}] \{u_g\}$$

If the set of structural points contains only members of the "a" set then this will be written as

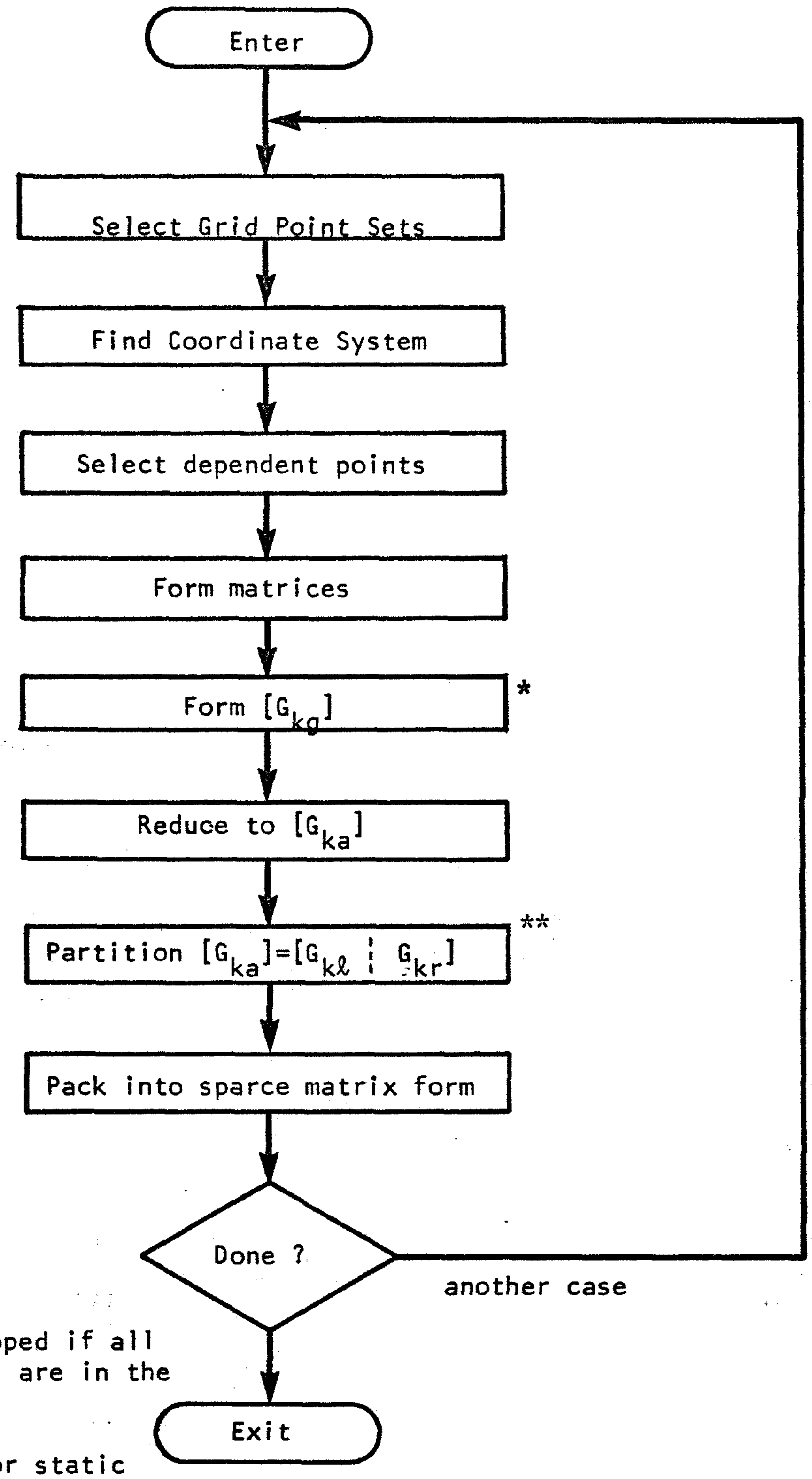
$$\{u_k\} = [G_{ka}] \{u_a\}$$

The "a" set is preferred since it leads to a reduction in matrix algebra, and thus in the time of execution of this module.

There are several ways to analyze splines. These include the three moment method, the stiffness method, and the influence function methods. The influence function methods have been chosen. The advantages include a uniform formulation for beams and plates, ease of interpolation for the "k" points, and the ease of putting springs at the grid point attachments. The disadvantage is the loss of banding in the matrix, thus requiring additional computation. This limitation has been accepted with the understanding that the number of independent degrees of freedom in the "g" set is usually small. A flow chart for the geometry interpolator is shown in figure 5.4-1.

5.4.1 Independent variables; the 'g' set.

The grid points in the independent u_g set for a particular spline are chosen by the user by means of SPLINE and SET data cards. (The components will be chosen by the choice of a value for an attachment spring rate on a SPLINE data card. A zero value implies that the component is not attached.) The first task performed by the program is to compile lists of grid points with the following format:



* May be skipped if all grid points are in the u_a set.

** Required for static analysis only.

Figure 5.4-1

The Geometry Interpolator Module

1.	SID	set of identification numbers
2.	N	number of grid points in the set
3 thru (N+2)		list of grid points

If the SET1 card (see Figure 6-11) is used, the data is directly available. If SET2 (Figure 6-12) is used, a set of inequalities are set up in the basic coordinate system, and all grid points are checked to see which ones satisfy the inequalities (i.e., which grid points lie within the volume defined by the inequalities). For SET2 definition, take the following steps:

Step 1. Locate the corners of the selected aerodynamic element in the basic coordinate system. These are found on the output list of the Aerodynamic Matrix Generator Module. First, find the referenced macro element. Then

$\overline{r_1}$	is the first vertex of subelement	S1.C1
$\overline{r_2}$	is the second vertex of subelement	S2.C1
$\overline{r_3}$	is the third vertex of subelement	S2.C2
$\overline{r_4}$	is the fourth vertex of subelement	S1.C2 .

Step 2. Form outward normal vectors for all surfaces

$$\bar{n}_{12} = \frac{[(\bar{r}_3-\bar{r}_2) \times (\bar{r}_2-\bar{r}_1)] \times (\bar{r}_2-\bar{r}_1)}{|[(\bar{r}_3-\bar{r}_2) \times (r_2-r_1)] \times (r_2-r_1)|}$$

permute indices $1\rightarrow2\rightarrow3\rightarrow4\rightarrow1$ to get $\bar{n}_{23}$, $\bar{n}_{34}$, $\bar{n}_{41}$. A vector $\bar{r}$ is within the cylinder if all of the following inequalities hold.

$$\bar{r}\cdot\bar{n}_{12} \leq r_1\cdot\bar{n}_{12}$$

$$\bar{r}\cdot\bar{n}_{23} \leq r_2\cdot\bar{n}_{23}$$

$$\bar{r}\cdot\bar{n}_{34} \leq r_3\cdot\bar{n}_{34}$$

$$\bar{r}\cdot\bar{n}_{41} \leq r_4\cdot\bar{n}_{41}$$

If $\bar{r}_1 = \bar{r}_2$, or $\bar{r}_4 = \bar{r}_3$, triangular element has been used. For this case, use only three tests.

Step 3. If height limits are given, define

$$\bar{n}_{up} = \frac{(\bar{r}_3-\bar{r}_2) \times (\bar{r}_2-\bar{r}_1)}{|(\bar{r}_3-\bar{r}_2) \times (\bar{r}_2-\bar{r}_1)|} .$$

Then the following inequalities must be satisfied.

$$\bar{r}\cdot\bar{n}_{up} \leq H1$$

$$\bar{r}\cdot\bar{n}_{up} \leq -H2$$

If $r_1=r_2$, use r_4 , r_2 and r_3 to define $\bar{n}_{up}$.

Step 4. If all of the above inequalities are satisfied, then the grid point is in the set.

5.4.2 Spline Coordinate System

The spline coordinate systems are chosen such that surfaces are in the x-y plane and line splines are on the y-axis, see Figure 5.4-2. This is consistent with the convention for flat airfoils, where the x direction is downstream and the z direction is vertical. The displacements u_x, u_y, u_z, θ_x, θ_y, θ_z are chosen to be in the coordinate directions using the right hand rule. Inplane motions consist of u_x, u_y, and θ_z. Out-of-plane motions consist of u_z, θ_x, and θ_y.

The linear splines (beams) have rigid arms, which are perpendicular to the beam and in the x-y plane; hence the rigid arms are parallel to the splines x-axis. These rigid arms are firmly attached to the beam so that their slopes and displacements are equal to those of the beam. The location of the beam axis (i.e., the beam coordinate system) can be chosen so that some arms come out of each side.

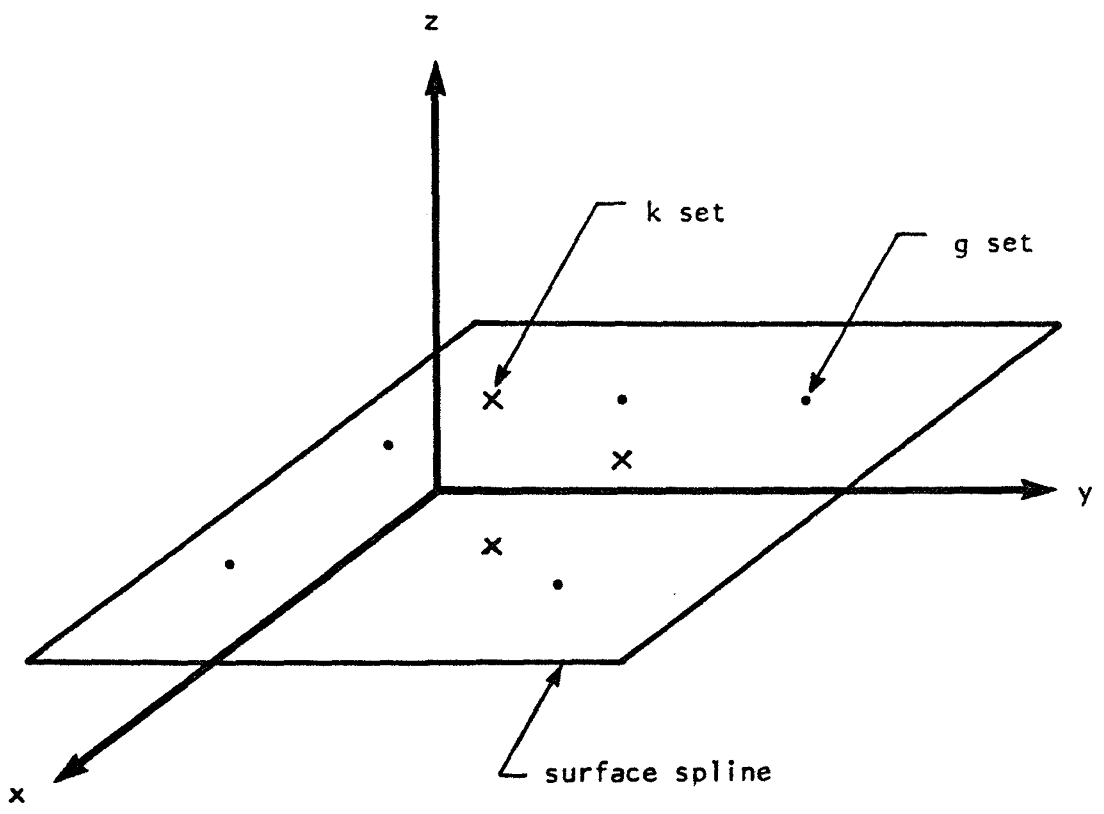
The spline coordinate system may be chosen in two ways. One way is to be specified by the user, in terms of an aero-cell or a given coordinate system. The other way is to choose the coordinate axes (x,y) in the plane of the spline such that the axes coincide with the principal axes of the g-set. The first method would be used if the user desired to fix the location, such as locating a beam spline along a wing spar. The second method is used if it is desired that a beam spline should have points of the "g"-set distributed on both sides. For surface splines, the results will be the same for either choice.

The spline location is defined by the user thru the use of the SPLINE data card, Figures 6-8 thru 6-10. Only one of the fields CAERØ and CID is used, which identifies the aerodynamic element or coordinate system which defines the location of the x-y plane. If the CAERØ method is used, the spline x-y plane will be the plane of the referenced aero macro element. If the CID method is used, it will be the x-y plane of the referenced coordinate system. There are two choices for AXES. If FIXED is chosen, then the x-y axes will agree with those of the defining aero element or coordinate system. If PRINC is chosen, then the x-y axes will be shifted to the center of gravity and principal axes system of the "g"-points (projected onto the plane). The x and y axes of the principal axes system will be within 45 degrees of the x and y axes of the FIXED direction. What is required is the transformation vector and matrix from NASTRAN basic coordinates: $\{r_0\}_i$ is the location of the origin in basic coordinates, and T_i is a 3 x 3 matrix whose columns are the unit vectors of the spline coordinates written in basic. Thus, if x, y, z is the rectangular representation, and $\{r\}_i^T = \lfloor x, y, z \rfloor$ is the spline coordinate vector, then

$$\{r\}_{basic} = \{r_0\}_i + [T_i] \{r\}_i$$

If the AXES are specified as FIXED, then the user has specified that the spline coordinate system is one of the defined coordinate systems. The vector $\{r\}$ and matrix $[T]$ can be found in the CSTM file. If the AXES are listed as "PRINC", then some fitting will be required. This will involve finding the principal axes of a set of points.

If CID, PRINC is specified, find the unit vector $\overline{k}$ (third column of T) and the location, r_0, of the origin in basic for the identified coordinate system. Compute the location of the points projected on the plane from



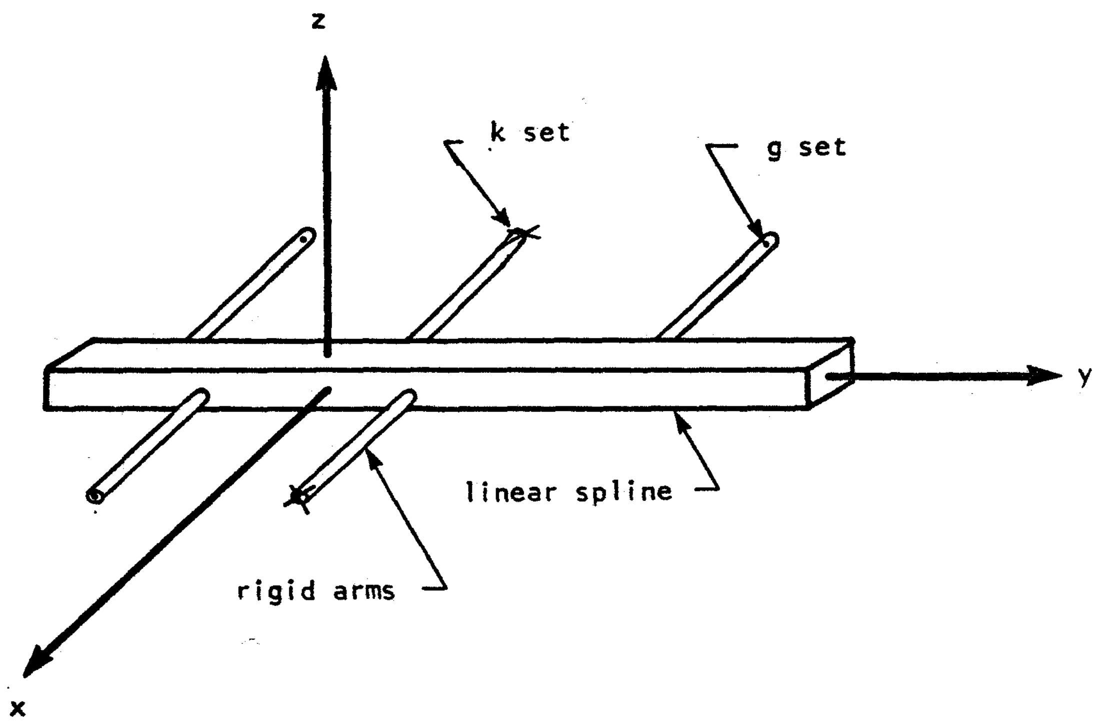


Figure 5.4-2

Splines and their coordinate systems

$$\bar{r}_{proj} = \bar{r} - \bar{k}\,[\bar{k}\,(\bar{r}-\bar{r}_0)] \quad .$$

Use a least squares fit to find a coordinate system which is a principal axis through the c.g. of the points.

If the CAERØ, PRINC is specified, find three distinct points $\bar{r}_1$, $\bar{r}_2$, and $\bar{r}_3$ (or $\bar{r}_1$, $\bar{r}_2$, and $\bar{r}_4$) and form the unit normal

$$\bar{n} = \frac{(\bar{r}_2-\bar{r}_1) \times (\bar{r}_3-\bar{r}_2)}{|(\bar{r}_2-\bar{r}_1) \times (\bar{r}_3-\bar{r}_2)|} \quad .$$

Proceed as in case CID, PRINC, except substituting $\bar{r}_1$ and $\bar{n}$ for $\bar{r}_0$ and $\bar{k}$.

The following algorithm will produce the principal axes of a set of points.

1. Function: Given a list of values for x, y, z, find a transformation to principal axes.

2. Input Data: N (number of points)

x_i, y_i, z_i	i=1,N
i_1, i_2, i_3	components of "i" vector
j_1, j_2, j_3	components of "j" vector.

3. Output

x_0, y_0, z_0	location of c.g.
T_{11}, T_{12}, T_{13},...T_{33}	transformation matrix
I_1, I_2, I_3	principal moments of inertia

4. Method

4.1 Compute c.g. $\left\{\begin{matrix} x_0 \\ y_0 \\ z_0 \end{matrix}\right\} = \frac{1}{N} \sum_{i=1}^{N} \left\{\begin{matrix} x_i \\ y_i \\ z_i \end{matrix}\right\}$

4.2 Compute matrix

$$A = \sum_{i=1}^{N} \begin{bmatrix} (x_i-x_0)^2 & (x_i-x_0)(y_i-y_0) & (x_i-x_0)(z_i-z_0) \\ & (y_i-y_0)^2 & (y_i-y_0)(z_i-z_0) \\ \text{SYM} & & (z_i-z_0)^2 \end{bmatrix}$$

4.3 Find eigenvalues λ_i and eigenvectors ϕ_i of matrix A. Normalize vectors to unit vectors.

4.4 Renumber so that λ_2 is the greatest, and λ_3 is the smallest.

4.5 Reverse the eigenvectors (if necessary) so that

$$\bar{i}\cdot\phi_1 > 0$$

$$\bar{j}\cdot\phi_2 > 0$$

and

$$(\phi_1 \times \phi_2)\cdot\phi_3 > 0 \quad ,$$

where $\bar{i}$ and $\bar{j}$ are input.

4.6 The output x_0, y_0, z_0 are the c.g. location

$$\begin{bmatrix} T_{11} & T_{12} & T_{13} \\ T_{21} & T_{22} & T_{23} \\ T_{31} & T_{32} & T_{33} \end{bmatrix} = \left[\begin{array}{c|c|c} \phi_1 & \phi_2 & \phi_3 \end{array}\right]$$

and the moments of inertia

$$I_i = \lambda_i \qquad i = 1, 2, 3 .$$

5.4.3 Dependent variables; the "k"-set

The dependent variables, u_k, to be included will be determined by the aerodynamic cell geometry (see CAERØ data cards) and the particular aerodynamic theory (which involves the PAERØ data cards). For each aerodynamic cell, a list of locations plus degrees of freedom has been formed in the Aerodynamic Element Generator Module. This function is table driven, so that if new aerodynamic theories are added, the additional degrees of freedom can be easily added.

The assignment of the degrees of freedom of k-set to splines is as follows.

1. Check for type of spline. Degrees of freedom 3, 4, and 5 must be interpolated with an out-of-plane spline. Degrees of freedom 1, 2, and 6 must use an inplane spline.

2. If the degrees of freedom are defined by an aero-cell, two subcases exist.

a. If the aero-cell is referenced by a SET 2 (i.e., via an aero-cell) and a spline of the appropriate type refers to that set, use that spline.

b. Otherwise assign the point to the nearest spline of the appropriate type.

3. If the degrees of freedom are defined by a GRIDK data card, two subcases exist.

a. If a spline is defined of the appropriate type, and is referenced by the GRIDK card, use that one.

b. Otherwise assign the point to the nearest spline of the appropriate type.

If no spline of the appropriate type exists, a fatal message occurs.

5.4.4 Mathematical analysis

The analysis is based upon a set of "influence coefficients" for a free-free spline.* The independent degrees of freedom are the "forces" applied to the splines at the grid points and the "rigid body" displacements. Thus the displacement at any point (in the g-set or the k-set) can be written as a linear function of the grid point forces, f_g, and the "rigid body" displacements, W_r.

$$u = A\, f_g + R\, W_r \tag{1}$$

The displacements u and forces f_g may include rotations as well as translations. The choice of spline types has been limited to a class where both matrices A and R are very easy to generate.

The first step is to solve for the forces and rigid body motions in terms of the displacements at the grid points.

* See Appendix E

$$u_g = A_{gg}\, f_g + R_{gr}\, W_r \tag{2}$$

$$0 = R^T_{gr}\, f_g \tag{3}$$

The first equations are the self (i.e., g-g) terms of the influence equations; the second equations are the equilibrium equations. Next the cross influence (g-k) functions are written.

$$u_k = A_{kg}\, f_g + R_{kr}\, W_r \tag{4}$$

If equations (2) and (3) are formally solved for f_g and W_r, and then eliminated from equation (4).

$$u_k = \left[A_{kg} \;\vdots\; R_{kr}\right] \left[\begin{array}{c|c} A_{gg} & R_{gr} \\ \hline R^T_{gr} & 0 \end{array}\right]^{-1} \left\{\begin{array}{c} u_g \\ \hline 0 \end{array}\right\} \tag{5}$$

$$= \left[G_{kg} \;\vdots\; X\right] \left\{\begin{array}{c} u_g \\ \hline 0 \end{array}\right\} = \left[G_{kg}\right] \left\{u_g\right\} \quad . \tag{6}$$

The matrix X seems to have no importance, so it is discarded. Combining the above, it is seen that G_{kg} can be found by solving

$$\left[\begin{array}{c|c} A_{gg} & R_{gr} \\ \hline R^T_{gr} & 0 \end{array}\right] \left[\begin{array}{c} G^T_{kg} \\ \hline X^T \end{array}\right] = \left[\begin{array}{c} A^T_{kg} \\ \hline R^T_{kr} \end{array}\right] \tag{7}$$

The matrix G_{kg} is expressed in the coordinate system of the spline. The components are transformed to the global coordinate system to complete the task. Thus the three tasks are

1. Form matrices A_{gg}, A_{kg}, R_{gr}, R_{kr}.
2. Solve equation (7) for G_{kg}.
3. Transform to global coordinates.

The A_{gg} and A_{kg} matrix elements are listed in table 5.4-1. The point with index j, with coordinates x_j and y_j are the point of application of the load, and are always associated with a "g" point. The point with index i is the location where the deflection is to be calculated, and may refer to either "g" points or "k" points. Thus the same formulas of table 5.4-1 are used for both A_{gg} and A_{kg}. The values for the rigidities (D, EI, GJ, and AE) and springs (k_x, k_y, k_z, $k_{\theta x}$, $k_{\theta y}$, and $k_{\theta z}$) are obtained from SPLINEi data cards (perhaps using default values which have been supplied). Separate formulas are given depending upon whether the i index and j index refer to translations or rotations. Also separate formulas exist for the three types of splines (S=surface, L=linear, Ø=out-of-plane, 1=inplane). When a zero appears in the table, the term is not available unless the spline is rigid (i.e., $D = \infty$), and a request for these formulas should lead to an error message.

The R_{gr} and R_{kr} rigid body matrices, (table 5.4-2) are simple geometric quantities and are the same for linear and surface splines.

After determining a G_{kg} matrix, all points in the NASTRAN dependent sets (called m for multipoint constraint, s for single point constraint, and o for omit) will be removed using the method of the present SSG2 module. If only points of the a set were used, this step is not needed. In static analysis G_{ka} will be further partitioned into $G_{k\ell}$ and G_{kr}.

The k displacements of the interpolated points must be transformed from the spline coordinates to the aerodynamic coordinates. This can be accomplished via the basic coordinates. Thus if u_k is a three vector (of displacements or rotations) then

$$\left\{ u_k \right\}_{aero} = \left[T_{aero} \right]^T \left[T_{spline} \right] \left\{ u_k \right\}_{spline} ,$$

where T_{aero} and T_{spline} are the coordinate transformations for aero and spline systems.

Finally, the matrix terms must be packed, using NASTRAN routines, into a file of G_{kg} matrices.

$$\begin{Bmatrix} u_z \\ \Theta_x \\ \Theta y \end{Bmatrix}_i = \left[\begin{array}{c:c:c} \dfrac{r_{ij}^2 \ln r_{ij}^2}{16\pi D} + \dfrac{\delta_{ij}}{k_z} & 0 & 0 \\ \hdashline \dfrac{(y_i - y_j)(1 + \ln r_{ij}^2)}{8\pi D} & 0 + \dfrac{\delta_{ij}}{k_{\theta x}} & 0 \\ \hdashline -\dfrac{(x_i - x_j)(1 + \ln r_{ij}^2)}{8\pi D} & 0 & 0 + \dfrac{\delta_{ij}}{k_{\theta y}} \end{array}\right] \begin{Bmatrix} P_z \\ M_x \\ M_y \end{Bmatrix}_j$$

The second and third columns are used for rigid plates only.

$$r_{ij} = (x_i - x_j)^2 + (y_j - y_j)^2$$

$$\delta_{ij} = \begin{cases} 1 \text{ for } i = j \\ 0 \text{ for } i \neq j \end{cases}$$

(a) Free-Free influence functions for SØ

Table 5.4-1

The A matrix terms

$$\begin{Bmatrix} u_z \\ \Theta_x \\ \Theta_y \end{Bmatrix}_i = \left[\begin{array}{c:c:c} \frac{|y_i-y_j|^3}{12EI} - \frac{x_i x_j |y_i-y_j|}{2GJ} + \frac{\delta_{ij}}{k_z} & -\frac{|y_i-y_j|(y_i-y_j)}{4EI} & \frac{x_i|y_i-y_j|}{2GJ} \\ \hdashline \frac{|y_i-y_j|(y_i-y_j)}{4EI} & -\frac{|y_i-y_j|}{2EI} + \frac{\delta_{ij}}{k_{\theta x}} & 0 \\ \hdashline \frac{x_j|y_i-y_j|}{2GJ} & 0 & -\frac{|y_i-y_j|}{2GJ} + \frac{\delta_{ij}}{k_{\theta y}} \end{array}\right] \begin{Bmatrix} P_z \\ M_x \\ M_y \end{Bmatrix}_j$$

(b) Free-Free influence functions (matrix A) for LØ, Flexible Beam

$$\begin{Bmatrix} u_x \\ u_y \\ \Theta_z \end{Bmatrix}_i = \left[\begin{array}{c:c:c} \frac{|y_i-y_j|^3}{12EI} + \frac{\delta_{ij}}{k_x} & \frac{x_i|y_i-y_j|(y_i-y_j)}{4EI} & \frac{|y_i-y_j|(y_i-y_j)}{4EI} \\ \hdashline -\frac{x_i|y_i-y_j|(y_i-y_j)}{4EI} & \frac{|y_i-y_j|}{2AE} - \frac{x_i x_j|y_i-y_j|}{2EI} \frac{\delta_{ij}}{k_y} & -\frac{x_i|y_i-y_j|}{2EI} + \frac{\zeta_{ij}}{k_{\theta z}} \\ \hdashline -\frac{|y_i-y_j|(y_i-y_j)}{4EI} & -\frac{x_j|y_i-y_j|}{2EI} & -\frac{|y_i-y_j|}{2EI} + \frac{\delta_{ij}}{k_{\theta z}} \end{array}\right] \begin{Bmatrix} P_x \\ P_y \\ M_z \end{Bmatrix}_j$$

(c) Free-Free influence functions for LI, Flexible Beam

Table 5.4-1 (Cont'd)

$$\begin{Bmatrix} U_z \\ \Theta_x \\ \Theta_y \end{Bmatrix}_i = \left[\begin{array}{c|c|c} 1 & y_i & -x_i \\ \hline 0 & 1 & 0 \\ \hline 0 & 0 & 1 \end{array}\right] \begin{Bmatrix} U_z \\ \Theta_x \\ \Theta_y \end{Bmatrix}_r$$

(a) For out-of-plane motion

$$\begin{Bmatrix} U_x \\ U_y \\ \Theta_z \end{Bmatrix}_i = \left[\begin{array}{c|c|c} 1 & 0 & -y_i \\ \hline 0 & 1 & x_i \\ \hline 0 & 0 & 1 \end{array}\right] \begin{Bmatrix} U_x \\ U_y \\ \Theta_z \end{Bmatrix}_r$$

(b) For inplane motion

Table 5.4-2

The R matrix terms

5.5 Aerodynamic Matrix Generator

5.5.1 Introduction

The Aerodynamic Matrix Generator Module will evaluate a matrix of aerodynamic coefficients in the aerodynamic coordinate system and the transformation matrices needed to convert these to the interpolated structural coordinate system. The module is designed to be able to do the calculation for a wide variety of aerodynamic theories, and to accept matrices prepared exterior to the program if necessary.

More than one theory may be used in one computer run. This allows (for example) the use of a lattice theory for the wing and piston theory for the tail, or uncoupled lattice theories on different areas. It assumes a finite aerodynamic element approach.

Parameters, such as Mach number m, and reduced frequency k, will be used in the formulation of the matrices. One basic assumption is that for some problems the most efficient way to evaluate the aerodynamic matrices as functions of the parameters is to

1. First evaluate at a chosen set of values of (m,k).
2. Then interpolate to the desired value of (m,k).

This method will be useful for the subsonic doublet lattice method where much time is spent computing the matrix elements and decomposing the matrix. The method is not desirable for a simple piston theory, and thus parametric interpolation will be a choice of methods, and not the only method available.

This module will be designed to accept new aerodynamic theories.

As much as possible, the format is chosen such that new theories can be added to the list with a minimum of labor.

If a restart (i.e. secondary NASTRAN run) is made, as many of the matrices as possible will be taken from the checkpoint tape. Thus if additional parameters are added to the list, and some parameters are deleted from the list, only the matrices for the new parameters will be calculated. This decision is based upon the estimate that much time is required to compute the A_{jj} matrix of the doublet lattice theory.

5.5.2 Choice of Aerodynamic Theories

The choice of which aerodynamic theories to implement will depend upon the amount of effort required and the desires of the users. The theories can be divided into two types.

a. Local theories (Newtonian, Strip, Piston)

b. Interaction theories (Doublet lattice, Kernel function)

The local theories are, in general, easy to implement and present no great difficulty; however they neglect some important aerodynamic effects and are thus not sufficient. The interaction theories have been developed only for sinusoidal (or steady) motion, and are of two basic types.

a. source (one sided, e.g. Mach box)

b. doublet or vortex (e.g. doublet lattice)

The source types are characterized by the need to introduce a diaphragm, and are of interest primarily in supersonic flow.

The doublet type has been developed primarily for subsonic flow; however it appears that it can be extended to supersonic flow.

A decision has been made to not implement the theories which require a diaphragm. This was based upon the following:

a. Alternate methods appear feasible.

b. The diaphragm requires the introduction of additional degrees of freedom, which change as a function of Mach Number, making the implementation more difficult.

The theories which are recommended for implementation are shown in Table 5.5A. Additional discussion is presented in Appendix G. Specific recommendations for each theory are as follows:

1. Modified Newtonian Theory

A Newtonian Theory using modified stagnation pressure coefficients, and which can be applied at large angles of attack, is recommended.

2. Piston Theory

Third order Piston Theory including trim angle of attack effects should be used. It is valid for arbitrary camber distributions and control surface configurations.

3. Modified Strip Theory

The NASTRAN version of Strip Theory should include a parabolic camber mode and an aerodynamically balanced control surface (with tab). Modifications should include an arbitrary circulation function, variation of the local lift curve slope, variation of the aerodynamic center location, and variation of the "three-quarter chord" neutral point. All modifications will be under user control via input data tables. Default values corresponding to 2-dimensional incompressible flow theory will be provided.

4. Doublet-Lattice Method

The Doublet-Lattice Method will accommodate arbitrary configurations of interferring surfaces in subsonic flow. It may be regarded as a typical method in which the aerodynamic degrees of freedom are motions and pressure at user-selected fixed points (finite element viewpoint). Provision will be made for two planes of symmetry or anti-symmetry.

5. Supersonic Doublet-Lattice Method

It is recommended that the doublet-lattice method be extended to the supersonic case. The Kernel function has been derived, but the method is undeveloped.

6. Body Interaction with Surfaces

The use of doublets along a line to represent the lift and side force on bodies is being developed (see Rodden, Geising, and Kálmán, reference 5.5-1 at the end of this section) to include interference with lifting surfaces. The method is considered to be an extension of the doublet-lattice method.

5.5.3 Aerodynamic Degrees of Freedom

Two sets of degrees of freedom will be introduced. Each aerodynamic theory has a set of degrees of freedom which are best for that theory. In addition, there may be another set of degrees of freedom which is best suited for interpolation to the structure. Both sets and the transformation matrices will be used.

The "j"-set is a set of degrees of freedom best suited for aerodynamics. The displacements w_j, may include downwash velocities, pitching velocities, angles of attack, camber motions, or any other dimensional or dimensionless variable. The forces f_j^a, may include pressures, moments, generalized forces, may have dimensions, or be dimensionless, and may act at different locations than the displacements w_j. What is required is that there is a non-singular matrix A_{jj}, determined by the aerodynamic theory such that

$$w_j = \frac{1}{q} A_{jj} f_j^a \tag{1}$$

The matrix A_{jj} would be computed directly by the doublet lattice theory, and its inverse would be computed directly by the local theories. The terms of A_{jj} (or A_{jj}^{-1}) will depend upon Mach number m, and reduced frequency k, and perhaps upon other parameters. Thus the "j"-set defines degrees of freedom which will depend upon which areodynamic theory is used.

The "k"-set is intermediate between the structural degrees of freedom and the "j"-set. The displacement and forces must be consistent, such that the product of two corresponding components of force and displacement represents work. The displacements must be linear or rotational motions on points of the structure which can be found by interpolation from the structural degrees of freedom, (see Section 5.4, Geometry Interpolator).

The transformation from the "k"-set to the "j"-set will, in general, be singular, with the "j"-set displacements dependent upon the "k"-set by

$$w_j = D_{jk} u_k \tag{2}$$

$$F_k^a = S_{kj} f_j^a \tag{3}$$

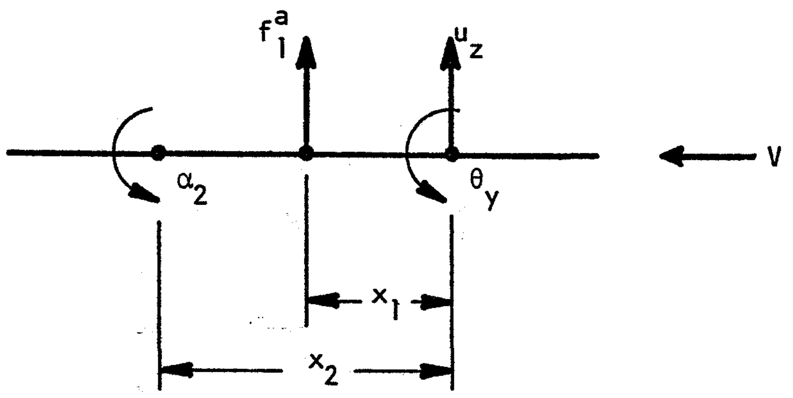
As an example to illustrate $[D_{jk}]$ and $[S_{kj}]$ let

$\{u_k\} = \begin{Bmatrix} u_z \\ \theta_y \end{Bmatrix}$ be the normal translation and rotation at the control point.

$\{w_j\} = \{\alpha_2\}$ be the angle of attack at point (2)

$\{f_j^a\} = \{f_1^a\}$ be the normal pressure at point (1)

$\{F_k^a\} = \begin{Bmatrix} F_z^a \\ M_y^a \end{Bmatrix}$ be the normal force and moment at the control point as illustrated below



Then

$$\alpha_2 = \theta_y - \frac{\dot{u}_z}{V} + \frac{x_2}{V}\dot{\theta}_y = \theta_y - i\frac{k}{b}u_z + i\frac{x_2 k}{b}\theta_y \qquad (4)$$

$$F_z^a = S \cdot f_1^a \qquad (5)$$

$$M_y^a = - Sx_1 f_1^a \tag{6}$$

so that

$$[S_{kj}] = \left[\begin{array}{c} S \\ \hdashline -Sx_1 \end{array}\right] \tag{7}$$

$$[D_{jk}] = \left[\begin{array}{c|c} -i\,\frac{k}{b} & 1 + i\,\frac{x_2 k}{b} \end{array}\right] \tag{8}$$

5.5.4 Generation of Aerodynamic Matrices

The code to compute the S_{kj}, A_{jj} and D_{jk} matrices will be different for each aerodynamic theory. The format of the output will conform to the following rules.

a. There is one real matrix S_{kj}, which is usually "diagonal", but need not be so. It is "diagonal" in the sense that non-zero elements are in rectangular partitions along a sloped diagonal of a rectangular matrix.

b. There is one complex matrix D_{jk}, which is the value of D_{jk} for reduced frequency, k = 1. For other values of reduced frequency

$$D_{jk}\,(k) = \mathrm{Re}\left[D_{jk}\,(k=1)\right] + \mathrm{Im}\left[D_{jk}\,(k=1)\right]\cdot k \tag{9}$$

D_{jk} is also usually "diagonal".

c. The aerodynamic matrices are either a single output, or a list of outputs. The list format is required if more than one theory is specified or if more than one set of (m,k) values is specified. The matrices will be either A_{jj}^{-1}, or the decomposition products of A_{jj}, whichever is directly computed by the theory.

The output is in a form suitable for use by the Aerodynamic Matrix Processor, or in the case of single output only, it can be used in any matrix operation. The calculation should be efficient for both simple local theories and for complex interaction theories requiring parametric interpolation.

The basic flow chart is shown in Figure 5.5-1. When symmetry is specified on the AERØ bulk data card, some degrees of freedom on the plane of symmetry may need to be deleted. For example, when using doublet lattice, remove all aero degrees of freedom in the plane of symmetry for a symmetric condition, and make no changes for antisymmetric. This operation is like SPC. The S_{kj} and D_{jk} matrices are the result of adding the matrices computed for each theory, and are independent of (m,k). Since the internal indices are strictly increasing (see Section 5.2.2 for internal indices of k and j sets), these additions are actually appending to the end of a compiled matrix. A list of (m,k) values for which the matrices are to be compiled has been user supplied on MKAERØ (or AERØ) bulk data cards. The details of the computation of A_{jj} will depend upon the choice of theory; see Section 5.5.5 for the details of the subsonic doublet lattice method. The format of the output will either be a matrix list, or a matrix depending upon the value of a parameter in the module call.

5.5.5 Details for the subsonic doublet lattice theory

The subsonic doublet lattice method has been chosen to illustrate how the Aerodynamic Matrix Generator Module will be written. This method has been chosen because it contains all of the features which need to be illustrated. The mathematics required for the method have been detailed in the literature by Rodden, Geising and Kalman, (see Reference 5.5-1).

There are many algebraic steps, which are summarized in Table 5.5B.

The choice of the u_k degrees of freedom for the Doublet-Lattice method is somewhat arbitrary. It is possible to use one normal displacement and a rotation per cell, or to use two normal displacements at different points. The results would differ only if there were substantial curvature within the aerodynamic cell. The method of two displacements is chosen since there is a distinct advantage for plotting, and there is a chance of inaccuracy when interpolating slopes at points near the structural grid points. Another choice is whether to use fixed locations (say, the ¼ chord, and 3/4 chord of a box at midspan) or points which depend upon parameters (the center of pressure and the downwash center). A decision was made to use the latter, since this will reduce the density of the S_{kj} and D_{jk} matrices, and it can be guaranteed that these points will not coincide (even in the supersonic case). Thus for doublet lattice, the following are chosen.

(k-set)

1. Normal displacement at center of pressure.
2. Normal displacement at downwash center.

(j-set)

1. Normal-wash and pressure coefficient.

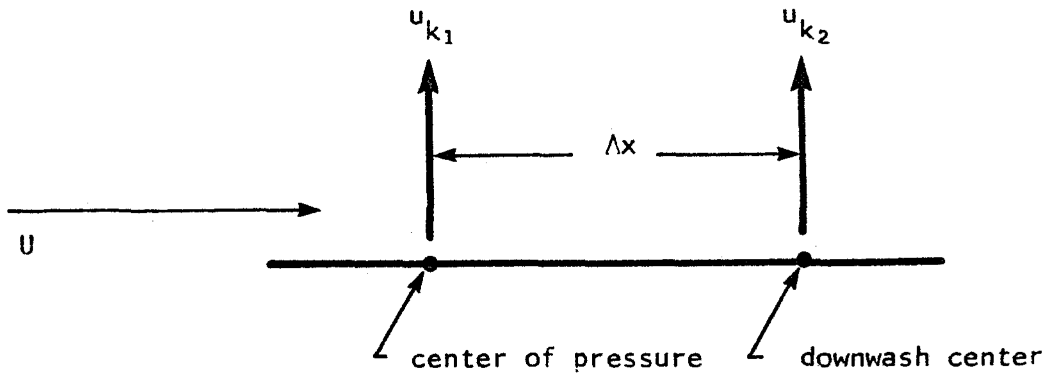
For this choice, the S_{kj} matrix for one cell is

$$\begin{Bmatrix} F^a_{k1} \\ F^a_{k2} \end{Bmatrix} = \begin{bmatrix} S \\ \hdashline 0 \end{bmatrix} \begin{Bmatrix} f^a_j \end{Bmatrix} \tag{10}$$

where S is the cell area. The D_{jk} matrix is given by

$$\left\{ w_j \right\} = \left[\frac{1}{\Delta x} \;\vdots\; -\left(\frac{1}{\Delta x} + i \frac{k}{b} \right) \right] \left\{ \begin{array}{c} u_{k_1} \\ \hdashline u_{k_2} \end{array} \right\} \tag{11}$$

where Δx is the distance from the center of pressure to the downwash center, and b is the reference length. u_{k1} and u_{k2} are in the positive coordinate system direction, while w_j is in the opposite direction.



The S_{kj} matrix will be of order 2N by N, where N is the number of boxes. The 2 x 1 submatrices along the "diagonal" are given by Eq. (10), and all of the rest of the matrix is zero. The area S is output by the Aerodynamic Element Generator. The D_{jk} matrix is N by 2N. The 1 x 2 submatrices along the diagonal are given by Eq. (11), with k = 1. Note that

$$\Delta x = (\text{cell chord}) \cdot (x1 - x0) \tag{12}$$

where the cell chord is output by the aerodynamic element generator, and x1 and x0 are on the PAERØ data card (default = .75 and .25). The reference length, b, is on the AERØ data card.

The A matrix is an N x N, and must be computed and decomposed for each (m,k) pair. The flow-chart for this operation (which is one box in Figure 5.5-1) is shown in Figure 5.5-2. The basic calculations to compute A_{jj}.

are shown in Table 5.5B, in the order in which they must be evaluated. All of the needed data is found on the data lists from the Aerodynamic Element Generator and the PAERØ (x0, x1) and AERØ data cards.

Table 5.5A

Recommended Aerodynamic Theories

Theory	Range of Application	Status
1. Modified Newtonian	a. $M >> 1$, $M\alpha >> 0$ b. $M > 0$, $\alpha > \alpha_{stall}$	Fully developed
2. Piston Theory	a. $M >> 1$, $k > 0$ b. $M > 0$, $Mk >> 1$	Fully developed
3. Modified Strip Theory Including Camber, Control Surface, and Tab.	$M > 0$, $k > 0$	Fully developed
4. Subsonic Doublet Lattice Method	$M < 1$, $k > 0$	Well developed
5. Supersonic Doublet Lattice Method	$M > 1$, $k > 0$	Undeveloped
6. Body Interaction with Surfaces	$M < 1$, $k > 0$	Being developed

$$\beta = (1-M^2)^{\frac{1}{2}} \tag{1}$$

$$\Delta x_s = \text{chord of sending box} \div b \tag{2}$$

$$e = \text{semi-width of sending box} \div b \tag{3}$$

$$\gamma_s = \text{dihedral angle of sending box} \tag{4}$$

$$\lambda_s = \text{sweep back angle of sending box} \tag{5}$$

$$\bar{x} = (x_r - x_s)/b \tag{6}$$

$$\bar{y} = \left[(y_r - y_s)\cos\gamma_s + (z_r - z_s)\sin\gamma_s\right]/b \tag{7}$$

$$\bar{z} = \left[-(y_r - y_s)\sin\gamma_s + (z_r - z_s)\cos\gamma_s\right]/b \tag{8}$$

$$\gamma_r = \text{dihedral angle of receiver box} \tag{9}$$

$$T_1 = \cos(\gamma_s - \gamma_r) \tag{10}$$

do for $\bar{\eta} = -e, 0, +e$

$$r_1(\bar{\eta}) = \left[(\bar{y} - \bar{\eta})^2 + \bar{z}^2\right]^{\frac{1}{2}} \tag{11}$$

$$T_2^*(\bar{\eta}) = \bar{z}\left[\bar{z}\cos(\gamma_s - \gamma_r) + (\bar{y} - \bar{\eta})\sin(\gamma_s - \gamma_r)\right] \tag{12}$$

$$R(\bar{\eta}) = \left[(\bar{x} - \bar{\eta}\tan\lambda_s)^2 + \beta^2 r_1^{\ 2}\right]^{\frac{1}{2}} \tag{13}$$

$$u_1(\bar{\eta}) = (MR - \bar{x} + \bar{y}\tan\lambda_s)/\beta^2 r_1 \tag{14}$$

$$k_1(\bar{\eta}) = \frac{r_1}{b}\,k \tag{15}$$

$$I_1(\bar{\eta}) = I_1\,(u_1, k_1) \tag{16}$$

$$I_2(\bar{\eta}) \quad I_2\,(u_1, k_1) \tag{17}$$

See Appendix A of ref. 5.5-1 for details of how to evaluate the approximate value of the functions using finite series approximation.

$$K_1(\bar{\eta}) = -I_1 - \exp(-ik_1u_1)Mr_1/R(1+u_1^{\ 2})^{\frac{1}{2}} \tag{18}$$

$$K_2(\bar{\eta}) = 3I_2 + ik_1\exp(-ik_1u_1)M^2r_1^{\ 2}/R^2(1+u_1^{\ 2})^{\frac{1}{2}} + \exp(-ik_1u_1)Mr_1 \left[(1+u_1^{\ 2})\beta^2r_1^{\ 2}/R^2 + 2 + Mr_1u_1/R\right]/R(1+u_1^{\ 2})^{\frac{3}{2}} \tag{19}$$

$$K_{10}(\bar{\eta}) = -1 - (\bar{x} - \bar{\eta}\tan\lambda_s)/R \tag{20}$$

Table 5.5B

Formulas to be evaluated for the doublet lattice method

(From Rodden, Giesing and Kálmán)

do for $\bar{\eta} = -e, 0, +e$

$$K_{20}(\bar{\eta}) = 2 + (\bar{x} - \bar{\eta} \tan \lambda_s)(2 + \beta^2 r_1{}^2/R^2)/R \tag{21}$$

$$P_1(\bar{\eta}) = \left\{ K_1 \exp\left[-ik\ (\bar{x} - \bar{\eta} \tan \lambda_s)\right] - K_{10} \right\} T_1 \tag{22}$$

$$P_2(\bar{\eta}) = \left\{ K_2 \exp\left[-ik\ (\bar{x} - \bar{\eta} \tan \lambda_s)\right] - K_{20} \right\} T_2{}^* \tag{23}$$

$$A_1 = \left[P_1(-e) - 2\,P_1(0) + P_1(e) \right] / 2e^2 \tag{24}$$

$$B_1 = \left[P_1(e) - P_1(-e) \right] / 2e \tag{25}$$

$$C_1 = P_1(0) \tag{26}$$

$$A_2 = \left[P_2(-e) - 2P_2(0) + P_2(e) \right] / 2e^2 \tag{27}$$

$$B_2 = \left[P_2(e) - P_2(-e) \right] / 2e \tag{28}$$

$$C_2 = P_2(0) \tag{29}$$

$F = \dfrac{1}{|\bar{z}|} \tan^{-1} \dfrac{2e|\bar{z}|}{\bar{y}^2 + \bar{z}^2 - e^2}$, see ref 5.5-1 if $|\bar{z}|$ is near zero. (30)

D_{0rs} is evaluated using steady vortex, with Prandtl-Glauert transformation. (31)

$$D_{1rs} = \left(\frac{\Delta x_s}{8\pi}\right) \left\{ \left[(\bar{y}^2 - \bar{z}^2)\, A_1 + \bar{y} B_1 + C_1 \right] F + (\tfrac{1}{2} B_1 + \bar{y} A_1) \log \frac{(\bar{y} - e)^2 + \bar{z}^2}{(\bar{y} + e)^2 + \bar{z}^2} + 2eA_1 \right\} \tag{32}$$

$$\begin{aligned} D_{2rs} = \frac{\Delta x_S}{16\pi \bar{z}^2} \Bigg\{ & \left[(\bar{y}^2 + \bar{z}^2) A_2 + \bar{y} B_2 + C_2 \right] F + \frac{1}{(\bar{y}+e)^2 + \bar{z}^2} \\ & \left(\left[(\bar{y}^2 + \bar{z}^2)\bar{y} + (\bar{y}^2 - \bar{z}^2) e \right] A_2 + (\bar{y}^2 + \bar{z}^2 + \bar{y}e)\, B_2 + (\bar{y} + e) C_2 \right) \\ & - \frac{1}{(\bar{y}-e)^2 + \bar{z}^2} \left(\left[(\bar{y}^2 + \bar{z}^2)\bar{y} - (\bar{y}^2 - \bar{z}^2) e \right] A_2 + (\bar{y}^2 + \bar{z}^2 - \bar{y}e) \right. \\ & \left. B_2 + (\bar{y} - e) C_2 \right) \Bigg\} \end{aligned} \tag{33}$$

$$(A_{jj})_{rs} = D_{0rs} + D_{1rs} + D_{2rs} \tag{34}$$

Table 5.5B
(Cont'd)

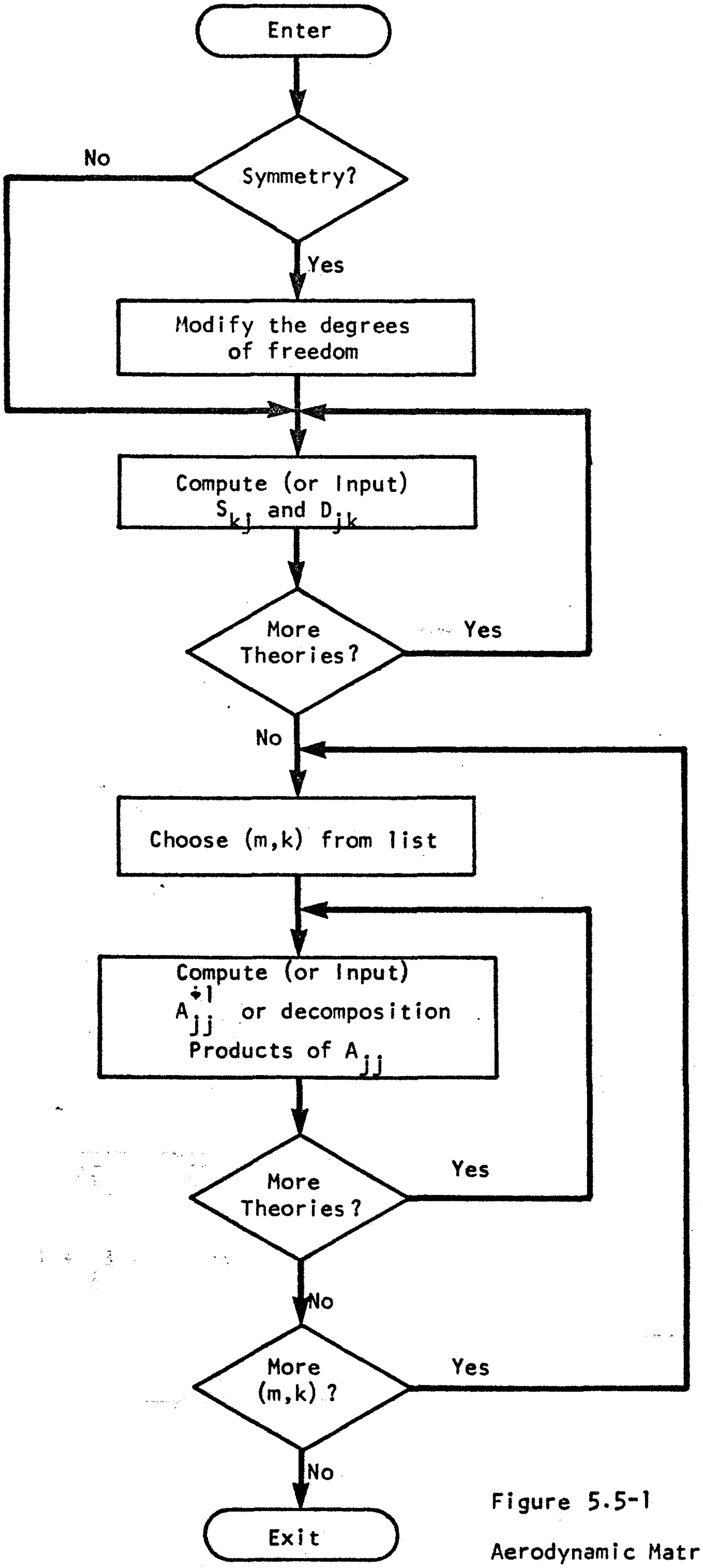

Enter
Symmetry?
No
Yes
Modify the degrees of freedom
Compute (or Input) S_{kj} and D_{jk}
More Theories?
Yes
No
Choose (m,k) from list
Compute (or Input) A_{jj}^{-1} or decomposition Products of A_{jj}
More Theories?
Yes
No
More (m,k) ?
Yes
No
Exit

Figure 5.5-1

Aerodynamic Matrix Generator

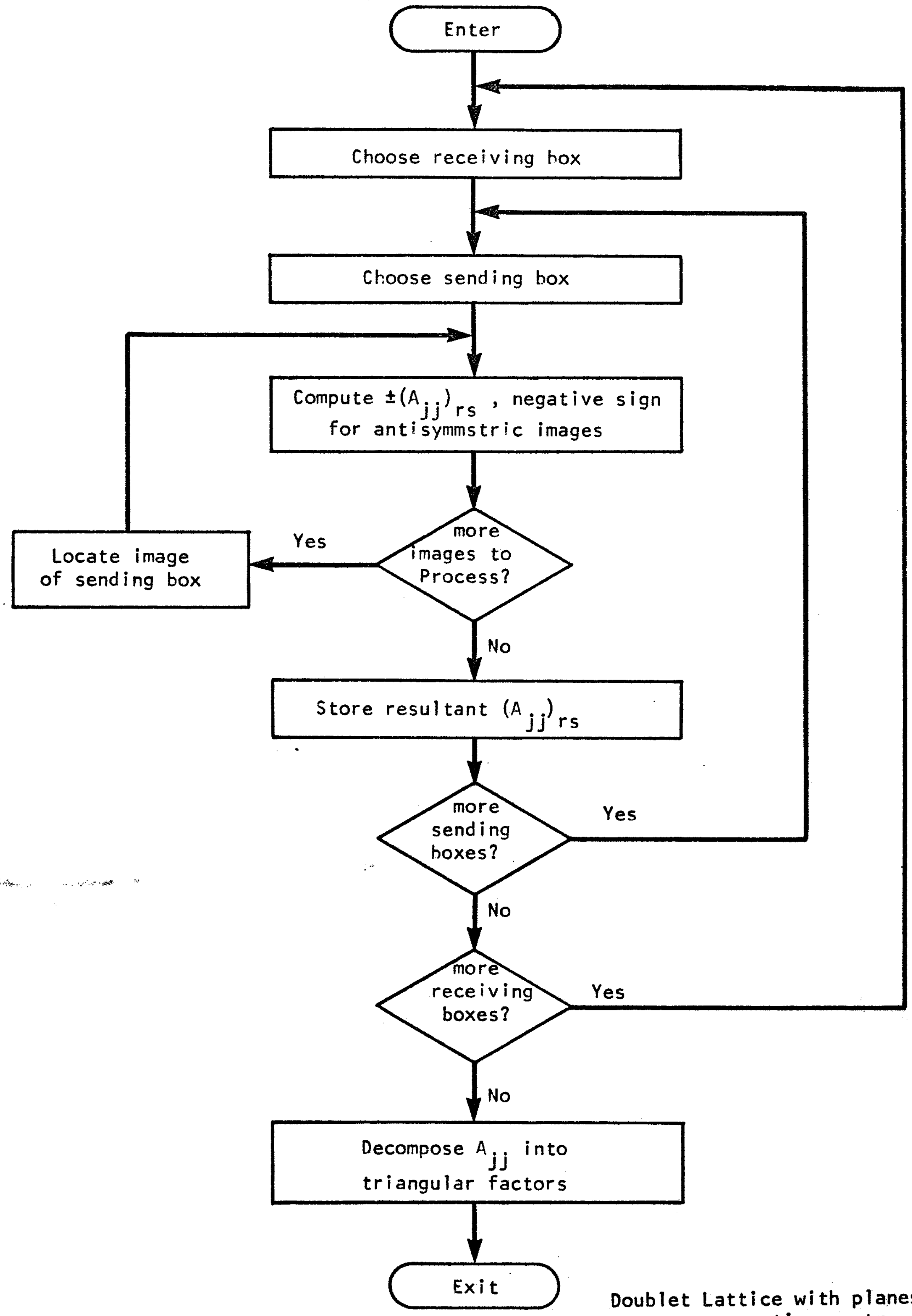

Enter
Choose receiving box
Choose sending box
Compute $\pm(A_{jj})_{rs}$, negative sign for antisymmstric images
more images to Process?
Yes
Locate image of sending box
No
Store resultant $(A_{jj})_{rs}$
more sending boxes?
Yes
No
more receiving boxes?
Yes
No
Decompose A_{jj} into triangular factors
Exit

Figure 5.5-2 Doublet Lattice with planes of symmetry or anti-symmetry, Calculation of A_{jj} for a single Mach number and reduced frequency.

REFERENCES
for Section 5.5

1. Rodden, W. P., Giesing, J. P., and Kálmán, T. P., "New Developments and Applications of the Subsonic Doublet-Lattice Method for Nonplanar Configurations," Paper presented to AGARD Symposium on Unsteady Aerodynamics for Aeroelastic Analyses of Interferring Surfaces, Tønsberg, Oslofjorden, Norway, 3-4 November 1970.

5.6 Aerodynamic Matrix Processor

The purpose of the Aerodynamic Matrix Processor is to produce aerodynamic matrices referred to structural or modal coordinates. It does this by using the aerodynamic matrices produced by the Aerodynamic Matrix Generator, the interpolation matrices produced by the Geometry Interpolator, and the modal transformation produced by the Real Eigenvalue Analysis Module. The Aerodynamic Matrix Processor will be able to act upon the lists of matrices associated with parameters, or with matrices associated with a single parameter value.

The matrices required for the direct approach are:

$$[Q_{aa}] = [G_{ka}]^T[S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{ka}] \qquad (1)$$

$$[Q_{ae}] = [G_{ka}]^T[S_{kj}][A_{jj}]^{-1}[D_{je}] \qquad (2)$$

$$[Q_{aj}] = [G_{ka}]^T[S_{kj}][A_{jj}]^{-1} \qquad (3)$$

Q_{aa} is needed in all problems, Q_{ae} is needed when extra points have been referenced by the Aerodynamic Matrix Generator, and Q_{aj} is needed when gust forces are introduced. The corresponding three equations for the modal approach are

$$[Q_{ii}] = [\Phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{ka}][\Phi_{ai}] \qquad (4)$$

$$[Q_{ie}] = [\Phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1}[D_{je}] \qquad (5)$$

$$[Q_{ij}] = [\Phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1} \qquad (6)$$

The data for A_{jj}^{-1} may be in the form of decomposition products.

For data recovery, when the aerodynamic matrices have been interpolated, it will also be necessary to interpolate $[A_{jj}]^{-1}$ before multiplying by the downwash. This is needed to find the pressures due to structural displacements, extra points and gusts. The matrix A_{jj}^{-1} will be formed from the decomposition products only when required for output, since this is a time consumming job. Thus if a no aerodynamic data recovery is made on a NASTRAN run, and the job is restarted to request output, this Aerodynamic Matrix Processor Module will have to be re-executed to produce A_{jj}^{-1}.

The Aerodynamic Matrix Processor will compute a list of desired matrices for each required data block. The decision of whether to compute the direct or modal form will depend upon whether the Φ matrix is listed as an input or has been purged in the DMAP module call. The choice of which of the three matrices from equation 1-3 or 4-6 to compute will depend upon the status of the output data blocks in the DMAP module call. In all cases, the order of calculation should be from left to right. The order of the matrices on the output lists (when there is more than one m,k pair) is the same as the input list.

5.7 Aerodynamic Matrix Interpolator

The Aerodynamic Matrix Interpolator Module is used to interpolate the aerodynamic matrices as functions of the parameters Mach number (m) and reduced frequency (k). The purpose of interpolation is to save time in computing aerodynamic matrices; thus it should only be used with theories like doublet lattice, which require much effort to compute and reduce the matrices.

The method to be described involves the use of linear splines. An alternative application of surface splines is described in Appendix J. A two-dimensional lattice of these splines is used, where the independent coordinates are m and k, see Figure 5.7-1. The computation is divided into three phases. Some aerodynamic calculations (for example, solving flutter with the p-k method) will require the evaluation of the aerodynamic matrices for several values of one parameter (say, k) and fixed values of the other parameter (say, m). The algorithm used will involve looping, with one parameter (say, k) changing in the inner loop. The module will be programmed such that either parameter may be varied in the inner loop. The three phases of the calculation are:

1. Preparation. Much of the calculation, as will be seen later, needs to be done only once. These calculations are in module AMI1 (Aerodynamic Matrix Interpolator-1), which will be placed in the algorithm just after the Aerodynamic Matrix Processor.

2. Outer Interpolation. Those calculations involved in interpolation using the first parameter will be in AMI2. This module should be placed in the outer loop of the algorithm.

3. Inner Interpolation. The final interpolation will be done in AM13. This calculation should be efficient, since it is likely to occur most often. This module will be placed in the inner loop.

This arrangement can also be used if there is only one parameter (for example, aerostatics in which the only parameter is Mach number). Also, the parameters need not be Mach number and reduced frequency. Reynolds number could be substituted as far as the Matrix Interpolator Module is concerned (of course, the Aerodynamic Matrix Generator Module would have to be programmed to output matrices at several Reynolds numbers).

5.7.1 Spline Interpolation

The one-dimensional spline is a classical beam, of uniform property and infinite length. Analysis by the three-moment method is known to be very efficient, and will be used.

To spline a function $F(x)$, given $F(x_i) = F_i$ for $i = 1, N$, divide the abcissa x into the N-1 closed intervals and two infinite end intervals (See Figure 5.7-2). The slope at the right end of a closed interval is given by

$$\left(\frac{dF}{dx}\right)_i = \frac{F_i - F_{i-1}}{x_i - x_{i-1}} + \frac{1}{6}\left(x_i - x_{i-1}\right)\left(2M_i + M_{i-1}\right) \qquad (1)$$

$$M_i = (d^2F/dx^2)_{x=x_i}$$

The slope at the left end of the interval is given by

$$\left(\frac{dF}{dx}\right)_i = \frac{F_{i+1} - F_i}{x_{i+1} - x_i} - \frac{1}{6}\left(x_{i+1} - x_i\right)\left(M_{i+1} + 2M_i\right) \qquad (2)$$

Eliminating the slope $(dF/dx)_i$ between (1) and (2), we get the three-moment equation:

$$\frac{(x_{i+1}-x_i)(M_{i+1}+2M_i) + (x_i-x_{i-1})(2M_i+M_{i-1})}{6} = \frac{F_{i+1}-F_i}{x_{i+1}-x_i} - \frac{F_i-F_{i-1}}{x_i-x_{i-1}} \qquad (3)$$

Equation (3) for $i = 2 \ldots (N-1)$, along with the end conditions $M_1 = M_N = 0$, gives N-1 equations in N-1 unknowns $M_2 \ldots M_{N-1}$. After solving for the moments M_i, either equation (1) or (2) can be used to solve for the slopes. An efficient algorithm for solving these equations is shown in Figure 5.7-3.

The function F(x) can be written in a closed interval

$$x_L \leq x \leq x_U$$

$$F(x) = (1-3\bar{x}^2 +2\bar{x}^3)F(x_L) + (3\bar{x}^2 -2\bar{x}^3)F(x_U) + (x_L-x_U)(\bar{x}-2\bar{x}^2+\bar{x}^3) \frac{dF}{dx}(x_L)$$
$$-(x_L-x_U)(\bar{x}^2-\bar{x}^3) \frac{dF}{dx}(x_U) \qquad (4)$$

$$\bar{x} = \frac{x-x_L}{x_U-x_L}$$

In an open interval (the right, for example)

$$x_L < x < \infty$$

$$F(x) = F(x_L) + (x-x_L) \frac{dF}{dx}(x_L) \qquad (5)$$

5.7.2 Use of Splines for two-dimensional interpolation

Splines may be used to interpolate in a two dimensional table. Assume that a function $F(m,k)$ is known for a set of values of m and k. For example, F may be a term in an aerodynamic matrix, m the mach number and k the reduced frequency. The problem is to compute a value of $F(m,k)$ for other values of (m,k) which may even lie outside the range of the data; i.e. involve extrapolation.

The first task is to compute a list of all the values for m and k. Call these $m_i (i=1, I)$ and $k_j (j=1, J)$. Then list all of the known values of $F_{ij} = F(m_i, k_j)$. If the value is unknown, a special symbol indicating "unknown" is used.

Second, construct linear splines through the known data points, for fixed $k_j (j=1, J)$, which will produce tentative values for the unknown ordinates F_{ij}, and values of the slopes $(\partial F/\partial m)_{ij}$.

Next construct linear splines through the known data points, for fixed $m_i (i=1, I)$ which will produce other tentative values for the unknown ordinates F_{ij}, and values for the slopes $(\partial F/\partial k)_{ij}$.

The values for the ordinate at the (m_i, k_j) points where F was not given will be taken as the average of the two tentative values. Thus, at every point (m_i, k_j) $(i=1, I)$, $(j=1, J)$; the values F_{ij}, $(\partial F/\partial m)_{ij}$, and $(\partial F/\partial k)_{ij}$ are now known.

Let S stand for either m or k. Define interpolating functions $f_L(S)$, $f_U(S)$, $g_L(S)$ and $g_U(S)$ which depend upon the end points of the interval. These functions are cubic polynomials associated with beam interpolation.

Let the interval be $S_L < S < S_U$ where S_L (lower limit) may be minus infinity or S_U (upper limit) may be plus infinity. Then

	for S_U = finite	for $S_U = +\infty$
for S_L = finite	$f_L = 3\left(\frac{S_U-S}{S_U-S_L}\right)^2 - 2\left(\frac{S_U-S}{S_U-S_L}\right)^3$ $f_U = 3\left(\frac{S-S_L}{S_U-S_L}\right)^2 - 2\left(\frac{S-S_L}{S_U-S_L}\right)^3$ $g_L = (S_U-S_L)\left[\left(\frac{S_U-S}{S_U-S_L}\right)^2 - \left(\frac{S_U-S}{S_U-S_L}\right)^3\right]$ $g_U = -(S_U-S_L)\left[\left(\frac{S-S_L}{S_U-S_L}\right)^2 - \left(\frac{S-S_L}{S_U-S_L}\right)^3\right]$	$f_L = 1$ $f_U = 0$ $g_L = S-S_L$ $g_U = 0$
for $S_L = -\infty$	$f_L = 0$ $f_U = 1$ $g_L = 0$ $g_U = -(S_U-S)$	Not allowed

The interpolated value of the function can be found in any interval, $m_L \leq m \leq m_U$, $k_L \leq k \leq k_U$, by

$$F(m,k) = \sum_{i=L,U} \sum_{j=L,U} \left[F(m_i,k_j)\ f_i(m)\ f_j(k) + \frac{\partial F}{\partial m}(m_i,k_j)\ g_i(m)\ f_j(k) + \frac{\partial F}{\partial k}(m_i,k_j)\ f_i(m)\ g_j(k) \right] \quad (6)$$

In the above sum,when m or k goes to plus or minus infinity, the product of interpolating polynomials will vanish, hence these terms do not enter. Along any boundary, the values of F, ∂F/∂m and ∂F/∂k depend only upon the end points of that edge segment, so that the same limit is obtained for the rectangles on both sides of a boundary; i.e. the function F and its derivatives are continuous.

If many interpolations on k will be made for a fixed value of m, then the formula can be broken into two steps. This is particularly useful in interpolation for the p-k method of flutter analysis, and in frequency response studies. In the first phase, find m_L and m_U such that $m_L < m < m_U$, and then compute for i = 1, J.

$$F(m, k_j) = \sum_{i=L,U} [F(m_i k_j) f_i(m) + \frac{\partial F}{\partial m}(m_i,k_j) g_i(m)] \quad (7)$$

and

$$\frac{\partial F}{\partial k}(m, k_j) = \sum_{i=L,U} \frac{\partial F}{\partial k}(m_i, k_j)\ f_i(m) \quad (8)$$

These 2J terms are stored for each matrix term to be interpolated. In the second phase, which is programmed within the inner loop, find k_L and k_U such that $k_L < k < k_U$, and compute

$$F(m,k) = \sum_{j=L,U} [F(m,k_j)f_j(k) + \frac{\partial F}{\partial k} (m,k_j)g_j(k)] \qquad (9)$$

5.7.3 The three phases of the module

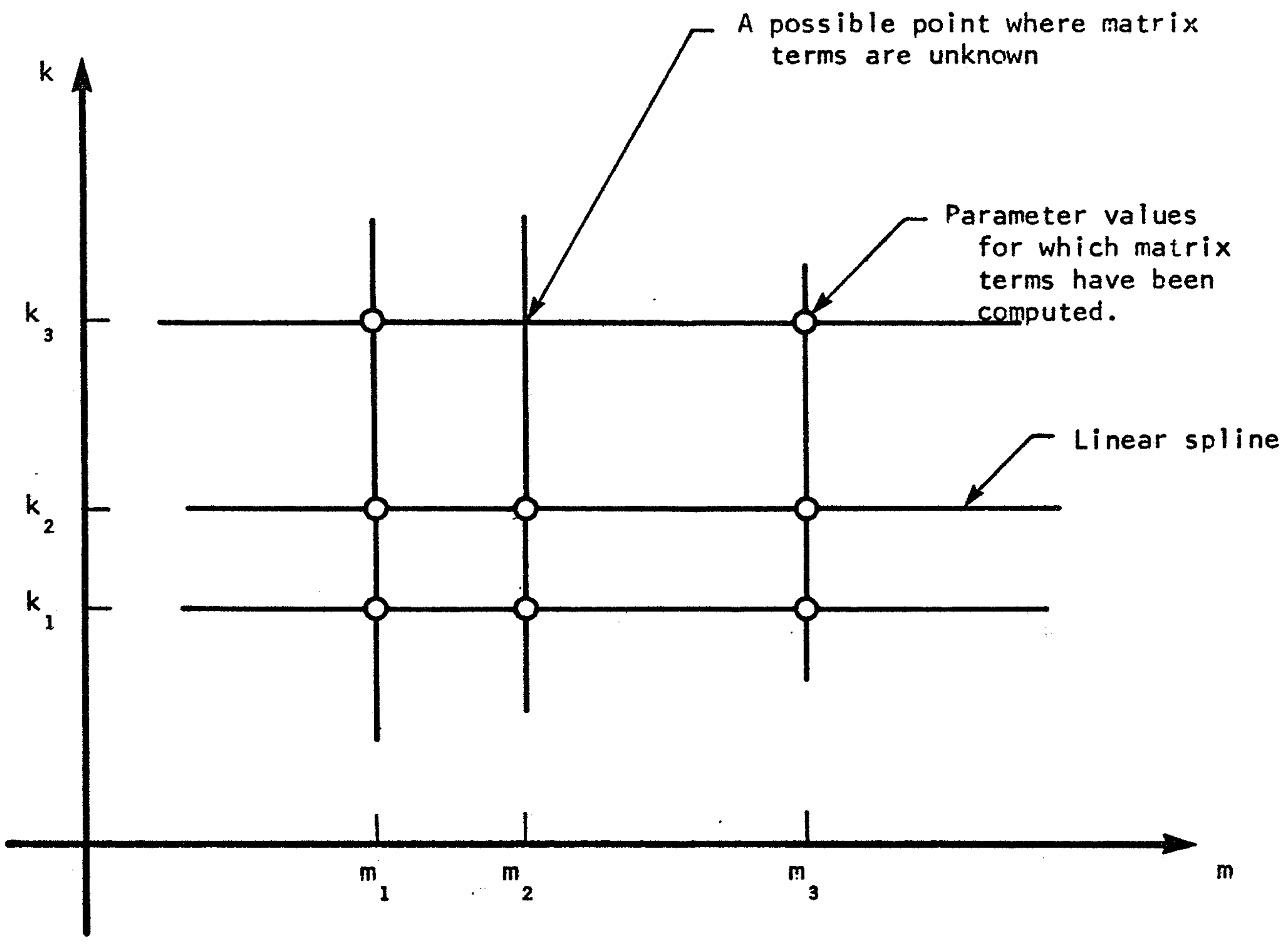
AMI1. The data from which the interpolation will be made has been output from the Aerodynamic Matrix Processor. It consists of a list of parameter pairs (m, k) plus the aerodynamic matrix (modal or physical coordinates) for each set of parameters. This list has been sorted such that all of the parametric values of a given matrix term are together, and the identically zero terms are eliminated. Thus the input to AMI1 contains

NPARAM	(the number if parameter pairs)
m_1, k_1 m_2, k_2 $\vdots$	a list of parameter pairs
i, j	indices of the first nonzero terms
$Q_{i,j}$	a list of values for the first non-zero matrix term
$\vdots$	
i, j	indices of the last non-zero terms
Q	a list of values for the last non-zero terms

The AMI1 module does the cross-splining in both directions and outputs a list. This list specifies first the number of m's and k's. The product of these numbers will be greater than the number of input pairs if there were imbedded blanks in the parameter pair set, (See Figure 5.7-1). Next is a list of the parameters m_i and k_j. Then a list for each non-zero matrix term of Q, $\partial Q/\partial m$, $\partial Q/\partial k$ for every parameter pair. Thus the output list of AMI1 contains the slopes needed for interpolation. Module AMI1 needs to be executed only once.

AMI2. This phase of the calculation uses equations (7) and (8) to reduce the number of Mach numbers to one, leaving only lists of Q and $\partial Q/\partial k$. An alternative is to interpolate with respect to the other parameter, which can be done by the formulas obtained by interchanging m and k in equations (7) and (8). The output list is similar to AMI1, except only one parameter is used.

AMI3. This phase of the calculation uses equation (9), and outputs a matrix which could be input to any desired module.



A possible point where matrix terms are unknown
Parameter values for which matrix terms have been computed.
Linear spline
k
k_3
k_2
k_1
m_1
m_2
m_3
m

Problem: Find F(m,k) for all m, k; if F(m,k) is known at the circled points.

Figure 5.7-1
Parametric Interpolation

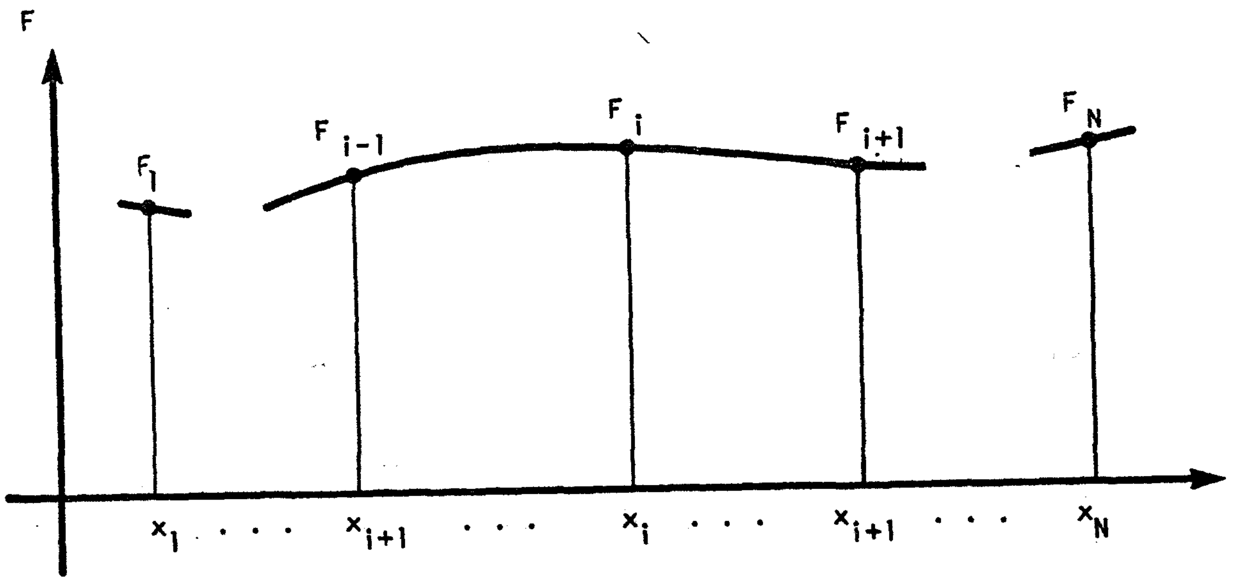

F
F_1
F_{i-1}
F_i
F_{i+1}
F_N
x_1 . . . x_{i+1} . . . x_i . . . x_{i+1} . . . x_N

Figure 5.7-2

One-dimensional Spline

Figure 5.7-3 An Algorithm for Solving the Three Moment Equations

INPUT	$x_1 < x_2 < \ldots < x_N$	(inequalities will be checked)
	$F_1, F_2, \ldots, F_N$	
OUTPUT	$M_1, M_2, \ldots, M_N$	$M = \frac{d^2f}{dx^2}$
	$\theta_1, \theta_2, \ldots, \theta_N$	$\theta = \frac{df}{dx}$
INTERMEDIATE RESULTS	$\delta_1, \delta_2, \ldots, \delta_N$	(δ can be stored in same storage as θ, and b can be stored in same storage as m)
	$b_1, b_2, \ldots, b_N$	

CALCULATIONS - see next page.

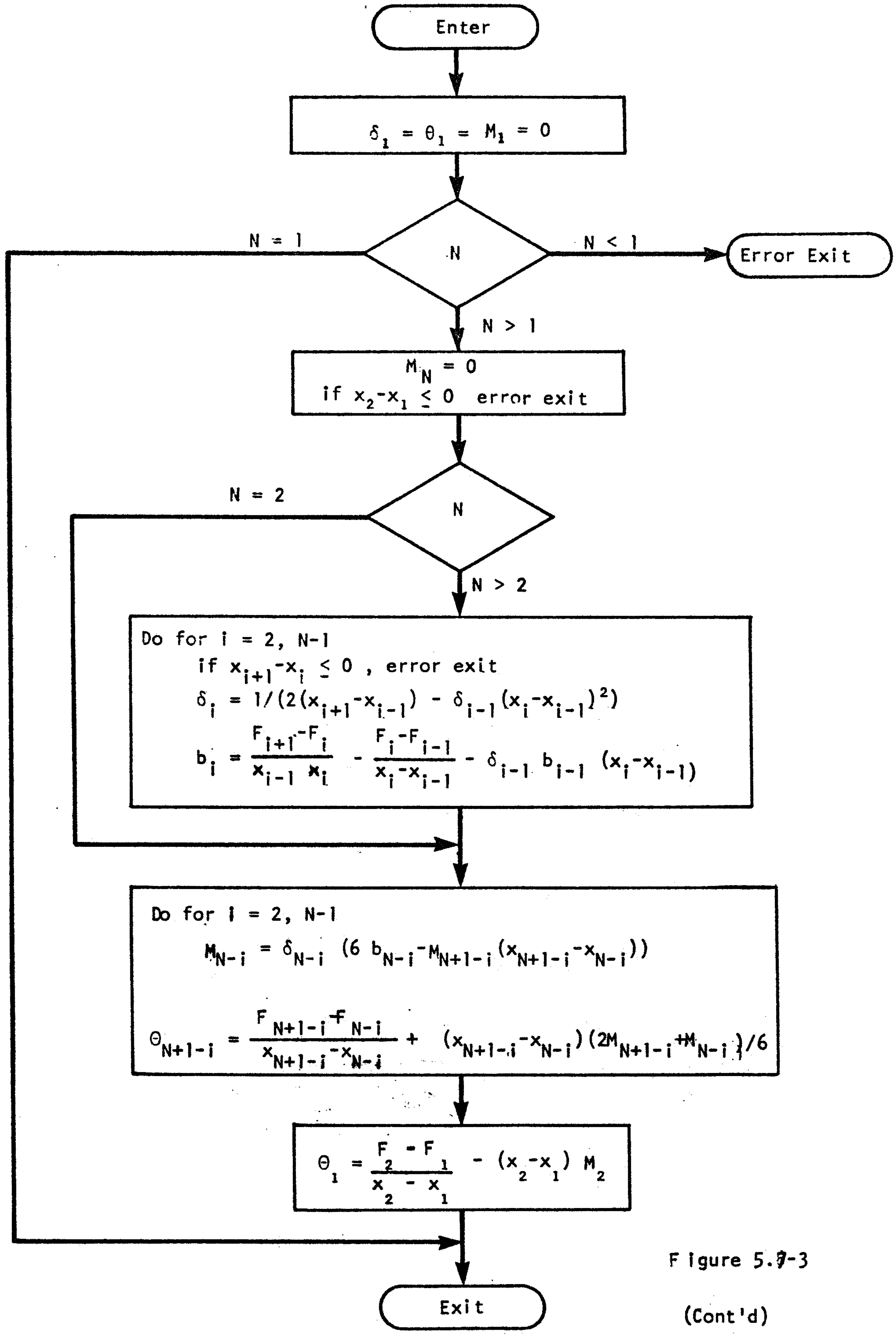

Enter
$\delta_1 = \theta_1 = M_1 = 0$
N
N = 1
N < 1
Error Exit
N > 1
$M_N = 0$
if $x_2 - x_1 \leq 0$ error exit
N
N = 2
N > 2
Do for i = 2, N-1
if $x_{i+1} - x_i \leq 0$, error exit
$\delta_i = 1/(2(x_{i+1} - x_{i-1}) - \delta_{i-1}(x_i - x_{i-1})^2)$
$b_i = \frac{F_{i+1} - F_i}{x_{i-1}\ x_i} - \frac{F_i - F_{i-1}}{x_i - x_{i-1}} - \delta_{i-1}\ b_{i-1}\ (x_i - x_{i-1})$
Do for i = 2, N-1
$M_{N-i} = \delta_{N-i}\ (6\ b_{N-i} - M_{N+1-i}(x_{N+1-i} - x_{N-i}))$
$\Theta_{N+1-i} = \frac{F_{N+1-i} - F_{N-i}}{x_{N+1-i} - x_{N-i}} + (x_{N+1-i} - x_{N-i})(2M_{N+1-i} + M_{N-i})/6$
$\Theta_1 = \frac{F_2 - F_1}{x_2 - x_1} - (x_2 - x_1)\ M_2$
Exit

Figure 5.7-3

(Cont'd)

5.8 Flutter Analysis

Flutter analysis is the solution for velocities, frequencies, density, etc., for which the system has a steady sinusoidal response with no excitation. The flutter point is on a boundary between stability and instability. One of the features of advanced unsteady aerodynamic theories is that the aerodynamic forces are not known except for sinusoidal motion, so there is a degree of uncertainty in the estimates of the amount of stability for parameters near the flutter boundary. There are several emthods of analysis, of which two will be implemented.

1. The k method.
2. The p-k method.

The matrix problem for sinusoidal equilibrium with no forces is

$$\left[-M\omega^2 + iB\omega + K - \frac{\rho V^2}{2} \quad Q\left(\frac{b\omega}{V}, \frac{V}{a}\right)\right]\{u\} = 0 \qquad (1)$$

The complex matrix Q has been generated by the Aerodynamic Matrix Processor. Typically, the speed of sound, "a", is a weak function of ρ (standard atmosphere). In NASTRAN, the structural damping factor, $(1+ig)$, is included automatically in K by the direct Dynamic Matrix Assembler, see Step (9) of Section 3. The problem is to find values for V, ω, and ρ for which a non-zero vector, $\{u\}$ exists. Thus the problem is similar to an eigenvalue problem, but it certainly is not in canonical form.

The k method treats the aerodynamic matrix as a mass term and results in

$$\left[\left(M + \frac{\rho b^2}{2k^2} \; Q(k,m)\right) p^2 + Bp + k\right] \{u\} = 0 \qquad (2)$$

where the eigenvalue

$$p = \alpha + i\omega = i\omega \left(1 - \frac{i\bar{g}}{2}\right) \qquad (3)$$

the Mach number $m = V/a$ and the reduced frequency $k = b\omega/V$. The p-k method treats the aerodynamic matrix as a stiffness and results in

$$\left[Mp^2 + Bp + \left(K - \rho \frac{m^2 a^2}{2} \; Q\,(k,m)\right)\right] \{u\} = 0 \qquad (4)$$

In both forms a value of k and m must be selected before an eigenvalue can be found. Thus the eigenvalue is a function $p(k,m,\rho)$. Solving equation (3)

$$\omega = \mathrm{Im}(p) \qquad (5)$$

$$\bar{g} = 2\,\frac{\mathrm{Re}(p)}{\mathrm{Im}(p)} \qquad (6)$$

Thus both ω and $\bar{g}$ become functions of the parameters (k,m,ρ). In order to be a valid solution

$$\omega = \frac{kma}{b} \qquad (7)$$

$$\bar{g} = g_0 \quad (\text{usually} = 0) \qquad (8)$$

In equations (7) and (8), the left sides are computed from equations (5) and (6), and the m and k on the right hand side of equation (7) are the values selected for computing the eigenvalue. Thus the solution of the flutter problem will require some iteration to find consistent solutions.

5.8.1 The k method.

The k method of analysis uses the aerodynamic mass formulation, equation (2). It is a very efficient method when the matrix Q does not depend upon Mach number m. Thus the eigenvalue can be found without a choice of m, and then equation (7) can be used to solve for m. Equation (8) is solved by plotting g (equation (6)) versus $V = b\omega/k$.

A flow chart is shown in Figure 5.8-1. If more than one Mach number is specified, then the analysis will be done for every m. The method is to compute eigenvalues for a list of values of m,k,ρ. The ρ value selections are placed in the inner loop since ρ is a simple multiplier of the aerodynamic mass matrix, thus saving time. The number of eigenvalues extracted is controlled by the user, and for each eigenvalue (p), the frequency (ω) and damping ($\overline{g}$) will be computed. A recommended method of eigenvalue extraction is described in Appendix H.

The module output consists of tables of eigenvalues, eigenvectors, frequencies, and damping factors, which can be selected by the user for output by the Output File Processor. Usually the user will request V-g and V-ω plots, where ρ and m are held fixed and k varies; lists in this format will be prepared by this module for the XY plotter.

5.8.2 The p-k method.

The p-k method (see Appendix F) is designed for use when iteration is required to solve the consistency equation (equation (7)). It uses the stiffness formulation (equation (4)). The procedure is, first, select values of m and ρ, then guess a value of k, and solve (approximately) for the eigenvalue, p. A new estimate for k is obtained by solving equation (7),

$$k_{new} = \frac{b\ \omega(k_{old}, m, \rho)}{m\ a} \tag{9}$$

The process is repeated until convergence occurs. A recommended method of eigenvalue extraction that converges simultaneously on k and p is described in Appendix I. A flow chart is shown in Figure 5.8-2. A list of ρ's and m's is supplied by the user, as well as the desired number of eigenvalues. The initial estimates of k are obtained from a previous solution for different values of m and ρ. The initial estimates will be automatically generated in a series of solutions (subcases) for different values of ρ and m by taking the converged values of k for the closest previous point in the ρ,m plane. The initial values for the first solution in the series will either be provided by the user or, in the case of a modal formulation, be evaluated from the structural vibration modes.

The module output for the p-k method consists of tables of eigenvalues, eigenvectors, frequencies and damping for converged (i.e., consistent with equation (7)) solutions. Usually, the user will request V-g and V-ω plots, where ρ is held fixed and m varies; lists in this format will be prepared for the XY plotter.

5.8.3 NASTRAN Implementation.

It is not desirable to incorporate all of the flutter calculations in a single module because of the technical NASTRAN requirement that checkpoints (i.e., points where tidy exits can be made from the program) can occur only after a module has been completed. It is shown in Section 5.7 that the Aerodynamic Matrix Interpolator can be separated into three phases. Phase 1 is preparatory to interpolation; phase 2 interpolates with respect to Mach number; phase 3 interpolates with respect to reduced frequency. This separation is useful for flutter analysis because the different phases can be placed in separate modules.

Figure 5.8-1 indicates the recommended sequence of module calls for the k method. The Aerodynamic Matrix Interpolator is separated into two modules. The calculation can be checkpointed after any eigenvalue extraction for a particular set of ρ, m, and k.

Figure 5.8-2 indicates the recommended sequence of module calls for the p-k method. The third stage of the Aerodynamic Matrix Interpolator is incorporated into the Flutter Analysis Module in order to minimize data transfers between high speed and low speed memories. Dummy modules are introduced in order to permit checkpoints between different values of m and ρ and also between different eigenvalues with the same values of m and ρ.

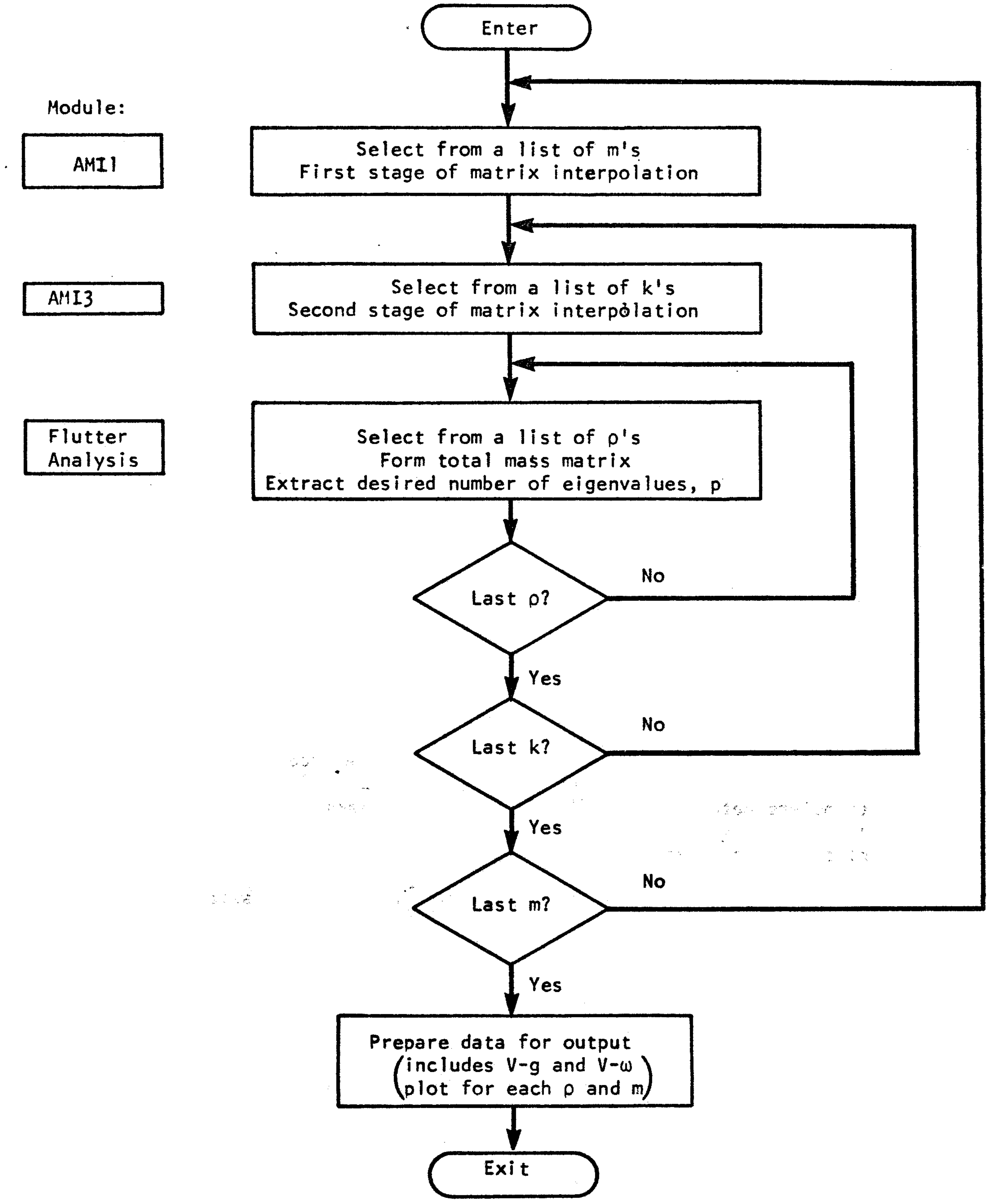

Enter
Module:
AMI1
Select from a list of m's
First stage of matrix interpolation
AMI3
Select from a list of k's
Second stage of matrix interpolation
Flutter Analysis
Select from a list of ρ's
Form total mass matrix
Extract desired number of eigenvalues, p
Last ρ?
No
Yes
Last k?
No
Yes
Last m?
No
Yes
Prepare data for output (includes V-g and V-ω plot for each ρ and m)
Exit

Figure 5.8-1

Flutter analysis by the k method

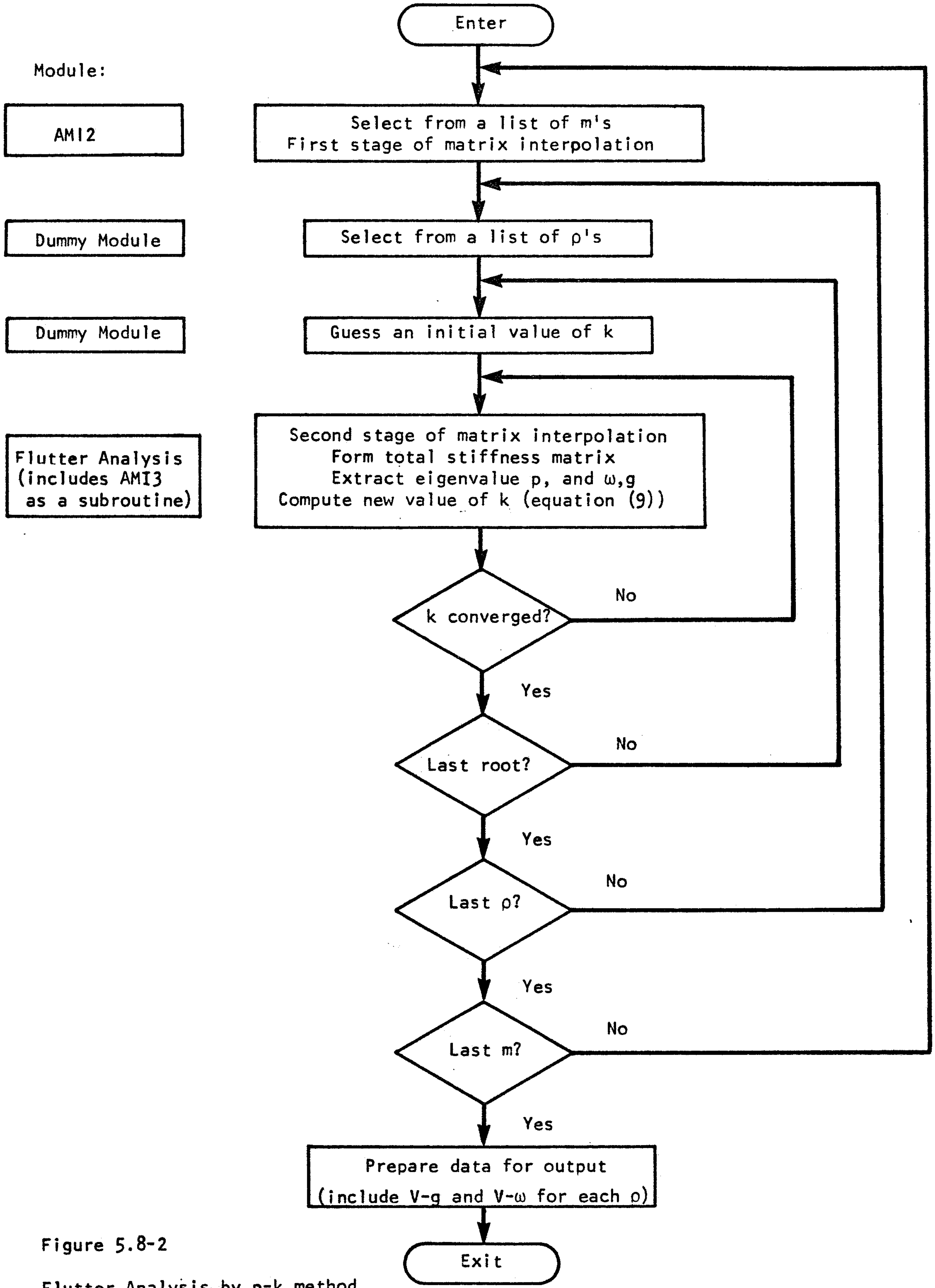

Enter
Module:
AMI2
Select from a list of m's
First stage of matrix interpolation
Dummy Module
Select from a list of ρ's
Dummy Module
Guess an initial value of k
Flutter Analysis (includes AMI3 as a subroutine)
Second stage of matrix interpolation
Form total stiffness matrix
Extract eigenvalue p, and ω,g
Compute new value of k (equation (9))
k converged?
No
Yes
Last root?
No
Yes
Last ρ?
No
Yes
Last m?
No
Yes
Prepare data for output
(include V-g and V-ω for each ρ)
Exit

Figure 5.8-2

Flutter Analysis by p-k method

5.9 Dynamic Aeroelastic Load Generator

The function of the Dynamic Aeroelastic Load Generator is to calculate load vectors that will be used in frequency response analysis. The load vectors will be calculated from two data sources: the standard NASTRAN dynamic load sets defined on RLØAD1, RLØAD2, TLØAD1 and TLØAD2 data cards; and the gust load sets defined on GUST cards, (see Figure 6-15). The desired combination of load sets will be indicated on a standard DLØAD card.

Four different rigid formats will use the results of the Dynamic Aeroelastic Load Generator. They are:

2A. Direct Aeroelastic Frequency and Random Response.

3A. Direct Aeroelastic Transient Response by Fourier Integral Method.

5A. Modal Aeroelastic Frequency and Random Response.

6A. Modal Aeroelastic Transient Response by Fourier Integral Method.

For Rigid Formats 3A and 6A, the time dependent load data must be converted to the frequency domain using the approach described in Appendix D. The user will supply a list of frequencies via standard FREQ, FREQ1, or FREQ2 data cards, at which response calculations will be made.

The data blocks produced by the module are the dynamic load vector for direct analysis $\{P_d\}$, or the dynamic load vector for modal analysis $\{P_h\}$, evaluated at the user-supplied list of frequencies. Formal algebraic expressions for these vectors are

a) Direct Analysis

$$\{P_d\} = \begin{Bmatrix} P_a^s \\ \hdashline P_e^s \end{Bmatrix} + \begin{Bmatrix} P_a^a \\ \hdashline 0 \end{Bmatrix} \qquad (1)$$

where $\{P_a^s\}$ and $\{P_e^s\}$ are dynamic structural loads and

$$\{P_a^a\} = q[G_{ka}]^T[S_{kj}][A_{jj}]^{-1} \{w_j^g\} \tag{2}$$

is the gust load vector.

b) Modal Analysis

$$\{P_h\} = \left\{\begin{array}{c} P_i^s \\ \hdashline P_e^s \end{array}\right\} + \left\{\begin{array}{c} P_i^a \\ \hdashline 0 \end{array}\right\} \tag{3}$$

where $\{P_i^s\}$ and $\{P_e^s\}$ are dynamic structural loads and

$$\{P_i^a\} = q[\phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1}\{w_j^g\} \tag{4}$$

Since the number of frequencies is presumed to be large and the number of degrees of freedom to which dynamic <u>structural</u> loads are applied may be small, the load vectors $\{P_a^s\}$, $\{P_e^s\}$, and $\{P_i^s\}$ will be calculated from the applied load vector $\{P_j^s\}$ (where the subscript j refers simply to the subset of all physical points, u_p, at which loads are applied) by the reduction procedure indicated in Section 11.1 of the NASTRAN Theoretical Manual. This procedure results in the expressions:

$$\{P_d^s\} = \left\{\begin{array}{c} P_a^s \\ \hdashline P_e^s \end{array}\right\} = [T_{dj}] \{P_j^s\} \tag{5}$$

and

$$\{P_h^s\} = \left\{\begin{array}{c} P_i^s \\ \hdashline P_e^s \end{array}\right\} = [T_{hj}] \{P_j^s\} \tag{6}$$

where $[T_{dj}]$ and $[T_{hj}] = [\phi_{dh}]^T[T_{dj}]$ are computed first.

If rigid formats 3A or 6A (transient response) are selected, the load vector $\{P_j^s\}$ will be a function of time, formed by combining individual load sets,

$$\{P_j^s\} = \sum_k S_{sk}\,\{P_j^k\} \tag{7}$$

where, if the data are obtained from a TLØAD1 card,

$$\{P_j^k\} = \{A_{jk}\}\,F_k(t-\tau_{jk}) \tag{8}$$

or if the data are obtained from a TLØAD2 card,

$$\{P_i^k\} = \{A_{jk}\}\,(\overline{t})^{n_k}\left(e^{\alpha_k t}\right)\ \cos\,(2\pi f_k\overline{t} + \phi_k) \qquad 0 < \overline{t} < T_{2k} - T_{1k} \tag{9}$$

where $\overline{t} = t - T_{1k} - \tau_{jk}$.

The formulas for converting equations (8) and (9) to the frequency domain are given in Appendix D. The value of n_k is restricted to be an integer, greater than or equal to zero, for aeroelastic response analysis.

The gust downwash velocity vector $\{w_j^g\}$ appearing in equations (2) and (4) is calculated from information appearing on GUST cards, from the velocity of the vehicle defined by the Mach number and the velocity of sound, and from the geometric description of the aerodynamic elements. More than one gust may be specified. For each gust, $\{w_j^g\}$ is the scalar product

$$w_j^g = \tilde{w}_j^g \cdot \tilde{n}_j \qquad (10)$$

where $\tilde{w}_j^g$ is the gust velocity vector at the aerodynamic element, and $\tilde{n}_j$ is the normal to the surface of the aerodynamic element. The gust velocity vector at the aerodynamic element is delayed in time by an amount τ_{kj} from the specification on the GUST card, i.e.,

$$\tilde{w}_j{}^g(\omega) = \tilde{w}^g \, e^{-i\omega\tau}{}_{jk} \qquad (11)$$

The time delay is computed from

$$T_{jk} = \frac{X_j - X_0}{V_p + V_f \cdot \hat{e}_f} \qquad (12)$$

where

V_p	is the velocity of propagation of the gust disturbance relative to the fluid.
V_f	is the velocity of the flow.
X_0	is the gust origin specified on GUST card.
X_j	is the X-coordinate of j^{th} aerodynamic downwash point in the gust coordinate system specified on the GUST card.
e_f	is the direction cosine of the aerodynamic X-axis on the gust X-axis.

If $\tilde{w}^g$ is specified as a function of time, a transformation to the frequency domain will be performed in the manner indicated earlier for dynamic structural loads. If more than one gust is specified, the vectors $\{w_j{}^g\}$ are combined using the load combination factors on the DLØAD card.

The gust velocities are reduced to equivalent aerodynamic load vectors by means of equation (2) or equation (4). Two cases are distinguished, corresponding to whether the elements of $[A_{jj}]^{-1}$ are provided or the elements of the triangular decomposition factors $[L_{jj}]$ and $[U_{jj}]$ of $[A_{jj}]$ are provided. In either case, the elements are stored for specific values of Mach number, m, and reduced frequency, $k = \frac{b\omega}{am}$, and an interpolation is made to the desired Mach number and the desired list of reduced frequencies, $k_n = \frac{b\omega_n}{V}$, where $V = am$. The aerodynamic load vector may be expressed for direct analysis, as

$$\{P_a{}^a\} = q[Q_{aj}]\{w_j{}^g\} \tag{13}$$

or, for modal analysis as

$$\{P_i{}^a\} = q[Q_{ij}]\{w_j{}^g\} \tag{14}$$

where

$$q = 1/2\ \rho V^2 \text{ , dynamic pressure}$$

$$[Q_{aj}] = [G_{ka}]^T[S_{kj}][A_{jj}]^{-1} \tag{15}$$

$$[A_{ij}] = [\phi_{ai}]^T[G_{ka}]^T[S_{kj}][A_{jj}]^{-1} \tag{16}$$

The interpolated values of $[Q_{aj}]$ and $[Q_{ij}]$ will be generated by the Aerodynamic Matrix Interpolator, Section 5.7. The Dynamic Aeroelastic Load Generator performs the calculations indicated in equation (13), or in equation (14), and combines the structural and aerodynamic loads by equation (1) or equation (3). A flow diagram for the calculations performed by the module is shown in Figure 5.9-1.

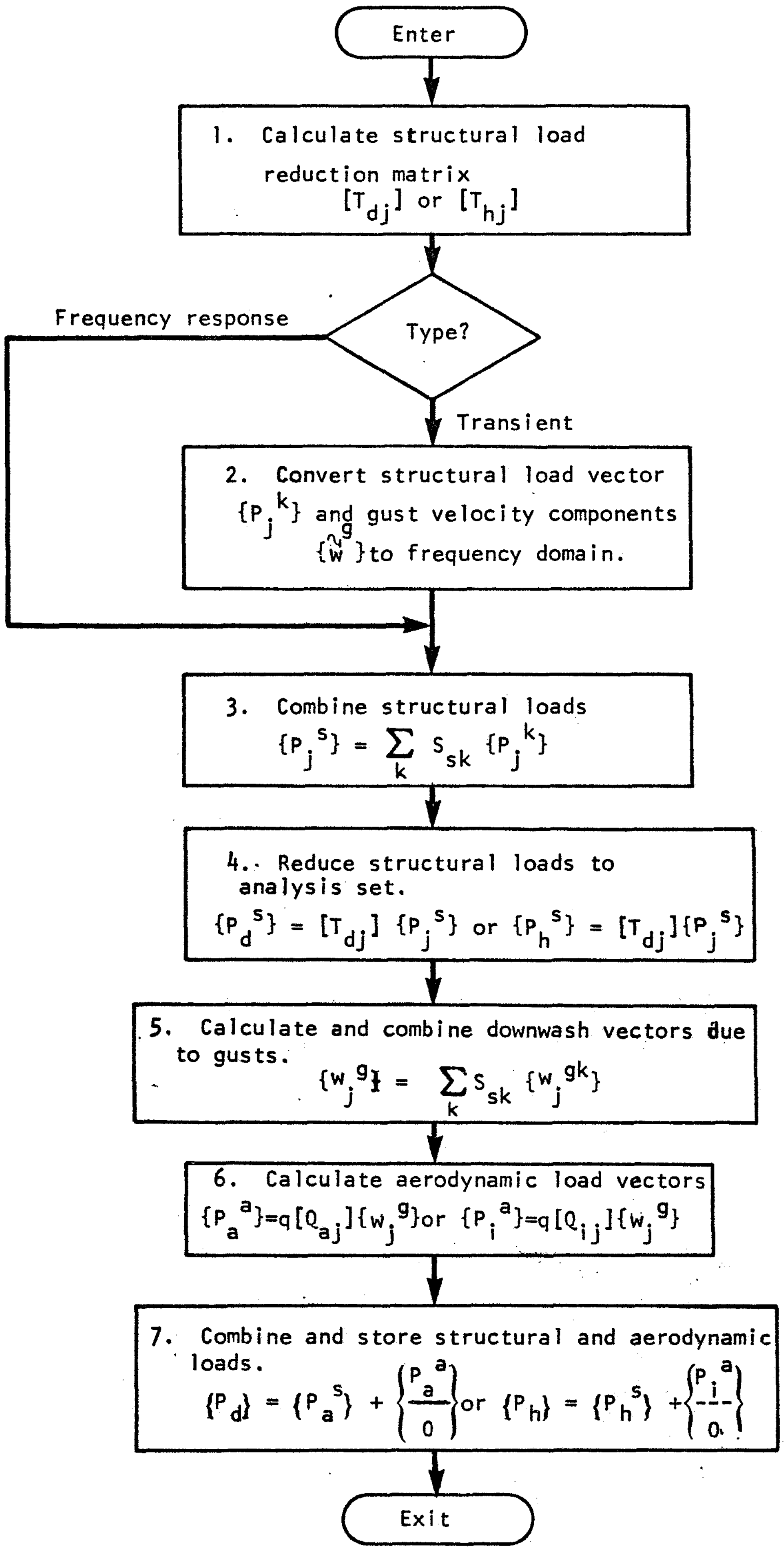

Enter
1. Calculate structural load reduction matrix $[T_{dj}]$ or $[T_{hj}]$
Type?
Frequency response
Transient
2. Convert structural load vector $\{P_j^k\}$ and gust velocity components $\{\tilde{w}^g\}$ to frequency domain.
3. Combine structural loads $\{P_j^s\} = \sum_k S_{sk} \{P_j^k\}$
4. Reduce structural loads to analysis set. $\{P_d^s\} = [T_{dj}] \{P_j^s\}$ or $\{P_h^s\} = [T_{dj}]\{P_j^s\}$
5. Calculate and combine downwash vectors due to gusts. $\{w_j^g\} = \sum_k S_{sk} \{w_j^{gk}\}$
6. Calculate aerodynamic load vectors $\{P_a^a\}=q[Q_{aj}]\{w_j^g\}$ or $\{P_i^a\}=q[Q_{ij}]\{w_j^g\}$
7. Combine and store structural and aerodynamic loads. $\{P_d\} = \{P_a^s\} + \left\{\frac{P_a^a}{0}\right\}$ or $\{P_h\} = \{P_h^s\} + \left\{\frac{P_i^a}{0}\right\}$
Exit

Figure 5.9-1 Flow Diagram for Dynamic Aeroelastic Load Generator

5.10 Aerodynamic Data Recovery Module

The Aerodynamic Data Recovery Module calculates those items of aerodynamic data pertaining to aerodynamic elements which the user has requested for output. The formulas for the different classes of aerodynamic data produced in dynamic analysis are:

Displacements at aerodynamic control points:

$$\{u_k\} = [G_{ka}]\ \{u_a\} \tag{1}$$

Aerodynamic motion variables:

$$\{w_j\} = [D_{jk}]\{u_k\} + [D_{je}]\{u_e\} + \{w_j^g\} \tag{2}$$

Aerodynamic force variables:

$$\{f_i^a\} = q[A_{jj}]^{-1}\{w_j\} \tag{3}$$

Forces at aerodynamic control points:

$$\{F_k^a\} = [S_{kj}]\{f_j^a\} \tag{4}$$

The calculations are performed for each solution produced by the Frequency Response Module or the Flutter Analysis Module.

5.11 Inverse Fourier Transform Module

The function of this module is to calculate the time history of a vector of transient response quantities $\{u(t)\}$ from the vector of frequency response quantities $\{u(\omega_n)\}$, $n = 0, 1, 2 \ldots$, produced by the Data Recovery Modules. The response quantities may include motions, stresses, internal forces and aerodynamic variables. The calculation is an approximation of the inverse Fourier transform

$$\{u(t)\} = \frac{1}{\pi} \int_0^\infty R\ell[e^{i\omega t}\{u(\omega)\}]d\omega \tag{1}$$

where $R\ell[\ ]$ indicates the real part of $[\ ]$. The user specifies the times, t_m, at which responses will be calculated via a standard TSTEP card.

The method that will be used is the method described in Appendix D for frequencies that are not uniformily spaced. It consists of fitting the frequency response data with a cubic spline and then performing the integration in equation (1) by exact quadratures.

The steps of the calculation are as follows:

1. Sort each frequency response quantity $u_j(\omega_n)$ into a list increasing with n.

2. Perform a cubic spline fit to each $u_j(\omega_n)$ by the method described in Section 5.7 for functions of one variable. The result of the calculation will be the parameters a_{nj}, b_{nj}, c_{nj} and d_{nj} in the representation of the function.

$$u_j(\omega) = a_{nj} + b_{nj}\,\overline{\omega} + c_{nj}\,\overline{\omega}^2 + d_{nj}\,\overline{\omega}^3 \tag{2}$$

$$\omega_n < \omega < \omega_{n+1}$$

where $\overline{\omega} = \omega - \omega_n$.

If the lowest frequency in the list is not zero, it will be designated ω_1 and the values of the parameters for $\omega_0 = 0$ will be calculated as follows:

a) If the quantity is a displacement acceleration, internal force, or stress:

$$u_j(0) = a_{0j} = a_{1j} - \frac{b_{1j}\omega_1}{2} \tag{3}$$

$$b_{0j} = 0$$

$$c_{0j} = b_{1j}/2\omega_1$$

$$d_{0j} = 0$$

b) If the quantity is a velocity

$$u_j(0) = a_{0j} = a_{1j} - b_{1j}\omega_1 \tag{4}$$

$$b_{0j} = b_{1j}$$

$$c_{0j} = d_{0j} = 0$$

The indicated extrapolation is designed to cope with the awkward situation that arises when the structure is a free body with singular stiffness matrix. Under these conditions it is not possible to specify zero as a frequency, since infinite displacements will result. Nevertheless, accelerations, internal forces, stresses and <u>relative</u> displacements will tend to constant values at zero frequency, and <u>relative</u> velocities will tend to zero.

The values of a_{nj}, b_{nj}, c_{nj} and d_{nj} are formed into lists (i.e., column vectors $\{a_{nj}\}$, $\{b_{nj}\}$, $\{c_{nj}\}$ and $\{d_{nj}\}$) increasing with $n \geq 0$.

3. The exact integration formula is given by equation (28) of Appendix D. It is implemented by the following operations.

a) Form and store the coefficient matrices $[A_{mn}]$, $[B_{mn}]$, $[C_{mn}]$ and $[D_{mn}]$ where the elements are

$$A_{mn} = \left[-i t_m^3 \left(e^{i\omega_{n+1} t_m} - e^{i\omega_n t_m} \right) \right] \qquad (5)$$

$$B_{mn} = \left[t_m^2 - i\Delta\omega_n t_m^3 \; e^{i\omega_{n+1} t_m} - t_m^2 \; e^{i\omega_n t_m} \right] \qquad (6)$$

$$C_{mn} = \left[(2it_m + 2\Delta\omega_n t_m^2 + i\Delta\omega_n^2 t_m^3) e^{i\omega_{n+1} t_m} - 2it_m e^{i\omega_n t_m} \right] \qquad (7)$$

$$D_{mn} = \left[(-6 + 6i\Delta\omega_n t_m + 3\Delta\omega_n^2 t_m^2 - i\Delta\omega_n^3 t_m^3) e^{i\omega_{n+1} t_m} + 6e^{i\omega_n t_m} \right] \qquad (8)$$

and $\Delta\omega_n = \omega_{n+1} - \omega_n$; $\omega_n = \omega_0, \omega_1, \omega_2, \ldots$; $t_m = t_1, t_2, \ldots.$ m is the row index and n is the column index.

Since a very large number if factors $e^{i\omega_n t_m} = \cos\omega_n t_m + i \sin \omega_n t_m$ are required, and since the required accuracy is not high, it is suggested that a special interpolation scheme be devised for evaluating the factors.

b) The time history of the jth response quantity is given by

$$u_j(t_m) = \frac{1}{\pi t_m^4} \; R\ell\left([A_{mn}]\{a_{nj}\} + [B_{mn}]\{b_{nj}\} + [C_{mn}]\{c_{nj}\} + [D_{mn}]\{d_{nj}\} \right) \qquad (9)$$

$$m \geq 1$$

c) For $t_0 = 0$ use, instead of equation (9),

$$u_j(0) = \frac{1}{\pi} R\ell\left(\lfloor A_{0n} \rfloor \{a_{nj}\} + \lfloor B_{0n} \rfloor \{b_{nj}\} + \lfloor C_{0n} \rfloor \{c_{nj}\} + \lfloor D_{0n} \rfloor \{d_{nj}\} \right) \qquad (10)$$

where elements in the row vectors are

$$\left.\begin{aligned} A_{0n} &= \Delta\omega_n \\ B_{0n} &= (1/2)\Delta\omega_n^{\,2} \\ C_{0n} &= (1/3)\Delta\omega_n^{\,3} \\ D_{0n} &= (1/4)\Delta\omega_n^{\,4} \end{aligned}\right\} \qquad (11)$$

5.12 Aerostatic Pool Distributor

The functions of the Aerostatic Pool Distributor are similar to those of the Aerodynamic Pool Distributor, Section 5.1. The details have not been formulated.

5.13 Aerostatic Matrix Generator

The Aerostatic Matrix Generator performs the same functions for static analysis that the Aerodynamic Matrix Generator performs for dynamic analysis. Specifically, it generates the following arrays for user-specified values of Mach number.

$$[D_{jk}],\ [S_{kj}],\ [A_{jj}] \text{ or } [A_{jj}]^{-1},\ \{w_j^{\,g}\} \text{ and } [D_{je}]$$

These arrays are defined in equations (2), (3) and (4) of Appendix B. The differences in their definitions between static and dynamic analysis are that $\{w_j^{\,g}\}$ refers to the static angle of attack distribution rather than to gust velocities, and that $[D_{je}]$, which relates angle of attack (and perhaps other aerodynamic variables) to vehicle velocity components and control surface rotations, is automatically generated in static analysis.

All of the arrays listed above are dependent on the particular aerodynamic theory being used. No work with specific static aerodynamic theories has been done as part of the present study.

5.14 Aerostatic Matrix Assembler

The operations performed in the Aerostatic Matrix Assembler are matrix additions, multiplications, and partitions that will use the standard NASTRAN matrix subroutines. The operations are defined in Step 9 of Section 4.

5.15 Divergence Analysis Module

The operation performed in the Divergence Analysis Module is the calculation of the eigenvalues and eigenvectors of

$$[K_{dd}^{1} + K_{dd}^{2} + \lambda K_{dd}^{a}] \{U_d\} = 0$$

where $[K_{dd}^{1} + K_{dd}^{2}]$ and $[K_{dd}^{a}]$ are generated in the Aerostatic Matrix Assembler, (see Step 9 of Section 4).

The algorithm that is used to obtain eigenvalues and eigenvectors is described in Appendix C. As suggested in Appendix C, the user will specify a set of shift points, λ_{01}, λ_{02}, λ_{03}, etc., and the program will find the eigenvalue closest to each shift point. The origin, $\lambda = 0$, will often be used as a shift point, and usually only one shift point will be specified. The output of the module includes the last trial vector obtained from each shift point, and all of the successive estimates of λ.

5.16 Aerostatic Load Generator

The Aerostatic Load Generator combines loads from three sources:

a) Structural loads generated in the static portion of NASTRAN, SSG1, and SSG2.

b) Aerodynamic loads

c) Loads on extra points, specified directly by the user.

Formulas for combination of the loads are shown in Section 4, Step 11, for both trimmed and untrimmed cases.

5.17 Static Aeroelastic Response Module

The Static Aeroelastic Response Module solves the matrix equation

$$[K_{dd}]\ \{u_d\} = \{P_d\} \tag{1}$$

by one of two methods selected by the user. $[K_{dd}]$ is a real, unsymmetric matrix. For the direct solution option, $[K_{dd}]$ is decomposed into its triangular factors and $\left\{u_d\right\}$ is obtained by forward and backward substitution. In the iterative solution option $[K_{dd}]$ is separated into aerodynamic and non-aerodynamic terms

$$[K_{dd}] = [K_{dd}^{a}] + [K_{dd}^{1} + K_{dd}^{2}] \tag{2}$$

and the problem is stated in the form

$$[K_{dd}^{1} + K_{dd}^{2}]\ \{u_d^{n}\} = \{P_d\} - [K_{dd}^{a}]\ \{u_d^{n-1}\} \tag{3}$$

where $\{u_d^n\}$ is the nth iterative approximation to $\{u_d\}$. The advantage of the iterative solution option is that the matrix which is decomposed, $[K_{dd}^1 + K_{dd}^2]$, is sparce and narrowly banded in comparison with $[K_{dd}^a]$. Substantial time-saving results for large problems if the number of iterations is small. The details of the proposed solution method, including convergence criteria, is explained in Section B.4 of Appendix B.

6. USER INTERFACE

6.1 Input Preparation and Options.

The addition of aerodynamics to the NASTRAN structural analysis program will follow the guidelines that presently exist in NASTRAN. The input will be introduced via control and data cards. The Executive Control Deck will provide the basic program to be selected. The Case Control Deck will choose options of data selection, output control and methods. The Bulk Data Deck provides all numerical data. In order to describe the input preparation and options, the new Bulk Data cards are shown in Section 6.2. Examination of these cards is the easiest way to describe the tasks required of the user for preparing input and choosing options.

There are many options in NASTRAN including:

1. Rigid Format, Alter, DMAP. The algorithm used to solve a particular aeroelastic problem may be one of the nine rigid formats in Table 2-4. These include choices of statics vs dynamics, and of direct vs modal approaches. These basic solution options can be modified with the NASTRAN ALTER feature for minor changes. The new aerodynamic modules can also be used with the DMAP (Direct Matrix Abstraction Program) feature to provide capability not provided in rigid formats.

2. Restart, Checkpoint. Any of the rigid formats can be restarted. This is useful for checking step by step during the solution, changing to a new rigid format, requesting additional output and for many other purposes. Restart may be very valuable in aeroelastic analysis, since aero-

dynamic matrices computed for flutter analysis can easily be used for other purposes such as a transient response by the Fourier transform technique.

3. Method. Each of the rigid formats has a choice of methods for some of the modules. For example, several methods of eigenvalue extraction are provided and two methods of flutter analysis are provided.

4. Aerodynamic Matrix. Several different aerodynamic theories will be provided automatically, in addition to the ability to incorporate aerodynamic matrices generated exterior to the program.

5. Matrix Interpolation. When aerodynamic matrices are required at many different reduced frequencies (k) and Mach numbers (m), automatic matrix interpolation between matrices for chosen values of k and m is provided. The user may ALTER the rigid format to force the matrix to be recomputed rather than interpolated when this is more desirable (see Section 5.7).

6. Geometric Interpolation. The aerodynamic coordinates and the structural coordinates are not (in general) at the same locations. An interpolation matrix is computed using either surface (plate-like) or linear (beam-like) splines (see Section 5.4).

7. Aerodynamic Elements. Several different forms of aerodynamic element cards (defining quadrilateral and cylindrical elements) are available (see Section 5.2).

8. Output. A large variety of automatic output is provided. The aerodynamic displacements and forces are available for both output tables and plots (see Section 6.3 for a list).

6.2 New Data Cards for Dynamic Aeroelasticity

Samples of the new data card descriptions are shown in Figures 6-1 thru 6-15. Most of these cards have been referenced in Section 5, where their use in the modules is discussed. They have been collected for presentation together.

The general guidelines followed in the choice and format of the cards are:

a. All input is (or should be) user-oriented.

b. There should be little redundant information, and if it exists, consistency checks should be made.

c. The user should be forced to make decisions best left to human judgement. For example, no default method of choosing aerodynamic cells is provided.

d. When a new data card performs a task similar to an existing card, the formats should be similar. For example, the format of the new GRIDK card is nearly identical to the format of the the existing GRID card.

The choice of recommended data cards for the description of aerodynamics has been made in a parallel fashion to the NASTRAN structural elements. Table 6-1 shows the equivalence.

Aerodynamic	Structural
CAERØi	CQUADi
PAERØi	PQUADi
AERØ, MKAERØ	MATi

Table 6-1

Bulk Data Cards

A structural CQUADi (or CTRIAi, etc.) connection card describes the geometry of an element by referring to grid points. The CAERØi bulk data card provides aerodynamic cell geometry by referring to grid points, or to coordinate locations and lengths. The (i) indicates that there are several alternative cards. The connection card references a property PQUADi card which gives data about the element such as its thickness. The PAERØi card gives the parameters of the i^{th} aerodynamic theory, such as center of pressure location, lift curve slope, initial angle of attack, etc. The PAERØi card refers to an aerodynamic AERØ (or MKAERØ) card which provides reference values of air velocity, density, and length.

The interpolation between structural and aerodynamic degrees of freedom is provided with SPLINEi bulk data cards. (There are three alternate forms.) These cards allow the user a large range of choice from complete user control to almost complete NASTRAN control of the setup of splines. Two alternate forms of SETi data cards are used in connection with the splines for selection of structural points for interpolation. GRIDK cards will be useful when interpolation is desired, but no aero-cells have been defined since they provide an alternative method to define k-points. This would be useful for direct matrix input methods, interpolation for output, etc. The GUST data card gives a method of introducing aerodynamic loads due to atmospheric turbulence or to acoustic waves, which can be added to the structural loads provided by existing NASTRAN cards.

6.3 Output.

At present NASTRAN provides a large selection of structural output information. Additional output capability will be required for aerodynamic information. There are three types of output: tables, drawings, (structural plots) and curves (XY plots). The aerodynamic quantities which may be output are:

1. u_k displacements at interpolated aerodynamic control points
2. w_j^a displacements of aerodynamic degrees of freedom
3. f_j^a aerodynamic forces
4. F_k^a forces on interpolated aerodynamic control points.

The preparation of these data require additional calculation (see Aerodynamic Data Recovery, Section 5.10), since the NASTRAN solution set will be structural points in the modal or analysis sets.

Output requests are made in case control. They control the operation of the Output File Processor, XY Plotter, and Deformed Structure Plotter. The structural quantities and the aerodynamic quantities listed above are output only when requested in case control. The sets of quantities which may be requested will include:

DISPLACEMENT SPCFØRCE ØLØAD STRESS	existing (structural)
KDISP JDISP KFØRCE JFØRCE	new (aerodynamic)

The quantities desired will be in sets defined by the user, in the manner presently used for structural quantities.

In addition, there will be some automatic output. A flutter analysis summary will include complex values of eigenvalues, frequencies, and damping factors, tabulated versus Mach number, density, and reduced frequency. The real and imaginary parts of flutter eigenvectors may be displayed on structure plots, as well as being tabulated.

Input Data Card CAERØ1

Description: Defines a quadrilateral macro aerodynamic element in terms of four grid points.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CAERØ1	EID	PID	G1	G2	G3	G4	NSPAN	NCHØRD	ABC
CAERØ1	13	21	12	14	24	22	3	LIST	ABC

+BC	lists	of div	ision p	oints f	or uneq	ual sub	divisio	n.	
+BC	.13	.29	.66	END					

Field	Contents
EID	Macro element identification number (integer > 0)
PID	Identification of a PAERØi property card (integer > 0)
G1, G2, G3, G4	Grid points in order going around element (integer> 0)
NSPAN, NCHØRD	The number of equally spaced elements in the spanwise and chordwise directions within the macro element (integer > 0) or "LIST".

Remarks:

1. The aero elements will be identified as A.B.C where "A" is the macro EID, and "B", and "C" are the span and chord index.

2. The word "LIST" in field 8 or 9, implies that the percent span and/or percent chord subdivisions have been supplied on continuation cards. The word "END" stops each list. When two lists are given, the span list comes first. Several continuations may be used.

3. A triangular element is formed if G4 = G1 or G3 = G2

4. The element should be (nearly) flat.

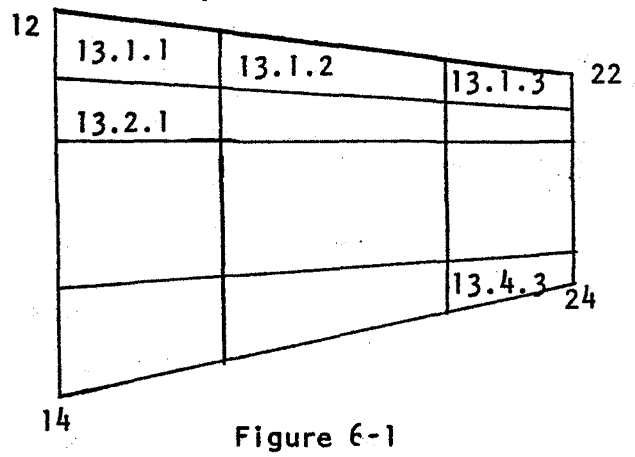


Figure 6-1

CAERØ data card

5. The positive direction for element displacements is determined from the order of connection and the right hand rule.

Figure 6-1

(CONT'D)

Input Data Card CAERØ2

Description: Defines a quadrilateral macro aerodynamic element in terms of four points.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CAERØ2	EID	PID					NSPAN	NCHØRD	ABC
	13	21					3	LIST	ABC

+BC	CID1	X1	Y1	Z1	CID2	X2	Y2	Z2	DEF
+BC		13.4	5.0	0.0	2	6.8	9.5	3.6	DEF

+EF	CID3	X3	Y3	Z3	CID4	X4	Y4	Z4	GHI
+EF		27.0	5.0	0.0	2	15.2	9.5	3.6	GHI

+HI	lists of division for unequal subdivision								JKL
+HI	.13	.29	.66	END					

Field	Contents
EID	Macro element identification number (integer > 0)
PID	Identification of a PAERØi property card (integer > 0)
NSPAN, NCHØRD	The number of equally spaced elements in the spanwise and chordwise directions within the macro element (integer > 0) or "LIST".
CID, X, Y, Z	Coordinate system identification number (integer > 0) and the coordinates of a corner point (real).

Remarks:

1. The aero elements will be identified as A.B.C where "A" is the macro EID, and "B" and "C" are the span and chord index.

2. The word "LIST" in field 8 or 9, implies that the percent span and/or percent chord subdivisions have been supplied on continuation cards. The word "END" stops each list. When two lists are given, the span list comes first. Several continuations may be used.

3. A triangular element is formed if corner 2 = corner 1 or if corner 3 = corner 4.

Figure 6-2
CAERØ Data Card

4. The element should be (nearly) flat.

5. The positive direction for element displacements is determined from the order of connection and the right hand rule.

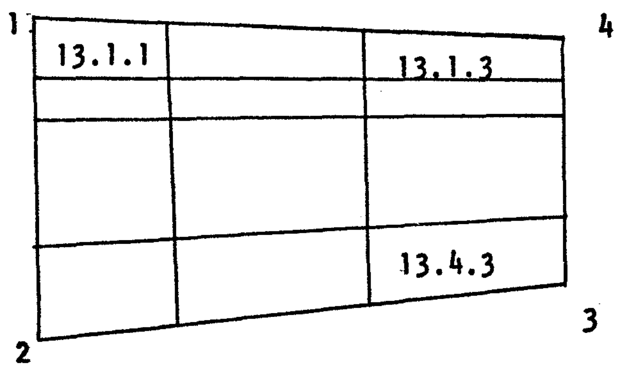


Figure 6-2

(Cont'd)

Input Data Card CAERØ3

Description: Defines a trapezoid macro aerodynamic element in terms of two leading edge points and edge chord lengths

Format and Example:

1	2	3	4	5	6	7	8	9	10
CAERØ3	EID	PID			X_{12}	X_{43}	NSPAN	NCHØRD	ABC
CAERØ3	13	21			13.6	8.4	3	LIST	ABC

+BC	CID1	X1	Y1	Z1	CID4	X4	Y4	Z4	DEF
+BC		13.4	5.0	0.0	2	6.8	9.5	3.6	DEF

+EF	lists of division points for unequal subdivision								
+EF	.13	.29	.66	END					

Field	Contents
EID	Macro element identification number (integer > 0)
PID	Identification of a PAERØi property card (integer > 0)
X_{12}, X_{43}	length of edges, stream direction (integer)
NSPAN, NCHØRD	The number of equally spaced elements in the spanwise and chordwise directions within the macro element (integer > 0) or "LIST".
CID, X, Y, Z	Coordinate system identification number (integer > 0) and the coordinates of a leading edge element (real).

Remarks:

1. The aero elements will be identified as A.B.C where "A" is the macro EID, and "B", and "C" are the span and chord index.

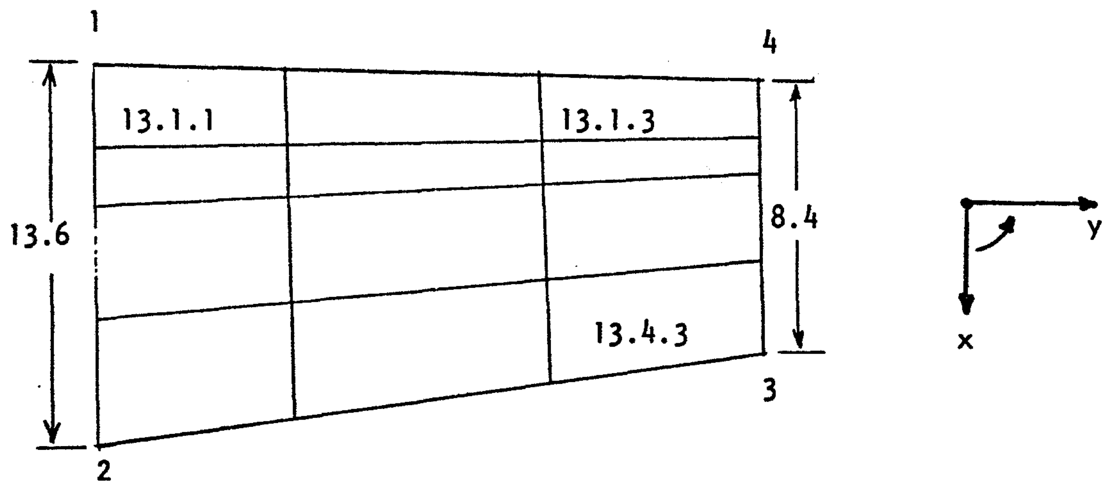
2. The word "LIST" in field 8 or 9, implies that the percent span and/or percent chord subdivisions have been supplied on continuation cards. The word "END" stops each list. When two lists are given, the span list comes first. Several continuations may be used.

3. A triangular element is formed if X_{12} or $X_{43} = 0$

Figure 6.3

CAERØ3 Data Card

4. The positive direction for element displacements is determined from the order of connection and the right hand rule. If point 4 is



farther outboard than point 1, the positive z axis will out of the paper in the following sketch.

5. The program constructs a macro element in which sides 12 and 43 are parallel to the flow.

Figure 6-3

(Cont'd)

Input Data Card CAERØ6

Description: Defines a cylindrical aerodynamic macro element in terms of a forward point and a length in the airstream direction.

Format and Example:

1	2	3	4	5	6	7	8	9	10
CAERØ6	EID	PID	CID	X1	Y1	Z1	ΔX	NCELLS	abc
	1	19	0	1.7	3.5	6.2	9.5	LIST	ABC

+bc	lis	ts of di	vision	points	for une	qual sub	divisi	on	
+BC	.16	.35	.70	END					

Field	Contents
EID	Macro-element identification number (integer > 0)
PID	Identification of a PAERØi property card (integer > 0)
CID	Coordinate system identification number for locating forward point (integer $\geq$ 0)
X1, Y1, Z1	Location of forward point in coordinate system CID, (real)
ΔX	Length of cylindrical axis, in direction of air flow (See AERØ card for direction) (real $\neq$ 0)
NCELLS	Number of equally spaced element subdivisions within the macro, integer > 0 or "LIST".

Remarks:

1. The cylindrical elements will be identified as A.B where "A" is the macro EID, and B is the element.

2. The word "LIST" in field 9 implies that the unequal division points have been supplied on a continuation card. The word END stops the list.

Figure 6-4

CAERØ6 Data Card

Input Data Card PAERØi

Description: Gives properties for aerodynamic theories.

Format and Example:

1	2	3	4	5	6	7	8	9	10
PAERØi	PID	TID	. .	. l	ist of	parame	ers .	. .	
PAERØ3	6	9	.71	.68					

Field	Contents
PID	Property Identification number (referenced by CAERØi cards).
TID	Theory identification number (aero cells with different TID's are uncoupled)
. . . parameters . . .	A list of parameters whose format depends upon the theory.

Remarks:

1. Fields 4 thru 9 and continuation cards may be used to input parameters to the theory, such as lift curve slope, center of pressure, etc. This sample shows the required format for all theories.

Figure 6-5

PAERØi Data Card

Input Data Card PAERØ1

Description: Gives properties for DOUBLET LATTICE method.

Format and Example:

1	2	3	4	5	6	7	8	9	10
PAERØ1	PID	TID	XØ	X1	α_0				
PAERØ1	1	9	.45	.95					

Field	Contents
PID	Property identification number (referenced by CAERØ)
TID	Theory identification (aero-cells with different TID's are uncoupled)
XØ	Center of pressure in fraction of box chord, (real) Default XØ = 0.25
X1	Downwash center in fraction of box chord, (real) Default X1 = 0.75
α_0	Initial angle of attack (real).

Remarks:

1. If symmetry is desired, see AERØ data card.

Figure 6-6

PAERØ1 Data Card

Input Data Card AERØ

Description: Gives basic aerodynamic parameters

Format and Example:

1	2	3	4	5	6	7	8	9	10
AERØ	CID	ASØUND	BREF	RHØZERØ	K	M	SYMXZ	SYMXY	
AERØ	3	1.3+4	100.	1.-5					

Field	Contents
CID	A rectangular coordinate system, flow is in the positive X1 direction (integer $\geq$ 0 or blank)
ASØUND	speed of sound
BREF	Reference length (for reduced frequency)
RHØZERØ	Reference density
K	Reduced frequency
M	Mach number
SYMXZ	Symmetry key for aero coordinate X-Z plane (word SYM, ANTI, or blank)
SYMXY	Symmetry key for aero coordinate X-Y plane (word SYM, ANTI, or blank) used to simulate ground effects (SYM) and in hydroelasticity, a free surface (ANTI).

Remarks:

1. This card is required for aerodynamic problems.

2. Fields 6 and 7 specify the value of reduced frequency and Mach number for which aerodynamic matrices are computed. If a series of values is desired, fields 6 and 7 are left blank and a MKAERØ bulk data card is used.

3. If field 2 is blank, the basic coordinate system is assigned.

Figure 6-7

AERØ Data Card

BULK DATA CARD

Input Data Card SPLINE1

Description: Defines a surface spline for interpolating out-of-plane motion.

Format and Example:

1	2	3	4	5	6	7	8	9	10
SPLINE1	EID	CAERØ	CID	AXES	SETG	⊠	D	⊠	abc
SPLINE1	3	4		FIXED	14		1.0		ABC

+bc	k_z	$k_{\theta x}$	$k_{\theta y}$	⊠	⊠	⊠	⊠	⊠	
+BC									

Field	Contents
EID	Element identification number (integer> 0)
CAERØ	Aero element which defines plane of spline
CID	Rectangular coordinate system which defines plane of spline
AXES	To choose the spline coordinate system : "FIXED" will cause the axes of the selected aero element or coordinate system to be chosen; "PRINC" will cause the principal axes of the g-set to be chosen.
SETG	refers to a SETi card which lists the structural "g"-set to which the spline is attached.
D	the plate rigidity $0 < D \leq$ "INF" (default = 1.0)
k_z	linear attachment spring $0 < k_z \leq$ "INF" (default = infinite, "INF")
$k_{\theta x}$, $k_{\theta y}$	torsional attachment spring $0 \leq k \leq$ "INF" (default = 0)

Figure 6-8

SPLINE1 Data Card

Remarks:

1. Either CAERØ or CID field must be blank.

2. The interpolated points (k-set) will be defined by aerocells or GRIDK data cards.

3. Continuation is not required if default springs are used.

Figure 6-8

(Cont'd)

Input Data Card SPLINE2

Description: Defines a beam spline for interpolating out-of-plane motion.

Format and Example:

1	2	3	4	5	6	7	8	9	10
SPLINE2	EID	CAERØ	CID	AXES	SETG	╳	EI	GJ	abc
SPLINE2	5	8		PRINC					

+bc	k_z	$k_{\theta x}$	$k_{\theta y}$	╳	╳	╳	╳	╳	

Field	Contents
EID	Element identification number (integer > 0)
CAERØ	Aero element which defines plane of spline
CID	Rectangular coordinate system which defines plane of spline.
AXES	To choose the spline coordinate system "FIXED" will cause the axes of the selected aero element or coordinate system to be chosen; "PRINC" will cause the principal axes of the g-set to be chosen.
SETG	Refers to a SETi card which lists the structural "g"-set to which the spline is attached.
EI, GJ	Beam bending and torsional rigidity $0 <$ EI, GJ $\leq$ "INF" (default EI = GJ = 1.0)
k_z	linear attachment spring $0 < k_z \leq$ "INF" (default = infinity, "INF").
k_x, k_y	torsional attachment spring $0 \leq k \leq$ "INF" (default = 0)

Remarks:

1. Either CAERØ or CID field must be blank.

Figure 6-9

SPLINE2 Data Card

2. The interpolated points (k-set) will be defined by aero cells or GRIDK data cards.

3. Continuation is not required is default springs are used.

Figure 6-9

(Cont'd)

Input Data Card SPLINE3

Description: Defines a beam spline for interpolating inplane motion

Format and Example:

1	2	3	4	5	6	7	8	9	10
SPLINE3	EID	CAERØ	CID	AXES	SETG	╳	EI	AE	abc
SPLINE3	9		3	FIXED	13				

+bc	k_x	k_y	$k_{\theta z}$	╳	╳	╳	╳	╳	

Field	Contents
EID	Element identification number (integer > 0)
CAERØ	Aero element which defines the plane of the spline.
CID	Rectangular coordinate system which defines the plane of spline.
AXES	To choose the spline coordinate system "FIXED" will cause the axes of the selected aero element or coordinate system to be chosen; "PRINC" will cause the principal axes of the g-set to be chosen.
SETG	Refers to a SETi card which lists the structural "g"-set to which the spline is attached.
EI, AE	beam bending and extensional rigidity $0 <$ EI,AE $\leq$ "INF" (default = "INF")
k_x, k_y	linear attachment springs $0 < k \leq$ "INF" (default = 1.0)
$k_{\theta z}$	torsional attachment spring $0 \leq k \leq$ "INF" (default = 0)

Figure 6-10

SPLINE3 Data Card

Remarks:

1. Either CAERØ or CID field must be blank.

2. The interpolated (k-set) points will be defined by aero-cells or GRIDK data cards.

3. The default values of rigidity and springs make the spline equivalent to a rigid plate.

4. Continuation not required if default springs are used.

Figure 6-10

(Cont'd)

Input Data Card SET1

Description: Defines a set of structural grid points by a list.

Format and Example:

1	2	3	4	5	6	7	8	9	10
SET1	SID	G1	G2	G3	G4	G5	G6	G7	ABC
SET1	3	31	62	93	124	16	17	18	ABC

+BC	G8								
+BC	19								

Field	Contents
SID	set of identification numbers (integer > 0)
G1 . . .	list of structural grid points

Remarks:

1. These cards are referenced by the SPLINE data card.

Figure 6-11

SET1 Data Card

Input Data Card SET2

Description: Defines a set of structural grid points in terms of aerodynamic macro elements.

Format and Example:

1	2	3	4	5	6	7	8	9	10
SET2	SID	MACRØ	S1	C1	S2	C2	H1	H2	
SET2	3	17	1	1	2	4		3.51	

Field	Contents
SID	set identification number (integer > 0)
MACRØ	element identification number of an aero macro element
S1, C1	span and chord identification number of the first element
S2, C2	span and chord identification number of the last element
H1, H2	greatest height above (using right hand rule with the order the corners as listed on a CAERØ card) to include in set, and the greatest distance below. (floating $\geq$ 0)

Remarks:

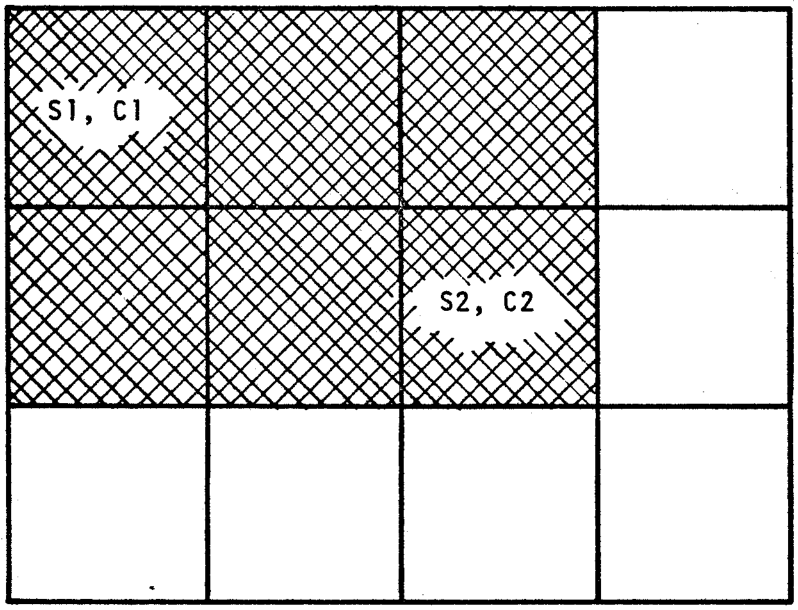
1. These cards are referenced by the SPLINE data cards.

2. The default values for H1, H2, are infinity.

3. Every grid point, within a prism whose cross-section includes the set of aero elements of which S1, C1 is the first and S2, C2 is the last, and within the height range, will be in the set. For example,

Figure 6-12

SET2 Data Card



The shaded area in the figure defines the cross-section of the prism

Figure 6-12

SET2 Data Card

Input Data Card GRIDK

Description: Defines a non-structural grid point, for use in interpolation of structural displacements.

Format and Example:

1	2	3	4	5	6	7	8	9	10
GRIDK	ID	CP	X1	X2	X3	SPLINE	PS		

<u>Field</u>	<u>Contents</u>
ID	Grid K point identification number (integer> 0)
CP	Number of coordinate system in which the location of the grid K point is defined (integer $\geq$ 0)
X1, X2, X3	Location of point in coordinate system CP (real)
SPLINE	Identification number of a spline to which the GRIDK point will be associated, (integer > 0) (default, the spline whose cg is closest).
PS	Permanent single point constraints in spline coordinate system (any of the digits 1-6 with no imbedded blanks).

Remarks:

1. Identification numbers of GRIDK points must be unique, and not that of any GRID, SPØINT, or EPØINT.

2. The coordinate system will be that of the spline (See SPLINEi data card).

3. No structural elements, loads, etc., may be applied to a GRIDK point.

4. The GRIDK points will be included in the set of u_k points used in aeroelastic analysis. They may also be used to interpolate deflection in non-aeroelastic problems.

Figure 6-13

GRIDK Data Card

BULK DATA CARD

Input Data Card MKAERØ

Description: Provides a list of Mach numbers (m) and reduced frequencies (k) for aerodynamic matrix calculation.

Format and Example:

1	2	3	4	5	6	7	8	9	10
MKAERØ	. . .	lists	of valu	es . .					abc
MKAERØ	.10	.30	.50	END	.7	.9	1.0	1.2	ABC

+bc	. . .	contin	uation	of list	s . . .				
+BC	END								

Fields	Contents
lists	List of Mach numbers (real) followed by "END", then a list of reduced frequencies followed by "END".

Remarks:

1. This card will cause the aerodynamic matrices to be computed for a two dimensional array of parametric values.
2. Several MKAERØ cards may be in the deck.
3. If only one m and k are desired, the AERØ card can be used.

Figure 6-14

MKAERØ Data Card

Input Data Card: GUST Aerodynamic Gust Load Description

Description: Defines components of gust velocity for use in aeroelastic analysis.

Format and Example:

1	2	3	4	5	6	7	8	9	10
GUST	SID	CID	TYPE	GX	GY	GZ	XO	V_p	
GUST	5	3	T2	7	4	3	-100.		

Field	Contents
SID	Gust set identification number (integer > 0)
CID	Identification of coordinate system for specifying components of gust velocity (integer > 0)
TYPE	Flag to identify the information in fields 5, 6 and 7
If TYPE = R1	GX, GY, GZ are identification numbers for tables where the gust velocity component = $C(f) + iD(f)$, a function of frequency, f.
If TYPE = R2	GX, GY, GZ are identification numbers for tables where the gust velocity component = $B(f)e^{i\phi(f)}$, a function of frequency, f.
If TYPE = T1	GX, GY, GZ are identification numbers for tables where the gust velocity component = G(t), a function of time.
If TYPE = T2	GX, GY, GZ are identification numbers for TFUNCT cards where the gust velocity component = $A\bar{t}^{B} e^{Ct} \cos(2\pi ft + \phi)$ for $T_1 < t < T_2$ where $\bar{t} = t-T_1$, and t = time.
GX, GY, GZ	Identification numbers for tables or data cards which evaluate the components of gust velocity parallel to the axes of the coordinate system identified in field 3. If blank, that component of gust is zero. (integers > 0 or blank).

Figure 6-15

GUST Data Card

XO	Point on the x axis of the referenced coordinate system where the gust components are evaluated for time $\geq$ 0. Default = 0.0 (real).
V_p	Velocity of propagation of gust disturbance relative to fluid in the positive x direction.

Remarks:

1. Gusts may be combined with either gusts and/or other dynamic loads by means of a DLØAD card.

2. The downwash velocity at an aerodynamic element is the projection of the gust vector on the normal to the surface of the element.

3. If $V_p = 0$, the arrival of the gust at the several aerodynamic elements is delayed in time (or in phase for frequency response) by the time required for the vehicle to travel from point XO to the aerodynamic element. If $V_p \neq 0$, V_p is added vectorally to the velocity of the vehicle to calculate the time delay.

Figure 6-15

(Cont'd)

APPENDIX A

APPENDIX A

NASTRAN Matrix Terminology

Many of the operations performed in computerized structural analysis are conveniently expressed in the notation of matrix algebra. In NASTRAN matrix arrays are represented by a root symbol that indicates the type of physical quantity and by one or more subscripts and superscripts that act as modifiers. The root symbols currently used in static and dynamic analysis with NASTRAN are listed in Table 1. Square brackets, [ ], indicate two-dimensional arrays and twisted brackets, { }, indicate column vectors. Row vectors, which are less common, are indicated either by appending the transpose symbol, T, to the twisted brackets, or by ⌊ ⌋.

Subscripts are used primarily to designate the subsets of displacement components to which the root symbol applies as for example in the equation,

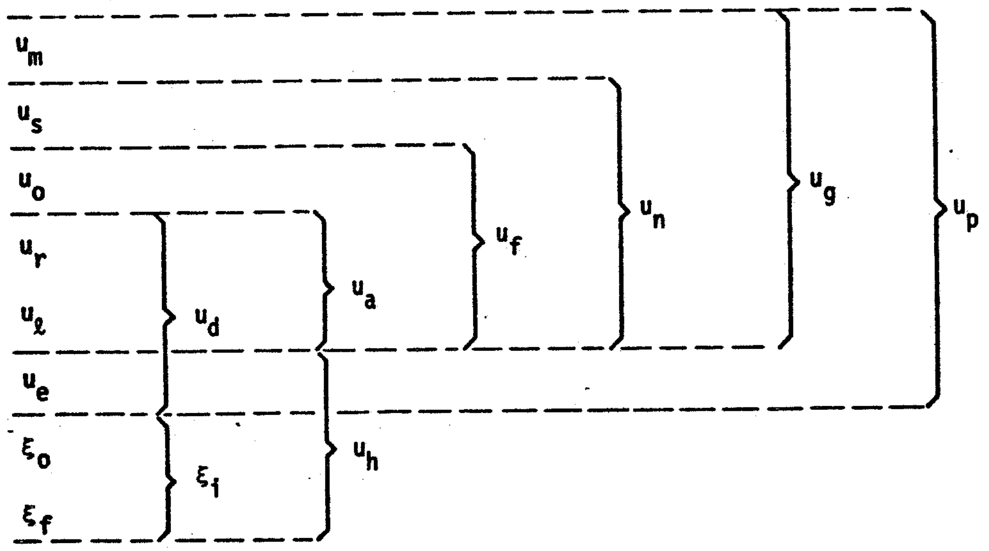
$$\{q_s\} = -\{P_s\} + [K_{fs}]^T\{u_f\} + [K_{ss}]\{u_s\}, \qquad (1)$$

which is used to recover single point forces of constraint, $\{q_s\}$, from displacements at constrained points, $\{u_s\}$, and at unconstrained (free) points, $\{u_f\}$. Nearly all of the matrix operations used in NASTRAN are concerned with partitioning, merging and transforming matrix arrays from one subset of displacement components to another. All the components of displacement of a given type (such as all points constrained by single-point constraints) form a vector set that is distinguished by a subscript from other sets. A given component of displacement can belong to several vector sets. The mutually exclusive vector sets, the sum of whose members are the set of all physical components of displacement, $\{u_p\}$, are listed in Table 2a.

In addition, a number of vector sets are defined as the union of two or more independent sets. See Table 2b.

In dynamic analysis additional vector sets are obtained by a modal transformation derived from real eigenvalue analysis of the set $\{u_a\}$. See Table 2c.

The nesting of the vector sets in Table 2 is depicted by the following diagram:



The gridpoint set $\{u_s\}$ contains all components of motion at structural gridpoints. The application of constraints and partitioning to the stiffness matrix involves, essentially, the elimination of $\{u_m\}$, $\{u_s\}$, $\{u_o\}$ and $\{u_r\}$ from $\{u_g\}$ to form a stiffness matrix referred to $\{u_\ell\}$, which is the set used for equation solution in static analysis.

Load vectors are distinguished by the same notation as displacement vectors. Rectangular matrices are, whenever necessary to clarify the meaning of the symbol, distinguished by double subscripts referring to the vector sets associated with the rows and columns of the array. Occasionally only the subscript associated with the row is used. Superscripts have no tensorial character and are used to identify arrays of different type or origin that refer to the same sets such as in the equation,

$$[M_{dd}] = [M^1_{dd} + M^2_{dd}] \ , \tag{2}$$

where $[M^1_{dd}]$ is the structural mass matrix and $[M^2_{dd}]$ is the direct input mass matrix.

The introduction of aeroelasticity into NASTRAN will require a few additional root symbols and a few additional subscripts and superscripts. The new root symbols are listed in Table 3. Use of the new subscripts is illustrated in Table 4. The subscripts j and k do not define new structural degrees of freedom and the quantities that they modify will not be processed by existing functional modules. They define instead what may be called aerodynamic degrees of freedom. The superscripts, both old and new, that will be used in aeroelasticity are illustrated in Table 5.

Table 1. Matrix Root Symbols Currently Used in NASTRAN.

$[B]$	damping matrix
$[b]$	modal damping matrix
$[D]$	rigid body transformation matrix
$[G]$	transformation matrix, as in $\{u_m\} = [G_m]\{u_n\}$
$[K]$	stiffness matrix
$[k]$	modal stiffness matrix
$[L]$	lower triangular factor
$[M]$	mass matrix
$[m]$	modal mass matrix
$\{N\}$	nonlinear force vector
$\{P\}$	vector of applied load components
$\{q\}$	vector of forces of reaction
$[R]$	matrix of constraint coefficients, as in $[R]\{u\} = 0$
$[T]$	load transformation matrix
$[U]$	upper triangular factor
$\{u\}$	vector of displacement components
$[X]$	rigid body stiffness matrix
$\{Y\}$	vector of enforced displacements
$\{\xi\}$	generalized coordinate
$\{\phi\}$	eigenvector
$[\phi]$	matrix of eigenvectors

Table 2. Displacement Vector Sets Currently Used in NASTRAN

Table 2a. Mutually Independent Vector Sets

u_m	coordinates eliminated as independent degrees of freedom by multi-point constraints
u_s	coordinates eliminated by single point constraints
u_o	coordinates omitted by structural matrix partitioning
u_r	coordinates to which determinate reactions are applied in static analysis
u_ℓ	the remaining structural coordinates used in static analysis (points left over)
u_e	extra degrees of freedom introduced in dynamic analysis to describe control systems, etc.

Table 2b. Combined Vector Sets

$u_a = u_r + u_\ell$, the set used in real eigenvalue analysis

$u_d = u_a + u_e$, the set used in dynamic analysis by the direct method

$u_f = u_a + u_o$, unconstrained (free) structural coordinates

$u_n = u_f + u_s$, all structural coordinates not constrained by multi-point constraints

$u_g = u_n + u_m$, all structural (grid) points including scalar points

$u_p = u_g + u_e$, all physical coordinates

Note: (+) sign indicates the union of sets

Table 2c. Modal Coordinate Sets

ξ_o	rigid body (zero frequency) modal coordinates
ξ_f	finite frequency modal coordinates

$\xi_i = \xi_o + \xi_f$, the set of all modal coordinates.

$u_h = \xi_i + u_e$, the set used in dynamic analysis by the modal method.

Note: (+) sign indicates the union of sets.

Table 3. New Root Symbols for Aeroelasticity

[D]	(new special meaning) matrix that gives transformation from displacements to downwashes
[S]	area coefficients
[A]	matrix that gives downwash vector as a function of pressures
[Q]	aerodynamic stiffness or transformation matrix divided by dynamic pressure
{w}	downwashes (or other similar aerodynamic variables)
{f}	vector of pressure coefficients (or other similar quantities)
{F}	vector of forces (distinguished from {P} which is a vector of externally applied forces)

Table 4. New Displacement Vector Sets for Aeroelasticity

$\{u_k\}$	displacements at aerodynamic control points
$\{w_j\}$	downwash coefficients for aerodynamic elements

Table 5. New and Old Superscripts for Aeroelasticity

$(\ )^1$	derived from structure as in $[K^1_{dd}]$ or $[M^1_{dd}]$
$(\ )^2$	direct user input as in $[K^2_{dd}]$ or $[M^2_{dd}]$
$(\ )^4$	structural damping derived as a property of structural elements as in $i[K^4_{dd}]$
$(\ )^a$	<u>aerodynamic</u> origin, as in $[M^a_{dd}]$
$(\ )^s$	<u>structural</u> origin, as in $\{P^s_a\}$
$(\ )^g$	externally generated aerodynamic quantity, as in $\{w^g_j\}$, the downwash velocity vector due to <u>gusts</u>
$(\ )^o$	<u>other</u> than aerodynamic origin, such as $\{u^o_e\}$
$(\ )^t$	refers to variables in $\{u_e\}$ that are used to adjust <u>trim</u> in static analysis, as in $\{u^t_e\}$
$(\ )^u$	refers to all variables in $\{u_e\}$ excluding $\{u^t_e\}$

APPENDIX B

APPENDIX B

Procedures for Static Aeroelastic Analysis

A flow diagram for static aeroelastic analysis with NASTRAN is shown in Table 2-2 of Section 2, and the formal matrix algebra is displayed in Section 4 of the report. The purpose of this appendix is to develop the theory for the formal procedures. The reader is referred to Appendix A for an explanation of the notation that will be employed.

The static part of NASTRAN is used to generate a structural stiffness matrix $[K_{\ell\ell}]$, referred to the displacement set $\{u_\ell\}$. It is also used to generate vectors of static loads $\{P^s_\ell\}$ and $\{P^s_r\}$ which may include, for example, gravity loads, pressure loads on structural panels, and loads due to thermal expansion. Gravity loads corresponding to different load factors can, incidentally, be specified by changing a single data entry. In addition, a number of data blocks ($[D]$, $[m_r]$, and $[M_{\ell\ell}D + M_{\ell r}]$) are generated which are used to treat inertia relief effects.

The displacement set $\{u_\ell\}$ is a subset of the dynamic analysis set $\{u_a\}$, which excludes the degrees of freedom $\{u_r\}$ that are restrained in order to provide a determinate set of reactions for free bodies. Thus,

$$\{u_a\} = \begin{Bmatrix} u_\ell \\ \hline u_r \end{Bmatrix} \tag{1}$$

The vectors $\{P^s_\ell\}$ and $\{P^s_r\}$ are, respectively, the static structural loads applied to $\{u_\ell\}$ and $\{u_r\}$.

B.1 Generation of Aerodynamic Matrices

The matrices generated by the Aerostatic Matrix Generator (see Section 5.13) are denoted by symbols identical to those used in dynamic analysis (see Section 5.5). Thus,

$$\{F_k^a\} = [S_{kj}]\{f_j^a\} \tag{2}$$

$$\{f_j^a\} = q[A_{jj}]^{-1}\{w_j\} \tag{3}$$

and

$$\{w_j\} = [D_{jk}]\{u_k\} + [D_{je}]\{u_e^a\} + \{w_j^g\} \tag{4}$$

where

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

$\{F_k^a\}$ is the vector of aerodynamic forces at aerodynamic control points.

$\{f_j^a\}$ is a vector of pressure coefficients, one or more for each aerodynamic element.

$\{w_j\}$ is a vector of aerodynamic degrees of freedom (e.g., angles of attack).

$\{u_e^a\}$ are NASTRAN "extra points" used to describe aerodynamic variables. They may be used to represent perturbation velocity components of the complete vehicle and to represent control surface deflections.

$\{w_j^g\}$ represents the static aerodynamic excitation. It includes, primarily, the static angle of attack distribution. It may also include, for example, terms to generate skin friction drag.

$\{u_k\}$ is a vector of structural displacements (deformations) at aerodynamic control points.

The matrices, $[S_{kj}]$, $[A_{jj}]$, $[D_{jk}]$ and $[D_{je}]$, are matrices of real constants, which may be functions of Mach number or other parameters. Each aerodynamic theory provides separate procedures for calculating them. $[A_{jj}]$ must be nonsingular.

As an example to illustrate the use of aerodynamic extra points, let the elements of $\{u_e^a\}$ be the components of a perturbation velocity vector

$$\{u_e^a\} = \left\{\begin{array}{c} V_x \\ \hline V_y \\ \hline V_z \end{array}\right\} \tag{5}$$

The magnitude of the total velocity is, for small perturbations,

$$V = V_o + a_1 V_x + a_2 V_y + a_3 V_z \tag{6}$$

where V_o is the magnitude of the steady velocity vector and a_1, a_2 and a_3 are its direction cosines in the coordinate system of V_x, V_y and V_z. In the aerodynamic pressure distribution given by Eq. (3), the dynamic pressure is

$$q = \frac{1}{2} \rho V^2 \simeq q_o + \rho V_o (a_1 V_x + a_2 V_y + a_3 V_z) \tag{7}$$

The sum of the static angle of attack distribution and the angle of attack distribution due to the perturbation velocities is

$$\{w_j\} = \frac{1}{V_o} \{n_{j1}V_x + n_{j2}V_y + n_{j3}V_z\} + \{w_j^g\} \tag{8}$$

where n_{j1}, n_{j2} and n_{j3} are the direction cosines of the normal to the jth surface in the coordinate system of V_x, V_y and V_z.

Substituting Eqs. (7) and (8) into Eq. (3):

$$\{f_j^a\} = \left(\frac{q_o}{V_o} + \rho(a_1V_x + a_2V_y + a_3V_z)\right)[A_{jj}]^{-1}\{n_{j1}V_x + n_{j2}V_y + n_{j3}V_z + V_o w_j^g\} \tag{9}$$

If the second order terms due to the products of perturbation velocities are neglected, Eq. (9) can be written as

$$\{f_j^a\} = \rho V_o [A_{jj}]^{-1} \left[\lceil w_j^g \rfloor [a] + \frac{1}{2} [n]\right] \begin{Bmatrix} V_x \\ \hline V_y \\ \hline V_z \end{Bmatrix} \tag{10}$$

where $\lceil w_j^g \rfloor$ is a diagonal matrix of static angles of attack,

$$[a] = \begin{bmatrix} a_1 & a_2 & a_3 \\ a_1 & a_2 & a_3 \\ a_1 & a_2 & a_3 \\ -- & \text{etc.} & -- \end{bmatrix} \tag{11}$$

$$[n] = \begin{bmatrix} n_{11} & n_{12} & n_{13} \\ n_{21} & n_{22} & n_{23} \\ --- & --- & --- \\ n_{j1} & n_{j2} & n_{j3} \\ --- & --- & --- \end{bmatrix} \tag{12}$$

Thus, by reference to Eqs. (3), (4) and (5)

$$[D_{je}] = \frac{2}{V_o}\left[\lceil w_j^g \rfloor [a] + [n]\right] \tag{13}$$

The first term in this equation is due to the influence of the perturbation velocities on the dynamic pressure. It is frequently neglected in aeroelastic analysis.

The aerodynamic forces applied to structural degrees of freedom can be written in the form

$$\begin{Bmatrix} F_\ell^a \\ \hline F_r^a \\ \hline F_e^a \end{Bmatrix} = - \left[\begin{array}{c|c|c} K_{\ell\ell}^a & K_{\ell r}^a & K_{\ell e}^a \\ \hline K_{r\ell}^a & K_{rr}^a & K_{re}^a \\ \hline 0 & 0 & 0 \end{array}\right] \begin{Bmatrix} u_\ell \\ \hline u_r \\ \hline u_e^a \end{Bmatrix} + \begin{Bmatrix} P_\ell^a \\ \hline P_r^a \\ \hline 0 \end{Bmatrix} \tag{14}$$

The partitions $[K_{\ell r}^a]$ and $[K_{rr}^a]$ of the aerodynamic stiffness matrix are not required because the rigid body displacements $\{u_r\}$ are assumed to be zero in calculating the loads $\{P^a_\ell\}$ and $\{P^a_r\}$. The non-zero partitions may be written as

$$\left[\begin{array}{c|c} K^a_{\ell\ell} & K^a_{\ell e} \\ \hline K^a_{\ell r} & K^a_{re} \end{array}\right] = -q \left[\begin{array}{c|c} Q_{\ell\ell} & Q_{\ell e} \\ \hline Q_{r\ell} & Q_{re} \end{array}\right] \tag{15}$$

The [Q] matrices are formed in the aerodynamic matrix processor by means of the formulas

$$\left.\begin{aligned} [Q_{\ell\ell}] &= [G_{k\ell}]^T[S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{k\ell}] \\ [Q_{\ell e}] &= [G_{k\ell}]^T[S_{kj}][A_{jj}]^{-1}[D_{je}] \\ [Q_{r\ell}] &= [G_{kr}]^T[S_{kj}][A_{jj}]^{-1}[D_{jk}][G_{k\ell}] \\ [Q_{re}] &= [G_{kr}]^T[S_{kj}][A_{jj}]^{-1}[D_{je}] \end{aligned}\right\} \tag{16}$$

The matrices $[G_{k\ell}]$ and $[G_{kr}]$ are formed in the Geometry Interpolator. If the [Q] matrices are computed for Mach numbers other than those desired, they are interpolated to the desired Mach numbers in the Aerodynamic Matrix Interpolator.

Expressions for the static aerodynamic load vectors are

$$\left\{\begin{array}{c} P^a_\ell \\ \hline P^a_r \end{array}\right\} = q \left[\begin{array}{c} Q_{\ell j} \\ \hline Q_{rj} \end{array}\right] \{w^g_j\} \tag{17}$$

where

$$\left.\begin{aligned} [Q_{\ell j}] &= [G_{k\ell}]^T[S_{kj}][A_{jj}]^{-1} \\ [Q_{rj}] &= [G_{kr}]^T[S_{kj}][A_{jj}]^{-1} \end{aligned}\right\} \tag{18}$$

$[Q_{\ell j}]$ and $[Q_{rj}]$ are formed in the Aerodynamic Matrix Processor and interpolated in the Aerodynamic Matrix Interpolator.

B.2 General Problem Formulation

Static aeroelastic problems may be classified as follows:

a. Calculation of static response.

b. Calculation of stability and control derivatives, i.e., the calculation of changes in the aerodynamic loading (and, more particularly, of changes in its resultants) due to small changes in the motions of the vehicle and of control surface deflections.

c. Divergence.

Each of the static aeroelastic problems is further classified as to whether the structure is supported, or free to move. If it is free to move, the inertia forces due to (steady) accelerations must be taken into account. A further complication arises if control surface deflections are used to trim out unwanted accelerations. All of these cases can be treated with just three NASTRAN rigid formats as follows:

7A	Aeroelastic Divergence
8A	Untrimmed Static Aeroelastic Response
9A	Trimmed Static Aeroelastic Response

It will be shown that the calculation of stability and control derivatives is a special case of Untrimmed Static Aeroelastic Response, and that solutions with supported structure are special cases of rigid formats 8A and 9A.

The equilibrium equation for the vector of structural degrees of freedom, $\{u_\ell\}$, which are measured relative to the rigid body motions, $\{u_r\}$, may be written

$$[K_{\ell\ell} + K^2_{\ell\ell}]\{u_\ell\} = \{P^s_\ell\} + \{F^i_\ell\} + \{F^a_\ell\} - [K^2_{\ell e}]\{u_e\} \tag{19}$$

where

$[K_{\ell\ell}]$ = structural stiffness matrix

$\left.\begin{matrix}[K^2_{\ell\ell}] \\ [K^2_{\ell e}]\end{matrix}\right\}$ = direct input stiffness matrices, supplied by user

$\{P^s_\ell\}$ = structural load vector

$\{F^i_\ell\}$ = inertia force vector due to rigid body accelerations

$\{F^a_\ell\}$ = aerodynamic force vector, including static terms and terms due to motions, see Eq. (14).

The equilibrium matrix equation for the rigid body motions is

$$\begin{aligned}[m_r]\{\ddot{u}_r\} &= \{P^s_r\} + \{F^a_r\} + [K^2_{r\ell}]\{u_\ell\} + [K^2_{re}]\{u_e\} \\ &+ [D]^T\left\{\{P^s_\ell\} + \{F^a_\ell\} - [K^2_{\ell\ell}]\{u_\ell\} - [K^2_{\ell e}]\{u_e\}\right\}\end{aligned} \tag{20}$$

where

$[m_r]$ = rigid body mass matrix

$\{P^s_r\}$ = vector of static loads applied directly to $\{u_r\}$

$\{F^a_r\}$ = aerodynamic force vector, applied directly to $\{u_r\}$

$\{\ddot{u}_r\}$ = vector of rigid body accelerations

$\left.\begin{matrix}[K^2_{r\ell}] \\ [K^2_{re}]\end{matrix}\right\}$ = direct input stiffness matrices, supplied by user

$[D]$ = rigid body matrix such that

$$\{\ddot{u}_\ell\} = [D]\{\ddot{u}_r\} \tag{21}$$

The vector premultiplied by $[D]^T$ in Eq. (20) is the vector of forces applied to $\{u_\ell\}$, excluding only elastic structural forces. [D] is computed from the structural stiffness matrix in the static portion of NASTRAN.

The inertia force vector in Eq. (19) is

$$\{F_\ell^i\} = - [M_{\ell r}]\{\ddot{u}_r\} - [M_{\ell\ell}][D]\{\ddot{u}_r\} \tag{22}$$

where $[M_{\ell\ell}]$ and $[M_{\ell r}]$ are partitions of the structural mass matrix.

For the most general case, the degrees of freedom will be

$$\{u_d\} = \begin{Bmatrix} u_\ell \\ \ddot{u}_r \\ u_e \end{Bmatrix} \tag{23}$$

The vector of extra points $\{u_e\}$ is further partitioned into a set $\{u_e^a\}$ that produces automatically calculated aerodynamic forces (see Eq. (4)) and a set $\{u_e^o\}$ that does not. Thus, in the most general case

$$\{u_d\} = \begin{Bmatrix} u_\ell \\ \ddot{u}_r \\ u_e^a \\ u_e^o \end{Bmatrix} \tag{24}$$

The aerodynamic force vectors include constant terms and terms proportional to displacements as shown in Eq. (14). The direct input stiffness matrices relating to extra points are similarly partitioned, so that

$$[K^2_{dd}] = \begin{array}{c} \begin{array}{cccc} u_\ell & \ddot{u}_r & u^a_e & u^o_e \end{array} \\ \left[\begin{array}{c|c|c|c} K^2_{\ell\ell} & 0 & K^2_{\ell a} & K^2_{\ell o} \\ \hline K^2_{r\ell} & 0 & K^2_{ra} & K^2_{ro} \\ \hline K^2_{a\ell} & 0 & I & K^2_{ao} \\ \hline K^2_{o\ell} & 0 & K^2_{oa} & K^2_{oo} \end{array}\right] \end{array} \begin{Bmatrix} u_\ell \\ \hline \ddot{u}_r \\ \hline u^a_e \\ \hline u^o_e \end{Bmatrix} \tag{25}$$

The identity matrix in the diagonal partition for $\{u^a_e\}$ is imposed in order to provide the following formula

$$\{u^a_e\} = \{P^a_e\} - [K^2_{a\ell}]\{u_\ell\} - [K^2_{ao}]\{u^o_e\} \tag{26}$$

where all coefficients, including the load vector $\{P^a_e\}$ are supplied by the user.

The equilibrium equations, Eqs. (19), (20), and (26), and a similar equation for $\{u^o_e\}$, may now be brought together into the single matrix equation

$$\left[\begin{array}{c|c|c|c} K_{\ell\ell} + K^2_{\ell\ell} + K^a_{\ell\ell} & M_{\ell r} + M_{\ell\ell}D & K^2_{\ell a} + K^a_{\ell a} & K^2_{\ell o} \\ \hline \begin{array}{c} D^T(K^2_{\ell\ell} + K^a_{\ell\ell}) \\ + K^2_{r\ell} + K^a_{r\ell} \end{array} & m_r & \begin{array}{c} D^T(K^2_{\ell a} + K^a_{\ell a}) \\ + K^2_{ra} + K^a_{ra} \end{array} & D^T K^2_{\ell o} + K^2_{ro} \\ \hline K^2_{a\ell} & 0 & I & K^2_{ao} \\ \hline K^2_{o\ell} & 0 & K^2_{oa} & K^2_{oo} \end{array}\right] \begin{Bmatrix} u_\ell \\ \hline \ddot{u}_r \\ \hline u^a_e \\ \hline u^o_e \end{Bmatrix} = \{P_d\} \tag{27}$$

where the load vector

$$\{P_d\} = \left\{\begin{array}{c} P^s_\ell + P^a_\ell \\ \hdashline P^s_r + P^a_r + D^T(P^s_\ell + P^a_\ell) \\ \hdashline P^a_e \\ \hdashline P^o_e \end{array}\right\} \tag{28}$$

The following special cases are noted:

1. Aeroelastic response of a supported structural component: in this case $\{\ddot{u}_r\}$ does not exist.

2. Stability and control derivatives: the inputs are selected unit values of the components of $\{P^a_e\}$, which (if $[K^2_{a\ell}] = 0$ and $[K^2_{ao}] = 0$) produce unit values of the aerodynamic "extra" degrees of freedom. The outputs are the components of $\{\ddot{u}_r\}$ or, alternatively, the force resultants,

$$\{F_r\} = [m_r]\{\ddot{u}_r\} \tag{29}$$

3. Divergence: In this case $\{P_d\} = 0$ and values of the dynamic pressure are sought which will render the matrix in Eq. (27) singular. Procedural details are discussed in Appendix C.

4. Trimmed Static Aeroelastic Response: In this case $\{\ddot{u}_r\} = 0$. Special procedures are required, as described in Section B.3 below.

5. Untrimmed Static Aeroelastic Response: This is the general case illustrated by Eq. (27).

B.3 Formulation of the Static Aeroelastic Problem in Trimmed Flight

For a trimmed flight condition, it is assumed that $\{\ddot{u}_r\} = 0$. The rigid body equilibrium condition, Eq. (20), must, however, still be satisfied, and this can be done only if the number of aerodynamic variables, $\{u_e^a\}$, equals or exceeds the number of elements of $\{\ddot{u}_r\}$. Let $\{u_e\}$ be partitioned into two parts

$$\{u_e\} = \begin{Bmatrix} u_e^t \\ \hline u_e^u \end{Bmatrix} \tag{30}$$

where

$\{u_e^t\}$ is a subset of $\{u_e^a\}$ used for trimming the vehicle. The number of members of $\{u_e^t\}$ must equal the number of members of $\{\ddot{u}_r\}$.

$\{u_e^u\}$ is the union of the remaining members of $\{u_e^a\}$ and the set $\{u_e^o\}$ of non-aerodynamic extra points (see Eq. (24)).

The direct input and aerodynamic matrices $[K^2]$ and $[K^a]$ are similarly partitioned:

$$[K_{dd}^2] = \begin{array}{c} \begin{array}{ccc} u_\ell & u_e^t & u_e^u \end{array} \\ \left[\begin{array}{c|c|c} K_{\ell\ell}^2 & 0 & K_{\ell u}^2 \\ \hline K_{r\ell}^2 & 0 & K_{ru}^2 \\ \hline K_{u\ell}^2 & K_{ut}^2 & K_{uu}^2 \end{array}\right] \end{array} \begin{Bmatrix} u_\ell \\ \hline u_r \\ \hline u_e^u \end{Bmatrix} \tag{31}$$

$$
[K^a_{dd}] = \begin{array}{c} \begin{array}{ccc} u_\ell & u^t_e & u^u_e \end{array} \\ \left[\begin{array}{c|c|c} K^a_{\ell\ell} & K^a_{\ell t} & K^a_{\ell u} \\ \hline K^a_{r} & K^a_{rt} & K^a_{ru} \\ \hline 0 & 0 & 0 \end{array}\right] \end{array} \begin{Bmatrix} u_\ell \\ \hline u_r \\ \hline u^u_e \end{Bmatrix} \tag{32}
$$

The complete equilibrium equations are

$$
\left[\begin{array}{c|c|c} K_{\ell\ell} + K^2_{\ell\ell} + K^a_{\ell\ell} & K^a_{\ell t} & K^a_{\ell u} + K^2_{\ell u} \\ \hline K^2_{r\ell} + K^a_{r\ell} + D^T(K^2_{\ell\ell} + K^a_{\ell\ell}) & K^a_{rt} + D^T K^a_{\ell t} & K^2_{ru} + K^a_{ru} + D^T(K^a_{\ell u} + K^2_{\ell u}) \\ \hline K^2_{u\ell} & K^2_{ut} & K^2_{uu} \end{array}\right] \begin{Bmatrix} u_\ell \\ \hline u^t_e \\ \hline u^u_e \end{Bmatrix} = \{P_d\} \tag{33}
$$

where

$$
\{P_d\} = \begin{Bmatrix} P^s_\ell + P^a_\ell \\ \hline P^s_r + P^a_r + D^T(P^s_\ell + P^a_\ell) \\ \hline P^u_e \end{Bmatrix} \tag{34}
$$

All direct input matrices $[K^2]$ and the load vector $\{P^u_e\}$ are user supplied. The partitions $[K^2_{\ell t}]$ and $[K^2_{rt}]$ are forbidden in order that the aerodynamic trim variables $\{u^t_e\}$ be determined by aerodynamic relationships only. The aerodynamic variables included in $\{u^u_e\}$ may be set to specified values or they may be slaved to structural deformations and/or to the aerodynamic trim variables $\{u^t_e\}$ by the relationship given by the bottom row of Eq. (33), i.e.,

$$
\{u^u_e\} = [K^2_{uu}]^{-1}\left\{\{P^u_e\} - [K^2_{u\ell}]\{u_\ell\} - [K^2_{ut}]\, u^t_e\right\} \tag{35}
$$

Evaluation of the parameters in this equation is entirely controlled by the user and not by an automatic aerodynamic procedure, except that the partition of $[K^2_{uu}]$ corresponding to the remaining members of $\{u^a_e\}$ will be automatically set equal to an identity matrix if no values are supplied by the user.

As an example, suppose that the rolling velocity, p, is placed in $\{u^u_e\}$ and that the aileron deflection, δa, is placed in $\{u^t_e\}$. If the component of $\{P^u_e\}$ is set equal to a desired value of steady rolling velocity and if $[K^2_{uu}]$ is an identity matrix, the program will then compute, in addition to other quantities, the aileron deflection required to produce the desired steady rolling velocity.

B.4 Procedures for Response Solutions

Equations (27) and (33) both have the general form

$$[K_{dd}]\{u_d\} = \{P_d\} \tag{36}$$

or, in terms of the matrices contributing to $[K_{dd}]$,

$$[K^1_{dd} + K^2_{dd} + K^a_{dd}]\{u_d\} = \{P_d\} \tag{37}$$

where

$[K^1_{dd}]$ is the structural stiffness matrix.

$[K^2_{dd}]$ represents direct input terms supplied by the user and also the inertia and identity matrix terms in Eq. (27).

$[K^a_{dd}]$ is the aerodynamic stiffness matrix.

The standard procedure for solving Eq. (36) is to decompose $[K_{dd}]$ into its triangular factors

$$[K_{dd}] = [L][U] \tag{38}$$

where [L] is a lower triangle with unit elements on the diagonal and [U] is an upper triangle. The solution algorithm is to solve

$$[L]\{y\} = \{P_d\} \tag{39}$$

and

$$[U]\{u_d\} = \{y\} \tag{40}$$

successively by forward and backward substitution. Since $[K_{dd}]$ is an unsymmetrical matrix, partial pivoting (i.e., row interchanges) is employed in the triangular decomposition.

The above solution technique may consume excessive computer time because the aerodynamic stiffness matrix $[K^a_{dd}]$ may, in contrast with $[K^1_{dd}]$ and $[K^2_{dd}]$, be relatively full. For such conditions an iterative procedure, based on the assumption that the terms in $[K^a_{dd}]$ are small compared to those in $[K^1_{dd}]$, may be more efficient. For example, consider the algorithm

$$[K^1_{dd} + K^2_{dd}]\{u^n_d\} = \{P_d\} - [K^a_{dd}]\{u^{n-1}_d\} \tag{41}$$

where $\{u^n_d\}$ is the nth iterate. If the algorithm converges to satisfactory accuracy in a small number of iterations, it may consume less time than the standard algorithm (Eqs. (39) and (40)), provided that the difference in time between the decomposition of $[K^1_{dd} + K^2_{dd}]$ and of $[K^1_{dd} + K^2_{dd} + K^a_{dd}]$ is substantial. An additional advantage accrues if a number of solutions corresponding to different $[K^a_{dd}]$ matrices are desired, since $[K^1_{dd} + K^2_{dd}]$ need only be decomposed once.

Details of the iteration algorithm, including convergence tests, are developed below. A practical form of the algorithm may be stated as follows:

1. Decompose $[K^1_{dd} + K^2_{dd}]$

2. Solve $[K^1_{dd} + K^2_{dd}]\{u^o_d\} = \{P_d\}$

3. Form $\{P^1_d\} = -[K^a_{dd}]\{u^o_d\}$

4. Solve $[K^1_{dd} + K^2_{dd}]\{\delta u^1_d\} = \{P^1_d\}$

5. Form $\{u^1_d\} = \{u^o_d\} + \{\delta u^1_d\}$

6. Repeat, for $n \geq 2$, until convergence:

   a. Form $\{P^n_d\} = -[K^a_{dd}]\{\delta u^{n-1}_d\}$

   b. Solve $[K^1_{dd} + K^2_{dd}]\{\delta u^n_d\} = \{P^n_d\}$

   c. Form $\{u^n_d\} = \{u^{n-1}_d\} + \{\delta u^n_d\}$

Knowledge is required of the conditions under which the algorithm converges in order to use it intelligently. The examination of stability will produce appropriate convergence tests as a byproduct.

The heart of the algorithm is, combining steps 6a and 6b,

$$[K^1_{dd} + K^2_{dd}]\{\delta u^n_d\} = -[K^a_{dd}]\{\delta u^{n-1}_d\} \tag{42}$$

In this form the algorithm closely resembles the power method used in eigenvalue extraction. Its stability may be examined by methods similar to those used to justify the power method (see, for example, the NASTRAN Theoretical Manual, Section 10.4).

The eigenvalue problem associated with Eq. (42) is

$$[K^1_{dd} + K^2_{dd} + \lambda K^a_{dd}]\{u\} = 0 \tag{43}$$

which is just the static aeroelastic divergence problem. Let the iterates $\{\delta u^n_d\}$ and $\{\delta u^{n-1}_d\}$ be expanded in terms of the eigenvectors $\{\phi_i\}$ of Eq. (43)

$$\{\delta u^n_d\} = \sum_i \alpha^n_i \{\phi_i\} \tag{44}$$

$$\{\delta u^{n-1}_d\} = \sum_i \alpha^{n-1}_i \{\phi_i\} \tag{45}$$

It may be proved quite generally (see, for example, Section 10.4.4.3 of the NASTRAN Theoretical Manual) that

$$\alpha^n_i = \frac{1}{\lambda_i} \alpha^{n-1}_i \tag{46}$$

where λ_i is the eigenvalue corresponding to the eigenvector, $\{\phi_i\}$.

The solution vector corresponding to the nth iteration is

$$\{u^n_d\} = \{u^o_d\} + \sum_{m=1}^{n} \{\delta u^m_d\} \tag{47}$$

or, using Eq. (44)

$$\{u^n_d\} = \{u^o_d\} + \sum_{m=1}^{n} \sum_i \alpha^m_i \{\phi_i\} \tag{48}$$

$\{u^o_d\}$ may also be expanded in terms of eigenvectors

$$\{u^o_d\} = \sum_i \alpha^o_i \{\phi_i\} \tag{49}$$

so that, substituting into Eq. (48),

$$\{u_d^n\} = \sum_i \left[\alpha_i^o + \sum_{m=1}^{n} \alpha_i^m\right]\{\phi_i\} \tag{50}$$

and, using Eq. (46) and the fact that $\{u_d^o\}$ appears on the right hand side of the first iteration,

$$\{u_d^n\} = \sum \alpha_i^o\left(1 + \frac{1}{\lambda_i} + \frac{1}{\lambda_i^2} + \ldots \frac{1}{\lambda_i^n}\right)\{\phi_i\} \tag{51}$$

The geometric series in Eq. (51) has the limit

$$\lim_{n\to\infty}\left(1 + \sum_{m=1}^{n}\left(\frac{1}{\lambda_i}\right)^m\right) = \frac{1}{1 - \frac{1}{\lambda_i}} \tag{52}$$

provided that $|\lambda_i| > 1$. Otherwise it does not converge.

If the aircraft is statically stable (i.e. nondivergent) then all real positive $\lambda_i > 1$. Convergence of the algorithm additionally requires that all real negative $\lambda_i < -1$ and that all complex eigenvalues (if any exist) satisfy $|\lambda_i| > 1$. These nonphysical requirements may not be satisfied even if the aircraft is stable. They will not, for example, be satisfied if a negative dynamic pressure would produce divergence. This may well be the case for control surfaces with weak elastic restraints. Thus, it is not possible to guarantee convergence of the algorithm for all statically stable aircraft, and it is, therefore, necessary to provide the standard solution algorithm (Eqs. (39) and (40)) as an alternate.

It remains to develop suitable convergence tests for the algorithm. The remainder of the geometric series in Eq. (51) is, after the nth iteration,

$$R_n = \left(\frac{1}{\lambda_i}\right)^n \cdot \frac{1}{\lambda_i - 1} \tag{53}$$

The error in $\{u_d^n\}$ is, therefore,

$$\{\varepsilon_d^n\} = \{u_d\} - \{u_d^n\} = \sum_i \alpha_i^o \frac{1}{\lambda_i^n(\lambda_i - 1)} \{\phi_i\} \tag{54}$$

or, using Eq. (46)

$$\{\varepsilon_d^n\} = \sum_i \frac{\alpha_i^n}{(\lambda_i - 1)} \{\phi_i\} \tag{55}$$

After many steps of iteration the eigenvectors whose eigenvalues are closest to 1.0 will dominate the error vector. If we assume that only one prominent eigenvector remains, then, using Eq. (44)

$$\{\varepsilon_d^n\} \simeq \frac{\alpha_1^n}{\lambda_1 - 1} \{\phi_i\} = \frac{1}{\lambda_1 - 1} \{\delta u_d^n\} \tag{56}$$

If a means for estimating λ_1 can be found, Eq. (56) provides an estimate of the error in the solution after n iterations. A good single number for estimating the error, that automatically normalizes the elements in the error vector, is

$$\varepsilon_n = \frac{\{\varepsilon_d^n\}^T\{P_d^n\}}{\{u_d^n\}^T\{P_d^n\}} \simeq \frac{1}{\lambda_1^{(n)} - 1} \frac{\{\delta u_d^n\}^T\{P_d^n\}}{\{u_d^n\}^T\{P_d^n\}} \tag{57}$$

$\lambda_1^{(n)}$ is the estimate of λ_1 obtained in the nth iteration. It is evaluated as follows. By virtue of Eq. (45), the matrix product

$$\{\delta u_d^{n-1}\}^T\{P_d^n\} = -\{\delta u_d^{n-1}\}^T[K_{dd}^a]\{\delta u_d^{n-1}\} = -\{\sum_i \alpha_i^{n-1}\phi_i\}^T[K_{dd}^a]\{\sum_j \alpha_j^{n-1}\phi_j\} \tag{58}$$

If it is assumed that the iteration has progressed to the point where only a single prominent eigenvector remains in $\{\delta u_d^{n-1}\}$,

$$\{\delta u_d^{n-1}\}^T\{P_d^n\} \approx -(\alpha_1^{n-1})^2\{\phi_1\}^T[K_{dd}^a]\{\phi_1\} \tag{59}$$

Likewise, using Eqs. (44) and (45):

$$\{\delta u_d^n\}^T\{P_d^n\} \approx -\frac{1}{\lambda_1}(\alpha_1^{n-1})^2\{\phi_1\}^T[K_{dd}^a]\{\phi_1\} \tag{60}$$

so that, dividing Eq. (59) by Eq. (60),

$$\lambda_1 \approx \lambda_1^{(n)} = \frac{\{\delta u_d^{n-1}\}^T\{P_d^n\}}{\{\delta u_d^n\}^T\{P_d^n\}} \tag{61}$$

The proposed convergence tests, to be used after all iterations for $n \geqslant 2$, are

a. Form $\lambda_1^{(n)}$ according to Eq. (61).

b. If $n \geqslant 3$ and $|\lambda_1^{(n)}| < 1$, abort the iteration.

c. Form ε_n according to Eq. (57).

d. If $|\varepsilon_n| < \varepsilon$ where ε is an user-supplied parameter, accept $\{u_d^n\}$ as the solution. If $|\varepsilon_n| > \varepsilon$, continue to iterate.

The number of additional iterations required to obtain a converged solution may be estimated as follows. From Eqs. (46) and (56) the ratio of the error after the nth and the (n+k)th iterations is approximately

$$\frac{\varepsilon_{n+k}}{\varepsilon_n} = \left(\frac{1}{\lambda_1}\right)^k \tag{62}$$

An estimate of the required number of additional iterations to make ε_{n+k} equal ε is, therefore

$$k = \frac{\log\left(\frac{\varepsilon_n}{\varepsilon}\right)}{\log \lambda_1^{(n)}} \tag{63}$$

If the user specifies a maximum number of iterations, M, the solution should be aborted if $n + k > M$. The following table gives values of k as a function of $\varepsilon_n/\varepsilon$ and $\lambda_1^{(n)}$.

$\lambda_1^{(n)}$ \ $\varepsilon_n/\varepsilon$	10	100	1000	10,000
	k			
1.1	24.4	48.8	73.2	97.6
1.2	12.8	25.6	38.4	51.2
1.5	5.7	11.4	17.1	22.8
2.0	3.3	6.6	9.9	13.2
5.0	1.4	2.8	4.2	5.6
10.0	1.0	2.0	3.0	4.0

It will be noted that the required number of iterations increases rapidly as the divergence limit is approached.

APPENDIX C

APPENDIX C

Notes on the Calculation of Static Aeroelastic Divergence

The problem is to calculate the eigenvalues and eigenvectors of

$$[K^1_{dd} + K^2_{dd} + \lambda K^a_{dd}]\{u\} = 0 \qquad (1)$$

The following observations are pertinent:

1. $[K^1_{dd}]$ is real, and symmetric. $[K^2_{dd}]$ and $[K^a_{dd}]$ are real but not, in general, symmetric.

2. The eigenvalues may be positive real, negative real, or they may occur in conjugate complex pairs. The physical significance of negative real and complex eigenvalues is not apparent.

3. The user is interested in positive real eigenvalues that occur in an interval $0 < \lambda < \lambda_b$ and he has particular interest in the smallest positive real eigenvalue. He may also wish to know of the existence of any complex eigenvalues with real parts in the interval $0 < \mathrm{Re}\lambda < \lambda_b$.

4. The matrices $[K^1_{dd}]$ and $[K^2_{dd}]$ are sparce whereas $[K^a_{dd}]$ may either be sparce or dense.

It is concluded from the above observations that some form of the power method is well suited to the problem. Neither of the forms of the power method provided with NASTRAN (Real Inverse Power and Complex Inverse Power) are directly applicable. The Real Inverse Power method assumes that the matrices are symmetric and that the eigenvalues are positive real. The

Complex Inverse Power method is used to solve problems of the form

$$[\lambda^2 M + \lambda B + K]\{u\} = 0 \qquad (2)$$

It assumes that [M] is not null (in making convergence tests) and that complex arithmetic is required.

A new version of the power method is, therefore, proposed for the calculation of static aeroelastic divergence. A brief investigation has been made of the following algorithm, to which the existing Complex Inverse Power method reduces when [M] = 0.

1. Let the problem be stated as

$$[K + \lambda B]\{u\} = 0 \qquad (3)$$

2. Let

$$\lambda = \lambda_o + \Lambda \qquad (4)$$

where λ_o is called the shift point. λ_o is a real number, greater than or equal to zero.

3. The algorithm is

$$[K + \lambda_o B]\{w_n\} = -[B]\{u_{n-1}\} \qquad (5)$$

$$\{\bar{u}_n\} = \frac{1}{c_n}\{w_n\} \qquad (6)$$

where c_n is the largest element of $\{w_n\}$.

The algorithm converges to the eigenvector whose eigenvalue is closest to the shift point, provided that the closest eigenvalue is real. An estimate of the shifted eigenvalue at any iteration may be obtained from

$$\Lambda_1 = \frac{1}{c_n} \frac{\{u_n\}^T [B] \{u_{n-1}\}}{\{u_n\}^T [B] \{u_n\}} \tag{7}$$

The iteration will be continued until three successive values of Λ_1 are separated by an amount less than ε, specified by the user. If this criterion is not satisfied within a number of iterations, N_i, specified by the user, the iteration will be terminated. In either case the last vector, $\{u_N\}$, and <u>all</u> of the successive estimates of Λ_1 will be output. The rate of convergence is approximately proportional to the ratio of the two closest eigenvalues, Λ_2/Λ_1. This produces the difficulty that, since the shift point, λ_o, is real, it will be equidistant from any pair of conjugate complex roots and Eq. (7) will not converge if the closest roots are complex. The existence of this situation will be clear from the differences between the successive estimates of Λ_1.

There is an advantage in selecting the origin, $\lambda = 0$, as the shift point because that selection restricts the triangular decomposition implied by Eq. (5) to $[K^1_{dd} + K^2_{dd}]$ which will be sparce and narrowly banded whereas $[K^a_{dd}]$ is relatively full. This choice may not be practical, however, due to the possible existence of a negative eigenvalue of smaller magnitude than the smallest positive eigenvalue, with the result that the iteration algorithm will converge on a negative eigenvalue (see discussion in Appendix B).

Another important point is that convergence to any eigenvalue can be speeded if the user places a shift point at its estimated location.

APPENDIX D

APPENDIX D

Notes on the Calculation of Transient Aeroelastic Response

An important difficulty occurs in the solution of transient aeroelastic problems in that all advanced aerodynamic theories are formulated in the frequency domain, i.e., steady state oscillatory motion is assumed. If the aerodynamic theory is so formulated, its application to transient analysis requires the assertion that the functional relationships expressed in terms of frequency, ω, can be continued analytically from the imaginary axis, $p = i\omega$, into the complex plane. If this is true, then the following two basic methods are available for the solution of transient problems.

1. Employ the Fourier transform technique, i.e., calculate the Fourier transforms of the excitation functions, obtain the frequency response of the system to the Fourier transforms of the excitation functions, and calculate the inverse Fourier transforms (i.e., time histories) of the response functions.

2. Perform an inverse Laplace transformation of the expressions for the aerodynamic forces, obtaining time-dependent functions which are then inserted into the equations of motion of the system and integrated to obtain the time histories of the response functions.

Each method has advantages and disadvantages. The main advantage of of the second method is that it can be applied to nonlinear problems or to problems with time varying coefficients whereas the first method cannot. It can, in addition, be used to obtain the response of unstable systems, whereas the first method cannot. The main advantage of the first method is that it accepts the frequency-domain aerodynamic formulations without difficulty or approximation. A rigorous application of the second method, on the other hand, leads to very cumbersome mathematical procedures for all but the simplest theories. Practical application of the second method requires, therefore, additional approximations. These approximations are not serious for strip theory but they may be unacceptable for more sophisticated theories.

Some of the details of each of the methods are examined below.

D.1 Fourier Transform Method

The problem to be solved may be stated as follows:

$$[A]\{u(t)\} = \{P(t)\} \quad . \tag{1}$$

where $\{u(t)\}$ is a vector of displacements, $\{P(t)\}$ is a vector of time-dependent applied forces, and $[A]$ is a matrix of linear integro-differential operators, including convolution integrals. Let $\{P(\omega)\}$ be the Fourier transform of $\{P(t)\}$ and let $\{u(\omega)\}$ be the Fourier transform of $\{u(t)\}$. Then, if the initial conditions for $\{u(t)\}$ and its first derivative are zero, the Fourier transform of $[A]\{u(t)\}$ can be written in the form $[A(\omega)]\{u(\omega)\}$. Thus,

$$\{u(\omega)\} = [A(\omega)]^{-1}\{P(\omega)\} = [H(\omega)]\{P(\omega)\} \tag{2}$$

is the Fourier transform of $\{u(t)\}$. The matrix $[A(\omega)]$ is the dynamic matrix used in frequency response analysis, i.e.,

$$[A(\omega)] = [-\omega^2 M_{dd} + i\omega B_{dd} + K_{dd}] \tag{3}$$

for a direct analysis, or

$$[A(\omega)] = [-\omega^2 M_{hh} + i\omega B_{hh} + K_{hh}] \qquad (4)$$

for a modal analysis. The mass matrices $[M_{dd}]$ or $[M_{hh}]$ include the aerodynamic mass matrix as a term.

The Fourier transform of the load vector is obtained by the defining equation

$$\{P(\omega)\} = \int_0^{\infty} \{P(t)\} e^{-i\omega t}\, dt \qquad (5)$$

Once $\{u(\omega)\}$ has been evaluated by Eq. (2), the real time solution is obtained from the inverse Fourier transform

$$\{u(t)\} = \frac{1}{\pi} \int_0^{\infty} R\ell[e^{i\omega t}\{u(\omega)\}]d\omega \qquad (6)$$

where $R\ell[\ ]$ signifies the real part of [ ].

The total calculation consists of the following three steps:

1. Evaluate the Fourier transform of the applied load vector at a sequence of frequencies, ω_1, ω_2, ---, ω_N, in a range $0 \le \omega_n < \omega_{max}$. The number and distribution of frequencies is selected as a compromise between sampling error and computational efficiency.

2. Form the dynamic matrix $[A(\omega_n)]$ and solve the matrix equation

$$[A(\omega_n)]\{u(\omega_n)\} = \{P(\omega_n)\} \qquad (7)$$

   for each selected frequency.

3. Evaluate the transient response by means of an appropriate numerical approximation to Eq. (6).

D.2 Fourier Transforms of Load Vectors

Fortunately the applied time-dependent load vectors used in NASTRAN have relatively simple Fourier transforms. Two separate forms are provided. The first, or general, form is (see NASTRAN Theoretical Manual p. 11.1-1):

$$\begin{aligned} \{P_j(t)\} &= \{A_j\}F(t-\tau_j) \qquad 0 < t-\tau_j < T \\ &= 0 \qquad 0 > t-\tau_j ; \quad t-\tau_j > T \end{aligned} \tag{8}$$

where $\{A_j\}$ and $\{\tau_j\}$ are tabulated coefficients that may be different for each loaded degree of freedom (j). F(t) is a tabulated function of time that is linearly interpolated between entries. The maximum time limit, T, is introduced in order to make the function transformable. T may, if desired, be set equal to the requested duration of the transient solution. The form provided by Eq. (8) is particularly useful for loads due to traveling waves. In such applications, F(t) may represent the pressure produced by the wave at some arbitrary point, A_j is the exposed area associated with the jth degree of freedom, and τ_j is the travel time required for the wave to travel from the arbitrary point to the jth degree of freedom.

The function F(t) has constant slope between adjacent breakpoints, t_m and t_{m+1}. The Fourier transform of the load vector is, by straightforward application of Eq. (5),

$$\{P(\omega)\} = \{A_j e^{-i\omega\tau_j}\}G(\omega) \tag{9}$$

where

$$G(\omega) = \frac{1}{i\omega}\left(F(0) - F(T)e^{-i\omega T}\right) + \frac{1}{\omega^2}\sum_{m=0}^{M-1}\frac{\Delta F_m}{\Delta t_m}\left(e^{-i\omega t_{m+1}} - e^{-i\omega t_m}\right) \tag{10}$$

and

$$\Delta F_m = F(t_{m+1}) - F(t_m) \tag{11}$$

$$\Delta t_m = t_{m+1} - t_m \tag{12}$$

Equation (9) is in standard form for frequency response analysis with NASTRAN, see Eq. (2), p. 12.1-1, of the NASTRAN Theoretical Manual.

The second form of the applied, time-dependent load vector provided with NASTRAN is

$$\begin{aligned} \{P_j(t)\} &= \{A_j\}(\bar{t})^n e^{\alpha t} \cos(\omega_k \bar{t} + \phi) \qquad 0 < \bar{t} < T_2 - T_1 \\ &= 0 \qquad 0 > \bar{t}\ ;\ \text{and } \bar{t} > T_2 - T_1 \end{aligned} \tag{13}$$

where

$$\bar{t} = t - T_1 - \tau_j \tag{14}$$

The six constants T_1, T_2, n, α, ω_k and ϕ may be selected to provide a wide variety of wave shapes. The coefficients A_j and τ_j have the same significance as they do for the general forcing function in Eq. (8). Any number of load vectors of either or both forms may be applied simultaneously.

In order to avoid difficult integrations, it will be assumed that the exponent n in Eq. (13) is an integer, greater than or equal to zero. The Fourier transform of the load vector is in standard NASTRAN form,

$$\{P(\omega)\} = \{A_j\, e^{-i\omega\tau_j}\}G(\omega) \tag{15}$$

where

$$G(\omega) = \frac{1}{2} e^{-i\omega T_1} [E(T_2 - T_1, a_1) + E(T_2 - T_1, a_2)] \qquad (16)$$

$$a_1 = \alpha + i(\omega_k t + \phi - \omega t) \qquad (17)$$

$$a_2 = \alpha - i(\omega_k t + \phi + \omega t) \qquad (18)$$

and

$$E(T, a) = e^{aT} \left[\frac{T^n}{a} - \frac{nT^{n-1}}{a^2} + \frac{n(n-1)T^{n-2}}{a^3} \cdots \right.$$

$$\left. + (-1)^{n-1} \frac{n!T}{a^n} \right] + (-1)^n \frac{n!}{a^{n+1}} (e^{aT}-1) \text{ for } n > 0 \qquad (19)$$

$$= \frac{1}{a} (e^{aT} - 1) \qquad \text{for } n = 0$$

D.3 Evaluation of the Frequency Response

The most time consuming part of the calculation is the evaluation of the frequency response to the Fourier-transformed excitation functions. The aerodynamic matrices $[A_{jj}]$ and $[D_{jk}]$, which are functions of reduced frequency, will be different for each frequency because $k = b\omega/V$ and the velocity is held fixed. Calculation of the aerodynamic mass matrices, $[M^a_{dd}]$ or $[M^a_{hh}]$, can be economized by interpolating between values tabulated for a few frequencies.

A more serious question is the selection of the frequencies at which to compute frequency response. The graph of the frequency response of a lightly damped structure will contain sharp peaks and broad valleys.

Thus, a nonuniform spacing of frequencies, with points concentrated near the peaks, will produce a more accurate transient response, for a given number of points, than will a uniform spacing. It is quite likely that the user will have advance knowledge of the location of the peaks. Even if he does not, a preliminary run can be made with a coarse mesh of points to approximately locate the peaks. In any case the user should be given the option of specifying a nonuniform spacing of frequencies. He should also have the option to merge the results for two runs with different frequencies.

Perhaps the most efficient method of calculating frequency response is to use the eigenvectors of the aerodynamically coupled structure. It is, unfortunately, impossible to calculate the eigenvectors unless the aerodynamic matrices can be evaluated for complex values of the reduced frequency. This method is not proposed.

D.4 Evaluation of the Inverse Fourier Transforms

If the frequencies are uniformly spaced, the most efficient method for computing the transient response is probably some version of the Fast Fourier Transform, Refs. 1, and 2. At present NASTRAN does not contain a Fast Fourier Transform (FFT) routine, but there should be no great difficulty in providing one because several efficient computer codes exist.

The essence of the calculation is as follows. By Eq. (6) the transient response vector, $u(t_m)$, is linearly related to the frequency response vector, $u(\omega_n)$, so that the result of a numerical approximation of Eq. (6) can be expressed in matrix form as

$$\begin{Bmatrix} u(t_1) \\ u(t_2) \\ u(t_3) \\ \vdots \\ u(t_M) \end{Bmatrix} = R\ell\ [F_{mn}] \begin{Bmatrix} u(\omega_1) \\ u(\omega_2) \\ u(\omega_3) \\ \vdots \\ u(\omega_N) \end{Bmatrix} \qquad (20)$$

where $[F_{mn}]$ is a matrix of the coefficients

$$F_{mn} = \frac{\Delta\omega}{\pi} e^{i\omega_n t_m} \qquad (21)$$

The form of Eq. (21) indicates that equal weights are given to all data points.

The calculation indicated by Eq. (20) has the discouraging aspect that M x N separate factors $e^{i\omega_n t_m}$ are required. This can be reduced to kN separate factors, where k is a small number, simply by using uniform spacing in time and frequency such that

$$\Delta\omega\Delta t = \frac{2\pi}{P} \qquad (22)$$

where P is an integer. In the FFT method, M, N, and P are all selected to be equal to 2^γ where γ is an integer. The period of the response is $T = 2\pi/\Delta\omega = N/f_{max}$. It may then be shown that $[F_{mn}]$ can be factored into

$$[F_{mn}] = [T_1][F_1][T_2][F_2] \text{ ---- } [T_\gamma][F_\gamma] \qquad (23)$$

where the $[F_k]$ matrices include only two non-zero terms in each row and the $[T_k]$ matrices perform a reordering of elements in the right-hand vector according to a fixed pattern. With these simplifications the indicated matrix multiplications are reduced to short arithmetic and logical operations. The total number of operations is proportional to $N\gamma = N \log_2 N$. The number of different trigonometric functions is equal to N. They need only be calculated once and stored, regardless of the number of output quantities that are processed.

If the frequencies are not uniformly spaced, the following methods (among others) are available.

a. Interpolate the frequency response data to a set of uniformly spaced points and use the FFT method.

b. Represent each frequency response function by a polynomial fit between data points and perform an exact integration.

The nonuniform frequency intervals have, presumably, been selected to minimize the error in method (b). The uniform frequency interval for method (a) should, therefore, be selected equal to the <u>smallest</u> interval present. Thus, the number of frequencies generated in method (a) may be much larger than the number of frequencies in method (b).

Both methods require analytic interpolation of the frequency response data. It is proposed that the data points be fitted by a cubic spline curve, i.e., by a curve that simulates the deflection of a beam passing through all of the data points. This curve has the property that its first and second derivatives are continuous at all data points. In any interval, $\omega_n < \omega < \omega_{n+1}$, the function is represented by

$$u = a_n + b_n\bar{\omega} + c_n\bar{\omega}^2 + d_n\bar{\omega}^3 \tag{24}$$

where $\bar{\omega} = \omega - \omega_n$. The second derivatives at three successive data points satisfy the relationship, for $2 \leq n \leq N-1$,

$$\Delta\omega_n u''_{n+1} + 2(\Delta\omega_n + \Delta\omega_{n-1})u''_n + \Delta\omega_{n-1}u''_{n-1} = 6\left[\frac{u_{n+1} - u_n}{\Delta\omega_n} - \frac{u_n - u_{n-1}}{\Delta\omega_{n-1}}\right] \quad (25)$$

where $\Delta\omega_n = \omega_{n+1} - \omega_n$. The boundary conditions are

$$u''_1 = u''_N = 0 \quad (26)$$

Equation (25) is a well-conditioned difference equation and the coding of a computer subroutine to solve it is a simple task.

For $\omega > \omega_N$ the response quantity is assumed to be zero. Presumably the highest frequency has been selected on the assumption that the contribution of higher frequencies to the transient response may be neglected. If the lowest frequency, ω_1, is not zero, the response must be extrapolated in the range $0 < \omega < \omega_1$ utilizing some reasonable presumptions regarding the behaviour of the response quantity near zero frequency. The response quantity will be represented by a quadratic curve that matches the displacement and slope of the cubic spline at $\omega = \omega_1$. The remaining coefficient will be selected to match assumed boundary conditions at $\omega = 0$ as follows:

for displacements, accelerations, internal forces and stresses:

$$\text{Slope, } \frac{du}{d\omega} = 0$$

for velocities

$$\text{Curvature, } \frac{d^2u}{d\omega^2} = 0, \text{ therefore the curve is a straight line.}$$

The exact integration referred to in method (b), obtained by substituting Eq. (24) into Eq. (6), is

$$u(t) = \frac{1}{\pi} R\ell \sum_{n=1}^{N-1} \int_{\omega_n}^{\omega_{n+1}} e^{i\omega t}(a_n + b_n\bar{\omega} + c_n\bar{\omega}^2 + d_n\bar{\omega}^3)d\omega \tag{27}$$

An explicit representation of this integral is as follows:

$$\begin{aligned} u(t) = \frac{1}{\pi t^4} R\ell \sum_{n=0}^{N} \Big\{ & a_n\left[-it^3\left(e^{i\omega_{n+1}t} - e^{i\omega_n t}\right)\right] \\ & + b_n\left[(t^2 - i\Delta\omega_n t^3)e^{i\omega_{n+1}t} - t^2 e^{i\omega_n t}\right] \\ & + c_n\left[(2it + 2\Delta\omega_n t^2 - i\Delta\omega_n^2 t^3)e^{i\omega_{n+1}t} - 2ite^{i\omega_n t}\right] \\ & + d_n\left[(-6 + 6i\Delta\omega_n t + 3\Delta\omega_n^2 t^2 - i\Delta\omega_n^3 t^3)e^{i\omega_{n+1}t} + 6e^{i\omega_n t}\right]\Big\} \end{aligned} \tag{28}$$

where

$$e^{i\omega_n t} = \cos\omega_n t + i\sin\omega_n t \tag{29}$$

If there is more than one response quantity, the coefficients of a_n, b_n, c_n and d_n in Eq. (28) may be held in core storage and used repeatedly.

Equation (28) is indeterminate for $t = 0$. For $t = 0$ use

$$u(t) = \frac{1}{\pi} R\ell \sum_{n=1}^{N-1}\left[\Delta\omega_n a_n + \frac{1}{2}\Delta\omega_n^2 b_n + \frac{1}{3}(\Delta\omega_n)^3 c_n + \frac{1}{4}(\Delta\omega_n)^4 d_n\right] \tag{30}$$

Evaluation of the trigonometric functions in Eq. (28) and (29) is a significant computational task since they must be evaluated NM times where N is the number of frequencies and M is the number of time steps. The calculation can be speeded, perhaps, by holding a short table of trigonometric functions in core storage and interpolating.

Each term indicated by the summation sign in Eq. (28) is evaluated NMR times where R is the number of response quantities. Thus, since there are four products in each term, the total number of complex multiplications is, approximately, 4NMR if R is large. For comparison the number of multiplications in the calculation of the frequency response is NDB^2 where D is the number of degrees of freedom and B is the semi-bandwidth. For aeroelastic problems solved by the modal method, the matrices are full and the effective bandwidth is $D/\sqrt{2}$. Thus the number of multiplications is approximately $\frac{1}{2}ND^3$. If we further assume that the number of desired response quantities is equal to the number of degrees of freedom, the ratio of computing times is approximately $8M/D^2$. Since, typically, the desired number of time steps is a few hundred, we see that the time to calculate the inverse Fourier transforms is greater than the time to calculate frequency response if there are fewer than about 50 degrees of freedom.

The ratio of the computing time by the FFT method (including interpolation of non-uniformly spaced frequencies) to the computing time by the exact integration method is of the order of $\frac{5}{M}(\log_2 N)$ and it will almost always be faster. Nevertheless the exact integration method offers greater freedom with respect to the selection of output times and it is a safer method because it makes fewer approximations. In particular, there is less danger of missing frequency response peaks. For these reasons it is recommended that the exact integration method be given higher priority.

D.5 Modifications of NASTRAN to Implement the Fourier Transform Method

Only two modifications are required. The first is to provide means for calculating the Fourier transforms of applied load vectors, see Section D.2 above. The representations of the load vectors in both the time and frequency domains are in standard NASTRAN form so that this modification is quite simple to implement.

The second modification is to provide means for calculating the time histories of response quantities by the Fast Fourier Transform method and also by the Spline Fit Integration method as described in Section D.4 above. The input data required in these calculations is already generated for use in the NASTRAN Random Analysis Module. The system implications of the modification are, therefore, slight.

D.6 Representation of Aerodynamic Forces in the Time Domain

The basic form required by the NASTRAN transient analysis module is

$$[M]\{\ddot{u}\} + [B]\{\dot{u}\} + [K]\{u\} = \{P(t)\} \tag{31}$$

where $\{u\}$ may include "extra points" as well as structural degrees of freedom. The NASTRAN Theoretical Manual (section 9.3.2) shows how the extra points may be used to represent transfer functions whose expression in the frequency domain is

$$e_2 = H_{12}(p)e_1 \tag{32}$$

where

$$H_{12} = \frac{a_0 + a_1 p + a_2 p^2}{b_0 + b_1 p + b_2 p^2} \tag{33}$$

The technique is to multiply both sides of Eq. (31) by the denominator of Eq. (33) with the result

$$(b_o + b_1p + b_2p^2)e_2 = (a_o + a_1p + a_2p^2)e_1 \qquad (34)$$

The proper interpretation of Eq. (34) in the time domain is simply that p = d/dt. Equation (34) is then in the basic form, Eq. (31). Higher order polynomial quotients can be treated by factoring or by partial fraction expansion.

Aerodynamic theories do not usually express aerodynamic transfer functions as polynomial quotients, but polynomial quotients have frequently been used in approximate calculations with considerable success, particularly in the case of strip theory (Refs. 3, 4 and 5). They can also be applied, without approximation, to piston theory and to Newtonian flow theory. For lifting surface theories, on the other hand, it is difficult to find sufficiently simple approximations with adequate accuracy. Another difficulty is that different approximations have different matrix formulations with the result that the unity of form that applies to the different aerodynamic theories when they are expressed in the frequency domain is lost. For these reasons the matter of incorporating time-domain aerodynamic formulations into NASTRAN will not be pursued further at the present time.

REFERENCES

1. Cooley, J. W. and Tukey, J. W., "An Algorithm for the Machine Calculation of Complex Fourier Series," Math. Comp. Vol. 19, pp 297-301, Apr. 1965.

2. Brigham, E. O. and Morrow, R. E., "The Fast Fourier Transform," IEEE Spectrum, Dec. 1967.

3. MacNeal, R. H., Electric Circuit Analogies for Elastic Structures, J. Wiley & Sons, 1962, pp 200-204.

4. Wilts, C. H., "Aerodynamic Forces in Analog Computation," California Institute of Technology Computing Center Tech Report No. 116, Sept. 1959.

5. Rodden, W. P. and Stahl, B., "A Strip Method for Prediction of Damping in Subsonic Wind Tunnel and Flight Flutter Tests," J. Aircraft, Vol. 6, No. 1, Jan-Feb 1969, pp 9-17.

APPENDIX E

APPENDIX E

Interpolation by Means of Elastically Connected Beams and Plates

Interpolation will be used for three purposes in the NASTRAN Aeroelastic program:

1. To find the equation of constraint between the aerodynamic degrees of freedom and the structural grid point deflections.

2. To interpolate aerodynamic matrices to new Mach numbers and reduced frequencies.

3. To find an analytic representation for a frequency response curve, so that the Inverse Fourier Transform can be done with quadratures.

Linear splines for one dimensional problems, and grids of linear splines for two dimensional problems (Reference 1) have been used successfully for structural interpolation. This has been expanded for the NASTRAN implementation to include surface splines (infinite uniform plate) and attachment springs (a method of achieving smoothing).

E.1 Surface Splines

A surface spline is a mathematical tool used to find a function W (X, Y) for all points (X, Y) when W is known for a discrete set of points $W_i = W(X_i, Y_i)$. A linear spline is a "beam" function which passes through the known points. The natural extension to two dimensions is to introduce an infinite plate, and solve for its deflection, given its deflection at a discrete set of points. This surface spline is a smooth continuous function which will become nearly linear in X and Y at large distances from the points (X_i, Y_i). Furthermore, the problem can be solved in closed form involving nothing more difficult than to evaluate some logarithm functions.

Solution to the plate equation

The deflection of the plate will be synthesized as the response due to a set of point loads on the infinite plate. The response due to a single load is called a fundamental solution. The fundamental solutions have polar symmetry. If we take the load at $X = Y = 0$, and use polar coordinates ($X = r\cos\theta$, $Y = r\sin\theta$), the governing differential equation is

$$D\,\nabla^4 W = D\,\frac{1}{r}\,\frac{d}{dr}\left\{ r\,\frac{d}{dr}\left[\frac{1}{r}\,\frac{d}{dr}\left(r\,\frac{dw}{dr}\right)\right]\right\} = q \quad . \tag{1}$$

The load q vanishes except near $r = 0$. The general solution to the homogenous form of Eq. (1) is

$$W = C_0 + C_1\,r^2 + C_2\,\ln r + C_3\,r^2\,\ln r. \tag{2}$$

Set $C_2 = 0$, to keep the solution finite as $r \to 0$. Multiply Eq. (1) by $2\pi r$ and integrate from $r = 0$ to $r = \varepsilon$ (a small number). Thus

$$\lim_{r\to 0}\; 2\pi r\,D\,\frac{d}{dr}\left[\frac{1}{r}\,\frac{d}{dr}\left(r\,\frac{dw}{dr}\right)\right] = P \quad , \tag{3}$$

where P is a concentrated force. Applying Eq. (3) to Eq. (2) we get $C_3 = P/8\pi D$. Rearranging Eq. (3) we get the fundamental solution:

$$W = A + Br^2 + \frac{P}{16\pi D} r^2 \ln r^2 \quad . \tag{4}$$

The fundamental solutions are superimposed to solve the entire plate problem with a solution of the form

$$W(X,Y) = \sum_{i=1}^{N} \left[A_i + B_i\, r_i^2 + \frac{P_i}{16\pi D} r_i^2 \ln r_i^2 \right] \quad , \tag{5}$$

where $r_i^2 = (X-X_i)^2 + (Y-Y_i)^2$.

The remaining requirement is to satisfy the boundary condition at infinity. To do this we expand the solution for $X^2 + Y^2 \to \infty$.

$$r_i^2 = (X^2+Y^2) - 2(X_iX+Y_iY) + (X_i^2+Y_i^2) \tag{6}$$

$$\ln r_i^2 = \ln(X^2+Y^2) - 2\frac{X_iX+Y_iY}{X^2+Y^2} + \mathcal{O}(X^2+Y^2)^{-1} \tag{7}$$

$$r_i^2 \ln r_i^2 = \left\{ \begin{array}{l} (X^2+Y^2)\ln(X^2+Y^2) - 2(X_iX+Y_iY)\ln(X^2+Y^2) - 2(X_iX+Y_iY) \\ + (X_i^2+Y_i^2)\ln(X^2+Y^2) + \mathcal{O}(1) \end{array} \right\} \tag{8}$$

In Eqs. (7) and (8), $\mathcal{O}\cdot(X^2+Y^2)^n$ means of order $(X^2+Y^2)^n$. Put expansions (6) and (8) into Eq. (5) to get

$$W(X,Y) = \left\{ \begin{array}{l} \frac{1}{16\pi D} (X^2+Y^2)\ln(X^2+Y^2) \sum P_i + (X^2+Y^2) \sum B_i \\ - \frac{X\ln(X^2+Y^2)}{8\pi D} \sum X_i P_i - \frac{Y\ln(X^2+Y^2)}{8\pi D} \sum Y_i P_i \\ + 2X \sum\left(\frac{X_i P_i}{16\pi D} - X_i B_i\right) + 2Y \sum\left(\frac{Y_i P_i}{16\pi D} - Y_i B_i\right) \\ + \frac{\ln(X^2+Y^2)}{16\pi D} \sum (X_i^2+Y_i^2) P_i + \mathcal{O}(1) \end{array} \right\} \tag{9}$$

In order to satisfy boundary conditions at infinity, we must get rid of all terms in $(X^2 + Y^2)\ln(X^2 + Y^2)$, $(X^2 + Y^2)$, $X\ln(X^2 + Y^2)$ and $Y\ln(X^2 + Y^2)$, leaving terms of order X, Y, $\ln(X^2+Y^2)$, and 1. Thus

$$\sum B_i = 0 \tag{10}$$

$$\sum P_i = 0 \tag{11}$$

$$\sum X_i P_i = 0 \tag{12}$$

$$\sum Y_i P_i = 0 \tag{13}$$

Equations (11) thru (13) are recognized as the equations of equilibrium. Using Eqs. (6) and (10) it is seen that

$$\sum_{i=1}^{N} (A_i + B_i r_i^2) = a_0 + a_1 X + a_2 Y \quad . \tag{14}$$

Thus one form of the solution for a two dimensional spline is, using Eq. (5),

$$W(X,Y) = a_0 + a_1X + a_2Y + \sum_{n=1}^{N} K_i(X,Y) \quad P_i \qquad (15)$$

$$\text{where } K_i(X,Y) = \frac{1}{16\pi D} r_i^2 \ln r_i^2 \ , \ r_i^2 = (X-X_i)^2 + (Y-Y_i)^2$$

The N + 3 unknowns (a_0; a_1; a_2; P_i; i=1,N) are determined from the N + 3 equations

$$\sum P_i = \sum X_iP_i = \sum Y_iP_i = 0, \quad \text{and}$$

$$W_j = a_0+a_1X_j + a_2Y_j + \sum_{n=1}^{N} K_{ij} \ P_i \qquad (j=1,N) \qquad (16)$$

where $K_{ij} = K_i(X_j, Y_j)$.

Note that $K_{ij} = K_{ji}$, and $K_{ij} = 0$ when $i = j$.

These equations can be summarized in matrix form:

$$W(X,Y) = \lfloor 1, X, Y \ \vdots \ K_1(X,Y), K_2(X,Y), \cdots, K_N(X,Y) \rfloor \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ \hline P_1 \\ P_2 \\ \vdots \\ P_N \end{Bmatrix} \qquad (17)$$

where $K_i(X,Y)$ is defined below Eq. (15)

The vector of a's and p's is found by solving

$$\begin{Bmatrix} 0 \\ 0 \\ 0 \\ W_1 \\ W_2 \\ \vdots \\ W_N \end{Bmatrix} = \left[\begin{array}{ccc|ccccc} 0 & 0 & 0 & 1 & 1 & \cdots & & 1 \\ 0 & 0 & 0 & X_1 & X_2 & \cdots & & X_N \\ 0 & 0 & 0 & Y_1 & Y_2 & \cdots & & Y_N \\ \hline 1 & X_1 & Y_1 & 0 & K_{12} & \cdots & & K_{1N} \\ 1 & X_2 & Y_2 & K_{21} & 0 & \cdots & & K_{2N} \\ \vdots & \vdots & \vdots & \vdots & \vdots & & & \vdots \\ 1 & X_N & Y_N & K_{N1} & K_{N2} & \cdots & & 0 \end{array}\right] \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ \hline P_1 \\ P_2 \\ \vdots \\ P_N \end{Bmatrix} \tag{18}$$

where K_{ij} is defined below Eq. (16).

The interpolation to any point in the plane (X,Y) is then achieved by evaluating W(X,Y) from Eq. (17).

E.2 Linear Splines, an Alternative Derivation.

Linear splines are easily solved by the three-moment method (see Section 5.7 for the details). This method is excellent for simple linear splines. Unfortunately, the method does not work as well for splines with torsion, rigid arms, and attachment springs as described in Section 5.4 for geometry interpolation. The derivations sketched below are based upon an analogy with the surface spline derivation.

a. Linear splines

Equation:

$$EI \frac{d^4w}{dx^4} = q - \frac{dm}{dx} \tag{19}$$

where q = applied load and m = applied moment

A symmetric fundamental solution for $X \neq 0$ is used for loads $q = P\delta(X)$.

$$W = C_0 + C_1|X| + C_2 X^2 + C_3|X|^3 \tag{20-s}$$

Continuity of slope requires $C_1 = 0$, and the equilibrium condition gives $C_3 = P/12EI$. An antisymmetric solution is used for applied moments. Thus, for $X \neq 0$

$$W = d_1 X + d_2 X|X| + d_3 X^3 \tag{20-a}$$

The equilibrium condition gives $d_2 = -\frac{M}{4EI}$. Combining fundamental solutions, and renaming the coefficients

$$W = A + BX + CX^2 + DX^3 - \frac{M}{4EI} X|X| + \frac{P}{12EI} |X|^3 \tag{21}$$

Superposition for loads at $X = X_i$ gives

$$W(X) = \sum_{i=1}^{N} [A_i+B_i(X-X_i)+C_i(X-X_i)^2+D_i(X-X_i)^3 - \frac{M_i(X-X_i)|X-X_i|}{4EI} + \frac{P_i|X-X_i|^3}{12EI} \tag{22}$$

As $X \to \pm\infty$,

$$W(X) = X^3 \sum \left(D_i \pm \frac{P_i}{12EI}\right) + X^2 \sum \left(C_i - 3D_iX_i \mp \frac{M_i+P_iX_i}{4EI}\right) + \mathcal{O}(X) \tag{23}$$

Thus, to satisfy the condition at infinity, which is that the slope approaches a constant,

$$\sum D_i = \sum C_i = \sum D_i X_i = 0 \tag{24}$$

$$\sum P_i = \sum (M_i + P_i X_i) = 0 \tag{25}$$

Equation (25) is recognized as the equilibrium condition. The solution is given by (using Eqs. (22) and (24))

$$W(X) = a_0 + a_1 X + \sum_{i=1}^{N} \left(- \frac{M_i (X-X_i)|X-X_i|}{4EI} + \frac{P_i |X-X_i|^3}{2EI} \right) \quad (26)$$

$$\theta(X) = \frac{dw}{dx} = a_1 + \sum_{i=1}^{N} \left(- \frac{M_i |X-X_i|}{2EI} + \frac{P(X-X_i)|X-X_i|}{4EI} \right) \quad (27)$$

These are written in matrix notation as

$$\begin{Bmatrix} W(X) \\ \hline \theta(X) \end{Bmatrix} = \left[\begin{array}{cc:cc:cc} 1 & X & \frac{|X-X_1|}{12EI} & \cdots & -\frac{(X-X_1)|X-X_1|}{4EI} & \cdots \\ \hdashline 0 & 1 & \frac{(X-X_1)|X-X_1|}{4EI} & \cdots & -\frac{|X-X_1|}{2EI} & \cdots \end{array} \right] \begin{Bmatrix} a_0 \\ a_1 \\ \hdashline P_1 \\ \vdots \\ P_N \\ \hdashline M_1 \\ \vdots \\ M_N \end{Bmatrix} \quad (28)$$

The unknowns a, P and M are found from

$$\begin{Bmatrix} 0 \\ 0 \\ \hdashline W_1 \\ \vdots \\ W_N \\ \hdashline \theta_1 \\ \vdots \\ \theta_N \end{Bmatrix} = \left[\begin{array}{c:c:c} 0 & R_1^T & R_2^T \\ \hdashline R_1 & A_{11} & A_{21}^T \\ \hdashline R_2 & A_{21} & A_{22} \end{array} \right] \begin{Bmatrix} a_0 \\ a_1 \\ \hdashline P_1 \\ P_2 \\ \vdots \\ P_N \\ \hdashline M_1 \\ M_2 \\ \vdots \\ M_N \end{Bmatrix} \quad (29)$$

where it has been assumed $X_1 < X_2 \cdot \cdot \cdot < X_N$, and

$$R_1^T = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ X_1 & X_2 & \cdots & X_N \end{bmatrix}$$

$$R_2^T = \begin{bmatrix} 0 & 0 & \cdots & 0 \\ 1 & 1 & \cdots & 1 \end{bmatrix}$$

$$A_{11} = \begin{bmatrix} 0 & \frac{(X_2-X_1)^3}{12EI} & \cdots & \frac{(X_N-X_1)^3}{12EI} \\ \frac{(X_2-X_1)^3}{12EI} & 0 & \cdots & \frac{(X_N-X_2)^3}{12EI} \\ \vdots & \vdots & & \vdots \\ \frac{(X_N-X_1)^3}{12EI} & \frac{(X_N-X_2)^3}{12EI} & \cdots & 0 \end{bmatrix}$$

$$A_{21} = \begin{bmatrix} 0 & -\frac{(X_2-X_1)^2}{4EI} & \cdots & -\frac{(X_N-X_1)^2}{4EI} \\ \frac{(X_2-X_1)^2}{4EI} & 0 & \cdots & -\frac{(X_N-X_2)^2}{4EI} \\ \vdots & \vdots & & \vdots \\ \frac{(X_N-X_1)^2}{4EI} & \frac{(X_N-X_2)^2}{4EI} & \cdots & 0 \end{bmatrix}$$

$$A_{22} = \begin{bmatrix} 0 & -\frac{(X_2-X_1)}{2EI} & \cdots & -\frac{(X_N-X_1)}{2EI} \\ -\frac{(X_2-X_1)}{2EI} & 0 & \cdots & -\frac{(X_N-X_2)}{2EI} \\ \vdots & \vdots & & \vdots \\ -\frac{(X_N-X_1)}{2EI} & -\frac{(X_N-X_2)}{2EI} & \cdots & 0 \end{bmatrix}$$

b. Tension and Torsion Bars.

An analogous derivation exists for torsion bars (and tension bars), which is sketched below:

Equation :

$$GJ \frac{d^2\theta}{dx^2} = -T \tag{30}$$

A fundamental solution for $x \neq 0$

$$\theta = C_0 + C_1|X| \tag{31}$$

Apply the jump condition to solve for C_1, and superimpose the solutions to get

$$\theta\ (X) = \sum_{i=1}^{N} \left[C_i - \frac{T_i|X-X_i|}{2GJ} \right] \tag{32}$$

To satisfy that θ = constant for $X = \pm\infty$

$$\sum T_i = 0 \tag{33}$$

The solution is

$$\theta\ (X) = \left[1 \;\vdots\; 0\ ,\ -\frac{|X_2-X_1|}{2GJ}\ ,\ \ldots\ ,\ -\frac{|X_N-X_1|}{2GJ} \right] \begin{Bmatrix} a_0 \\ \hline T_1 \\ T_2 \\ \vdots \\ T_N \end{Bmatrix} \tag{34}$$

where the unknowns come from

$$\begin{Bmatrix} 0 \\ \hline \theta_1 \\ \theta_2 \\ \vdots \\ \theta_N \end{Bmatrix} = \left[\begin{array}{c|cccc} 0 & 1 & 1 & \cdots & 1 \\ \hline 1 & 0 & -\frac{|x_2-x_1|}{2GJ} & \cdots & -\frac{|x_N-x_1|}{2GJ} \\ 1 & -\frac{|x_2-x_1|}{2GJ} & 0 & & -\frac{|x_N-x_2|}{2GJ} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & -\frac{|x_N-x_1|}{2GJ} & -\frac{|x_N-x_2|}{2GJ} & \cdots & 0 \end{array}\right] \begin{Bmatrix} a_0 \\ \hline T_1 \\ T_2 \\ \vdots \\ T_N \end{Bmatrix} \tag{35}$$

For tension bars, simply replace

$$\theta \rightarrow U$$
$$GJ \rightarrow AE$$
$$T \rightarrow F$$

in Eqs. (30) thru (35).

E.3 Attachment of splines with elastic springs.

A refinement of the spline interpolation analogy with structural mechanics is to attach the splines to the structural grid points with springs. The purpose of the springs is to provide a measure of smoothing to the data. The question naturally arises as to how stiff to make the springs. If the springs are infinitely stiff, the plate will pass through all of the data points. If the springs have infinitesimal stiffness, the plate will pass through a least-squares linear fit of the data points.

The question can be analyzed for the linear spline interpolation by means of a spring-connected beam. Displacement of the beam satisfies the equation

$$EI \frac{d^4w}{dx^4} = \sum_n (u_n - w) K_n \delta(x - x_n) \qquad (36)$$

where $u_n(x_n)$ is one of the data points, K_n is the spring constant, and $\delta(x-x_n)$ is a delta function such that

$$\int_{x_n-\varepsilon}^{x_n+\varepsilon} \delta(x - x_n)dx = 1 \qquad \text{for all } \varepsilon > 0 \qquad (37)$$

It is convenient to assume that the data points are uniformly, closely spaced so that, passing to the limit as the spacing, Δx, approaches zero,

$$EI \frac{d^4w}{dx^4} = K'(u - w) \qquad (38)$$

where $K' = \frac{K_n}{\Delta x}$. Let us examine the case where the beam is infinitely long and u is zero everywhere except near the origin where $u(0) = A\delta(x)$. The solution is

$$w = \frac{A}{2\ell} e^{-\frac{x}{\ell}} \left[\cos\left(\frac{x}{\ell}\right) + \sin\left(\frac{x}{\ell}\right)\right] \qquad (39)$$

where

$$\ell = \sqrt{2} \left(\frac{EI}{K'}\right)^{1/4} \qquad (40)$$

ℓ is, evidently, related to the "effective" length over which the value of a single data point has an important influence on the smoothed curve, w. (For example, when $x = \ell$, $w = 0.51\ A/2\ell$.) ℓ can be selected by the user on physical grounds or according to some intuitive concept of smoothness.

As a practical matter, the values of the spring constants, K_n, are required to perform the interpolation. It is suggested that they be determined as follows.

1. The user selects a smoothing length, "ℓ".
2. K'/EI is computed from Eq. (40):

$$K'/EI = \frac{4}{\ell^4} \tag{41}$$

3. Δx is assumed equal for all data points (in order to give them equal weights) and is computed from $\Delta x = L/N$ where L is the difference between the largest and smallest values of x_n and N is the number of data points.
4. The spring constant is, therefore

$$K_n = 4(L/N)\ EI/\ell^4 \tag{42}$$

The analysis of the two-dimensional plate problem leads to similar results except that the trignometric function in the solution is replaced by a Bessel function. The effective length, obtained from analogy with Eq. (40), is

$$\ell = \sqrt{2}\left(\frac{D}{K''}\right)^{1/4} \tag{43}$$

where K'' is the spring constant per unit area and $D = 1/12\ Et^3$ is the bending stiffness of the plate. It is suggested that the spring constant be evaluated as follows:

1. The user selects a smoothing length, ℓ.
2. Equation (43) is used to calculate

$$\frac{K''}{D} = \frac{4}{\ell^4} \tag{44}$$

3. The spring constants for data points are assumed equal, and equal to

$$K_n = \Delta SK'' = \frac{\pi R^2}{N} \frac{4D}{\ell^4} \tag{45}$$

where R is the mean distance from the center of gravity of the data points to the data points, $R = \left[\left(\sum r_n{}^2\right) / N\right]^{1/2}$, and N is the number of data points.

The change in the formulas for splines to accommodate the springs is very easy. A derivation, valid for the several types of splines, is as follows.

The spline deflection is given by Eq. (17), (28) or (34) and can be written

$$u_k^{(r)} = \lfloor R(r) \rfloor \{a\} + \lfloor A_j(r) \rfloor \{P\} \tag{46}$$

where u_k is the deflection of the spline and the r may be a one or two dimensional argument. Thus, including the equilibrium equations (18), (29) or (35):

$$0 = [R_i]^T \{P\} \tag{47}$$

and

$$\{u_k\} = [R_i] \{a\} + [A_{ij}] \{P\} \tag{48}$$

The structure deflection u_g will differ from the spline deflection by the deformation of the spring, resulting in forces

$$\{P\} = [K_s] \{u_g - u_k\} \tag{49}$$

where the matrix, K_s, has the spring constants, K, along the diagonal. These are non-zero (if K were = 0, then there would be no attachment and we would discard that grid point) and thus the inverse of K_s is:

$$[K_s]^{-1} = \begin{bmatrix} \frac{1}{K} \cdot & & 0 \\ & \cdot & \\ 0 & & \cdot \frac{1}{K} \end{bmatrix} \tag{50}$$

Eliminate u_k between Eqs. (48) and (49) to get

$$\{u_g\} = [R_i]\ \{a\} + \left([A_{ij}] + [K_s]^{-1}\right)\{P\} \tag{51}$$

Thus all that is required to accommodate springs is to add the spring flexibilities to the diagonal of the spline influence coefficient matrix. This is obvious by physical reasoning, since the spring and spline flexibilities are in series and can be added directly.

E.4 Rigid arms on linear splines.

The linear splines used for geometry interpolation have rigid arms (see Figure 5.4-2). Mathematically, these represent equations of constraint between the displacements and rotations at the spline end and attachment end of the spline. The constraint equations are used to transform the influence functions from the spline ends to influence functions at the attachment ends. The complete transformed influence functions are shown in Table 5.4-1.

REFERENCES

1. Done, G.T.S., Interpolation of Mode Shapes: A Matrix Scheme Using Two-Way Spline Curves. Aeronautical Quarterly, Vol. XVI, Part 4, pp. 333-349, Nov. 1965.

APPENDIX F

APPENDIX F

Procedures in Flutter Analysis

The most general form of eigenvalue problem solved by NASTRAN has the form

$$[Mp^2 + Bp + K]\{u\} = 0 \qquad (1)$$

The solution vector {u} may either be physical displacements or modal coordinates. In the conventional approach to flutter analysis, the aerodynamic properties are included in the mass matrix which then has the general form

$$[M] = [M^1 + M^2 + M^a] \qquad (2)$$

where $[M^1]$ is the structural mass matrix, $[M^2]$ is the matrix of direct input terms prescribed by the user and $[M^a]$ is the aerodynamic mass matrix. The mass matrix may be written as

$$[M^a] = \frac{\rho b^2}{2k^2}[Q] \qquad (3)$$

where the matrix [Q] is a relatively slowly varying function of Mach number, m, and of reduced frequency, k.

An alternative formulation of the flutter problem is obtained by treating the aerodynamic effect by means of an equivalent stiffness matrix

$$[K^a] = -\frac{1}{2}\rho V^2[Q] \qquad (4)$$

which is added to [K] in Eq. (1). This formulation has the apparent disadvantage that the velocity, V, is related to reduced frequency by $V = b\omega/k$ so that $q = \frac{1}{2}\rho V^2$ cannot be specified as a fixed parameter in solving the eigenvalue problem. The actual situation is not, however, greatly different for the two formulations. The "independent" parameters in the mass formulation are k, m, and ρ, while those in the stiffness formulation are k, m, and q. Neither set of parameters is truly independent because at the flutter condition, the flutter frequency

$$\omega_f = \frac{kV}{b} = \frac{kma}{b} \tag{5}$$

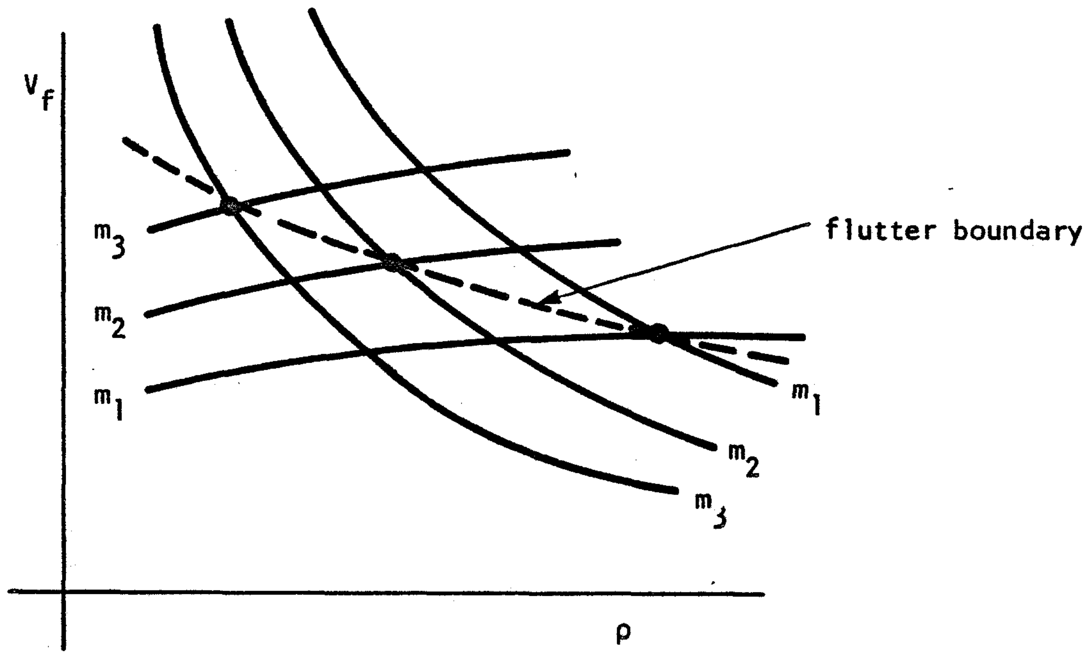
where a is the velocity of sound. Only if [Q] is independent of Mach number do k and ρ become independent parameters for the mass formulation while k, m, and q never yield an independent pair for the stiffness formulation. Historically, flutter analysis began with low subsonic speeds where Mach number is insignificant which is why the mass formulation is now standard.

Whichever formulation is used, the determination of true flutter boundaries requires cross-plotting of the results for combinations of the independent parameters. In the case of the mass formulation the flutter speed satisfies the pair of equations

$$\left.\begin{aligned} V_f &= \frac{b\omega_f}{k_f} = V_f(\rho,m) \\ \text{and} \qquad & \\ V_f &= ma(\rho) \end{aligned}\right\} \tag{6}$$

The dependence of a on ρ may be obtained from a standard atmosphere table.

The two equations in Eq. (6) may be cross-plotted versus density as follows:

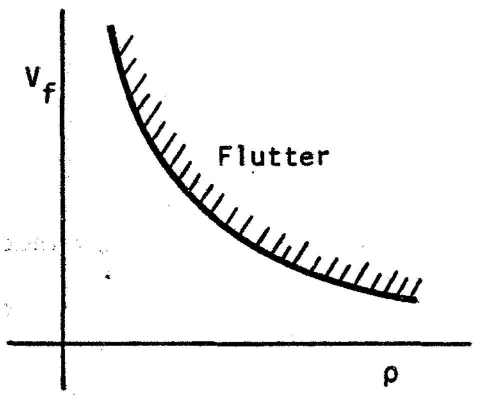


In the case of the stiffness formulation, the flutter speed satisfies the same pair of equations which, however, are now functions of Mach number and dynamic pressure

$$\left.\begin{aligned} V_f &= \frac{b\omega_f}{k_f} = V_f(q,m) \\ V_f &= m\,a(q,m) \end{aligned}\right\} \qquad (7)$$

which may be cross-plotted versus dynamic pressure to obtain a flutter boundary.

Since both formulations lead to feasible solutions, the selection of one or the other depends partly on computational efficiency and partly on precedent. The mass formulation has a large computational advantage when the aerodynamic matrix [Q] is assumed independent of Mach number and it also has the advantage of precedent. The stiffness formulation has a significant advantage for some methods of eigenvalue extraction such as the p-k method, Ref. 1. In the p-k method the reduced frequency is eliminated as an independent parameter during the process of iteration toward particular roots of the flutter problem. Thus, if V is specified as a parameter and ω_i is the current estimate of frequency, an estimate of the reduced frequency can be found from $k_{i+1} = b\omega_i/V$. With this method, V and ρ are specified as independent parameters, the Mach number is found from $m = V/a$, and the flutter boundary is obtained directly by plotting flutter speeds vs density.



In cases where the aerodynamic matrix [Q] is a function of Mach number, the p-k method requires one less independent parameter than the standard formulation (k method). It pays for this advantage by requiring a much less efficient method of eigenvalue extraction. With the k-method highly efficient transformation techniques (see Appendix H) can be used. Reference 1 recommends determinant iteration for the p-k method. We propose instead a modified form of the shifted power method described in Appendix I, which should be more efficient. Both the determinant and power methods rely on approximate knowledge of the eigenvalues based on prior solutions.

Figure 1 shows a proposed flow diagram for flutter analysis by both the k and p-k methods, detailing the decision points. For both methods, all instructions and data for the inner decision loop should be core held. It will be noted that cross-plotting of flutter speeds vs ρ and m is not done automatically. In our opinion the analyst can do a much better job, particularly if the number of data points is small. Since flutter analysis is expensive, excessive numbers of data points should be avoided.

Both methods shown in Figure 1 require some modification of the eigenvalue routines used in NASTRAN. The modifications for the p-k method are discussed in Appendix I. For the k method a transformation method that will handle complex unsymmetric matrices, such as the one described in Appendix H, should be added to provide improved efficiency for the extraction of eigenvalues from full matrices. It is assumed, in this connection, that structural modes will usually be used as degrees of freedom, with the result that aerodynamic matrix will be of relatively low order, but full.

It is seen in Figure 1 that the output data includes both ρ and m as parameters, and it has been shown in the previous discussion that the user is required to cross-plot the results to obtain flutter speeds. It is, however, possible for the user to apply the density loop in the k method in order to minimize the cost of flutter calculations, as will now be demonstrated.

Wind tunnel data are frequently summarized in terms of an altitude-stiffness parameter as a function of Mach number. One form of a stiffness-altitude parameter is $V/b\omega\sqrt{\mu}$ where μ is the mass ratio, $\mu = M_s/\pi\rho b^2 s$, in

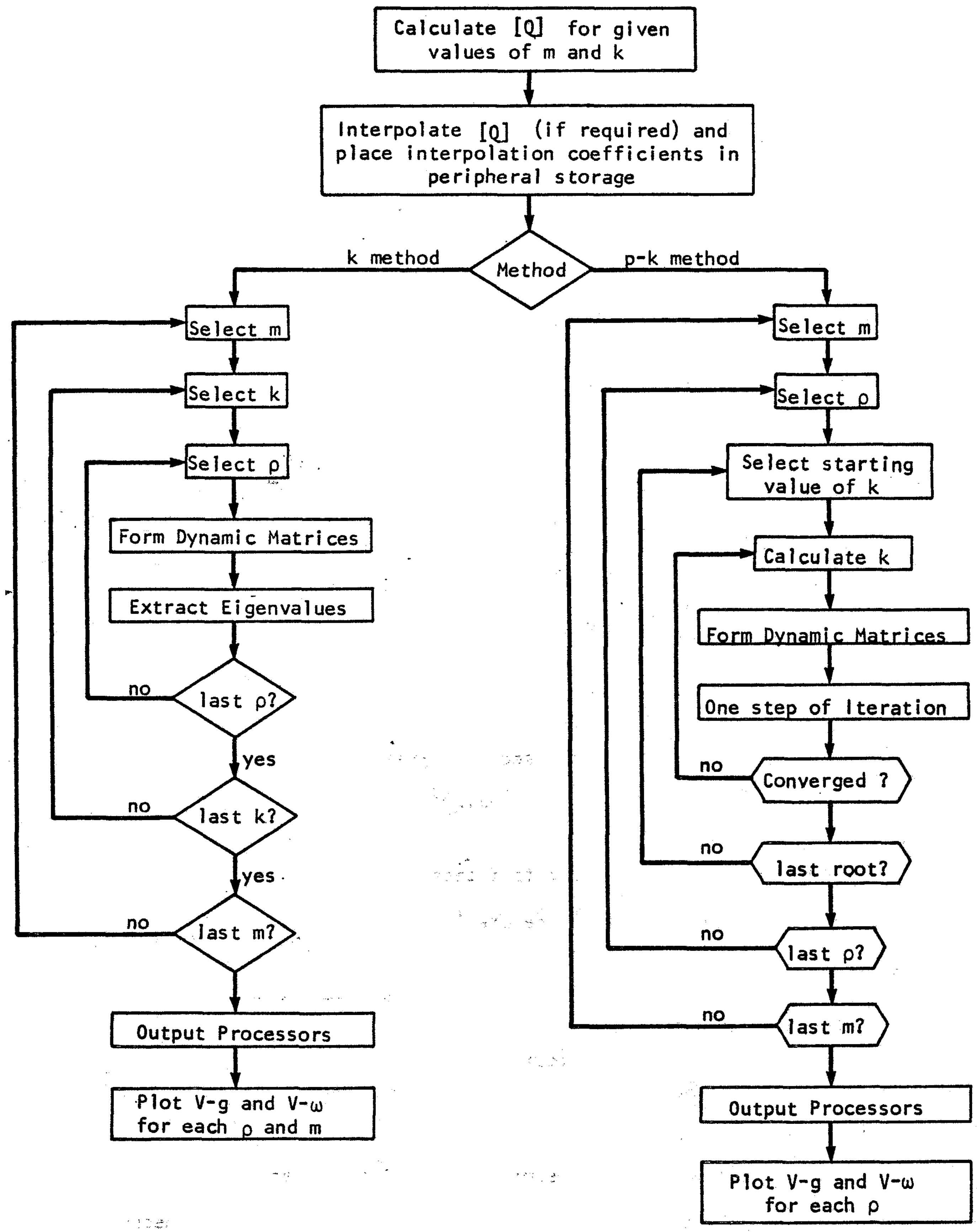


FIG. 1. Basic Flow Diagram for Flutter Analysis

which $\pi \rho b^2 s$ represents the mass of air in the conical frustum circumscribing the wing and M_s is the structural mass of the wing. The stability curve appears as in Figure 2.

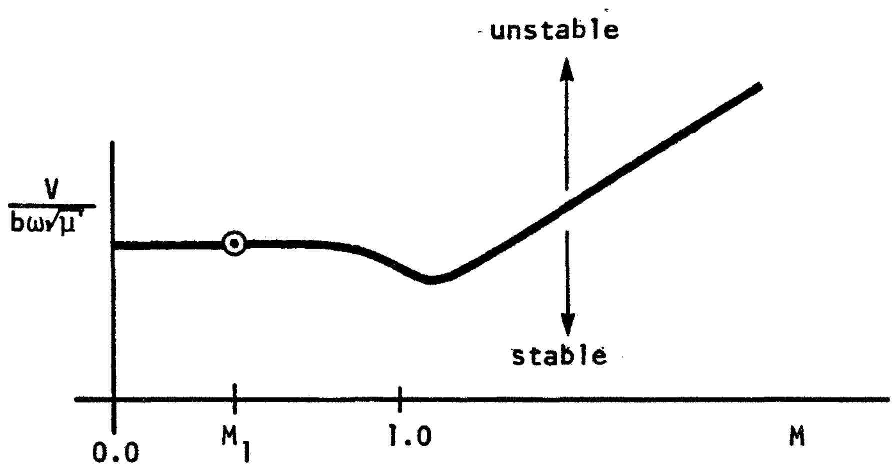


FIG. 2. Stiffness-Altitude Stability Parameter

The stability curve can be generated analytically for each Mach number M_i by choosing a <u>single</u> representative reduced freuqency k_i and obtaining the corresponding aerodynamic influence coefficients (AIC's). With this AIC matrix and a series of densities, ρ_i, the eigenvalue solutions will lead to a g-ρ curve and an ω-ρ curve as shown in Figure 3.

The flutter point(s) correspond(s) to the density(ies) at which $g = 0$. From this density, say ρ_1, the mass ratio at flutter, μ_1, is known and the stiffness-altitude parameter, $V/b\omega\sqrt{\mu} = 1/k_1\sqrt{\mu_1}$ is known for the Mach number selected, say M_i. This gives an analytically derived point as shown in Figure 2. The entire curve in Figure 2 is obtained by repeating the procedure at additional Mach numbers.

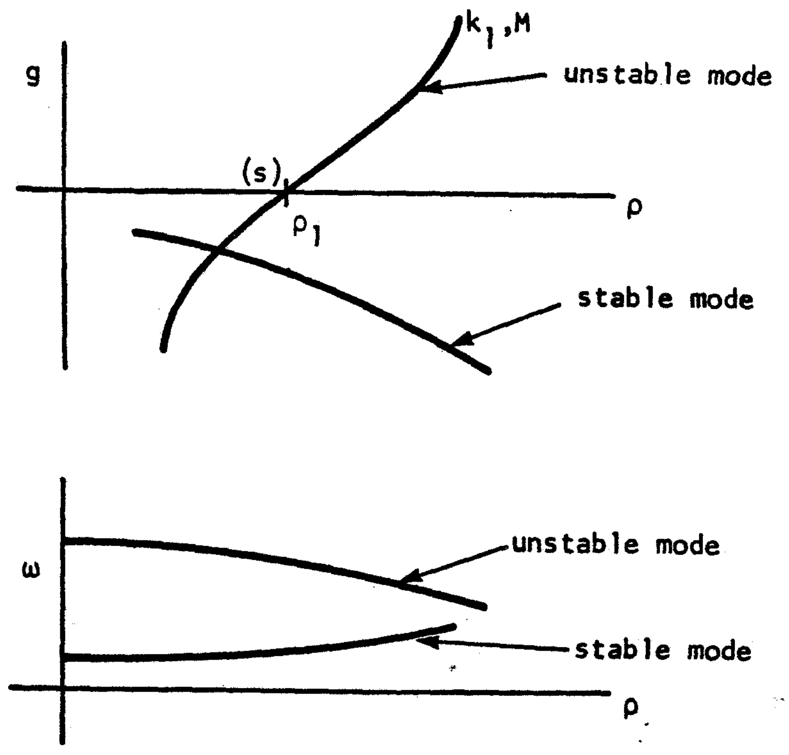


FIG. 3 Variable Density Damping and Frequency Curves

For applications to a specific altitude corresponding to mass ratio μ_2, the choice of representative reduced frequency k_1 may not have been appropriate and an improved value may be estimated from $k_2 = k_1\sqrt{\mu_1/\mu_2}$. The curve of Fig. 2 may then be refined by repeating the above sequence of calculations to determine its sensitivity to the choice of representative reduced frequency. The fundamental assumption is that the stiffness-altitude parameter, which is proportional to $\sqrt{\rho}/k$, adequately describes the contributions of ρ and k to the flutter problem so that their individual effects need not be determined in every case. This is an oversimplifying assumption that has been discussed by Yates[3], Appendix C, so that conventional flutter methods must eventually be employed. Once a representative reduced frequency is obtained, say k_3, the g-V curve can be generated in the region near k_3 by either the k or the p-k method for constant altitude (density) to determine the flutter speed accurately at that altitude.

The possibility of substantial savings in aerodynamic analysis suggests this approach is deserving of serious attention. Its primary value can be seen in preliminary design work in which numerous planforms are under consideration.

REFERENCES

1. H.J. Hassig, "Determinant Iteration Applied to the P-K Method of Solving the Flutter Equation". AFDC Meeting, Dallas, Texas, Dec. 2 - 4, 1970.

2. J. H. Wilkinson, THE ALGEBRAIC EIGENVALUE PROBLEM, Chapter 6, Clarendon Press, Oxford, 1965.

3. E. C. Yates, Jr., "Subsonic and Supersonic Flutter Analysis of a Highly Tapered Swept-Wing Planform, Including Effects of Density Variation and Finite Wing Thickness, and Comparison with Experiments," TN D-4230, November, 1967, NASA.

APPENDIX G

APPENDIX G

A Comparison of the Characteristics of Candidate Unsteady Aerodynamic Theories

Numerous candidate aerodynamic theories for lifting surfaces in subsonic to hypersonic flight regimes were discussed and compared in Ref. 1. We are now able to review the leading candidates in each flight regime in the light of the general capabilities desired in NASTRAN, as discussed in the Introduction. The desired general capabilities include versatility with respect to permissible aerodynamic configurations, accuracy, efficiency both in terms of user conveniences and in terms of computational procedures, and compatibility with existing NASTRAN capabilities.

We regard the subsonic Doublet-Lattice Method as a technique that is typical of aerodynamic interference methods, but also as one that satisfies all of the above noted desirable features. Specifically,

a. It accommodates multiple aerodynamic surfaces with arbitrary planform shapes. The major restrictions are that the surfaces should be (nearly) parallel to the undisturbed flow.

b. It converges to theoretically correct results, within the limits of small perturbation theory, for the entire reduced frequency ranges and for $0 \leq M < 1$ as the number of aerodynamic elements is increased. No limit is placed on the number of elements.

c. It requires a minimum effort from the user to describe aerodynamic elements and to provide other required input data.

d. It is a reasonably efficient computational scheme, although not so efficient as those methods (such as the kernel function methods) that produce aerodynamic influence coefficients from a priori assumptions regarding the pressure distribution. The speed of modern computers tends to minimize the importance of this feature.

e. It is generally compatible with the finite element concept of structural analysis. The geometric and matrix properties of its finite aerodynamic elements are easily related to the geometric and matrix properties of the structural model.

The possibility of a Sonic/Supersonic-Doublet-Lattice Method also exists but it is a future development whose success remains to be proved. The most advanced proven methods available for speeds above subsonic include the Transonic Sonic Box Method of Stenton and Andrew[2], the Supersonic Element Method of Kariappa and Smith[3], and, of course, Piston Theory[4] for high supersonic and hypersonic speeds.

The specific requirements for implementing the Doublet-Lattice Method in NASTRAN are discussed in Section 5.5 of this report. This appendix discusses the additional requirements for implementing sonic, supersonic, and hypersonic capability. We discuss the supersonic case first because it has been more refined than the sonic development, which should be regarded as a special case. A discussion of the subsonic and supersonic kernel function method is also presented after the discussion of the finite element methods.

G.1 Supersonic Finite Element Methods

The fundamental aerodynamic element of the subsonic Doublet-Lattice Method is a trapezoid with its parallel edges aligned with the free-stream.

The general development of supersonic methods has utilized a fundamental element known as the Mach Box, which is a rectangle with diagonals parallel to Mach lines. This element achieved a fair measure of success, in spite of its geometric incompatibility with typical planforms, in a method[5,6] that further limited its accuracy by global interpolation of downwashes and velocity potential functions. The recent development of Kariappa and Smith[3] chooses a triangle with one side parallel to the free-stream as its fundamental element and uses local interpolation for downwashes and velocity potentials. As a result, the method of Kariappa and Smith is compatible with arbitrary planforms and it has resulted in a significant improvement in accuracy and computational efficiency.

The subsonic Doublet-Lattice Method (and also the Kernel Function Method) is based on an acceleration (pressure) potential formulation of the aerodynamic lifting integral equation and it is therefore only concerned with potentials on the lifting surfaces. In contrast, the supersonic box methods have employed the velocity potential formulation and are concerned with potentials not only on the lifting surfaces but also off of them in the so-called diaphragm regions. A comprehensive description of diaphragm configurations for nonplanar interfering configurations has been given by Andrew and Moore[6,7].

The consideration of diaphragm regions in addition to lifting surfaces in NASTRAN would require additional input from the user, since the choice of diaphragm configurations is not unique. The diaphragm may be regarded as another kind of lifting surface, and its geometric features may be treated in the same manner as the actual lifting surfaces. However, the boundary conditions on the two kinds of lifting surfaces are different: the downwashes (normalwash) on the actual lifting surfaces are specified and the velocity potentials are unknown; the velocity potentials on the diaphragms are specified by the zero lifting pressure condition, and the downwashes are unknown.

A nonplanar version of the Kariappa-Smith supersonic method can be developed easily for inclusion in NASTRAN if the selection of diaphragm configurations is left to the user. For very complex interfering configurations this may place a very large burden on the user. If the selection of diaphragm configurations is to be automated, a severe programming requirement will exist that may limit the generality of configurations that can be accomodated. The logical complexity of the program of Andrew[6] for a limited number of empennage configurations is a case in point. A combination of computer-generated and user-specified diaphragm configurations may be a workable compromise with the generality desired in NASTRAN.

G.2 The <u>Transonic Box Method</u>

The Transonic Box Method is also a velocity potential formulation of the lifting problem and therefore requires auxilliary diaphragm regions as in the supersonic methods. The fundamental element employed by Stenton and Andrew[2] is square (the Mach box is infinite spanwise at $M = 1.0$) and again results in incompatibility with typical planforms. Its accuracy is also limited by global interpolation of downwashes and velocity potentials as in the supersonic Mach Box Methods[5,6].

Since the theory for transonic flow is only a limiting case of supersonic flow, the procedures employed by Kariappa and Smith[3] can be adapted to permit the Mach number to approach unity and the discussion of the supersonic problem in Section G.1 may be regarded as including the sonic case.

G.3 Kernel Function Methods

The modal solutions of the subsonic and supersonic lifting surface problems are known as kernel function methods. A series of pressure functions or modes is assumed and their amplitudes are determined by collocation at an optimum set of control points. The subsonic techniques are more advanced than the supersonic but they are still somewhat restricted as far as interfering and intersecting surfaces are concerned, particularly with regard to control surfaces. The first computational procedure was given by Watkins, Woolston, and Cunningham[8]. An extension to a nonplanar surface was given by Vivian and Andrew, a control surface was added by Berman et al.[9], wing-tail interference was considered by Albano et al.[10], and intersecting surfaces have been investigated by Andrew[11].

The supersonic case of a planar wing has been investigated by Cunningham[12,13] and by Curtis and Lingard[14]. No extensions have yet been made to include control surfaces or multiple surface interference.

A subsonic kernel function procedure could be developed that satisfies the general requirements of NASTRAN by combining the best features of Refs. 8 - 11. In this way versatility with respect to permissible aerodynamic configurations, efficiency in computation, user convenience, and compatibility with existing NASTRAN capabilities would be achieved. The question of accuracy would require further investigation because the techniques of Refs. 9 and 11 have been observed to have erratic convergence characteristics. However, it must be noted that the programming effort to permit a very large variety of aerodynamic configurations will be substantial.

It does not appear that the Supersonic Kernel Function Method has been investigated sufficiently that its development for NASTRAN is warranted at this time.

G.4 Piston Theory

While Piston Theory is obviously a finite element method, since a point relationship exists between pressure and downwash[4], it has never been automated as such. It has always been considered as a strip theory with a rigid chord[15], or with a parabolically cambering chord[16] and with an aerodynamically unbalanced control surface. In the format of the Doublet-Lattice Method, the pressure-downwash matrix for Piston Theory becomes a diagonal matrix. In its most extended form, third-order Piston Theory leads to the diagonal elements that relate the upper- and lower-surface pressures to the unsteady box downwash in terms of the Mach number and the upper- and lower-surface steady downwashes. The steady downwash at each surface depends on the local quasi-steady angle of attack and on the rate of change of airfoil thickness along the chord. The downwash at each box can be evaluated at its center, and the pressure difference can be assumed to act at the center; the steady downwash on each surface of a box can be evaluated using the average rate of change of thickness along the box chord.

Two modifications to the basic Piston Theory of Ashley and Zartarian[4] have been suggested by Rodden et al.[15] in order to extend the usefulness of Piston Theory to lower supersonic Mach numbers. One is a correction for sweep; the other is an adjustment of the pressure coefficients to agree with the low frequency, second order, two dimensional, supersonic airfoil theory of Van Dyke[17].

Two additional modifications should be considered in any implementation for NASTRAN because of the simplicity of the format of the aerodynamic influence coefficients. The first is to prevent the pressure-downwash relation from predicting a pressure lower than vacuum on one surface, e.g., the upper surface at a very high reentry angle of attack. This may be included by a simple test on the local pressure coefficient or by use of shock-expansion theory as has been done recently by Yates and Bennett[18]. The second is the quasi-steady correction for finite span in the region of a tip Mach cone that was applied by Rodden and Revell[19] to the strip theory flutter coefficients derived from the supersonic airfoil theory of Van Dyke[17].

REFERENCES

1. MacNeal, R. H., and Rodden, W. P., "Requirements for the Addition of Aeroelastic Capability to NASTRAN," MacNeal-Schwendler Corp. Report No. MSR-21, March 1970.

2. Stenton, T. E., and Andrew, L. V., "Transonic Unsteady Aerodynamics for Planar Wings with Trailing Edge Control Surfaces," FDL-TR-67-180, November 1967, U. S. Air Force Flight Dynamics Laboratory

3. Kariappa and Smith, G.C.C., "Further Developments in Consistent Unsteady Supersonic Aerodynamic Coefficients," AIAA Paper No. 71-177, presented at AIAA 9th Aerospace Sciences Meeting, New York, 25-27 January, 1971.

4. Ashley, H., and Zartarian, G., "Piston Theory - A New Aerodynamic Tool for the Aeroelastician," J. Aero. Sci., Vol. 23, No. 12, December 1956, pp 1109 - 1118.

5. Donato, V. W., and Huhn, C. R., "Supersonic Unsteady Aerodynamics for Wings with Trailing Edge Control Surfaces and Folded Tips," AFFDL-TR-68-30, January 1968, U. S. Air Force Flight Dynamics Laboratory.

6. Andrew, L. V., "Unsteady Aerodynamics for Advanced Configurations; Part VI - Application of the Supersonic Mach Box Method to T-Tails, V-Tails, and Top Mounted Verticals," FDL-TDR-64-152, Pt. VI, October 1968, U. S. Air Force Flight Dynamics Laboratory.

7. Andrew, L. V., and Moore, M. T., "Further Developments in Supersonic Aerodynamic Influence Coefficient Methods," Proc. of AIAA Specialists Meeting on Structural Dynamics and Aeroelasticity, August 1965, pp 203-213.

8. Watkins, E. E., Woolston, D. S., and Cunningham, H. J., "A Systematic Kernel Function Procedure for Determining Aerodynamic Forces on Oscillating or Steady Finite Wings at Subsonic Speeds," NASA Report R-48, 1959.

9. Berman, J. H., Shyprykevich, P., Smedfjeld, J. B., and Kelly, R. F., "Unsteady Aerodynamic Forces for General Wing/Control-Surface Configurations in Subsonic Flow," AFFDL-TR-67-117, May 1968, U. S. Air Force Flight Dynamics Laboratory.

10. Albano, E., Perkinson, F., and Rodden, W. P., "Subsonic Lifting-Surface Theory Aerodynamics and Flutter Analysis of Interfering Wing/Horizontal-Tail Configuration," AFFDL-TR-70-59, September 1970, U. S. Air Force Flight Dynamics Laboratory.

11. Andrew, L. V., "Subsonic Generalized Forces on Aerodynamically Interfering Surfaces," Report No. NA-69-904, March 1971, North American Rockwell Corporation.

12. Cunningham, H. J., "Improved Numerical Procedure for Harmonically Deforming Lifting Surfaces from the Supersonic Kernel Function Method," AIAA Journal, Vol. 4, No. 11, 1966, pp 1961 - 1968.

13. Cunningham, H. J., "Application of a Supersonic Kernel-Function Procedure to Flutter Analysis of Thin Lifting Surfaces," NASA TN D-6012, November 1970.

14. Curtis, A. R., and Lingard, R. W., Jr., "Unsteady Aerodynamic Distributions for Harmonically Deforming Wings in Supersonic Flow," AIAA Paper No. 68-74, January 1968.

15. Rodden, W. P., Farkes, E. T., Malcom, H. A., and Kliszewski, A. M., "Aerodynamic Influence Coefficients from Piston Theory: Analytical Development and Computational Procedure," Report No. TDR-169 (3230-11) TN-2, 15 August 1962, Aerospace Corporation.

16. Anon., "Modal Flutter Analysis Study; Vol. I - Supersonic Piston Theory Unsteady Aerodynamics Program," Report No. P71-01-CP-Vol. 1, January 1971, Hughes Aircraft Company.

17. Van Dyke, M. D., "Supersonic Flow Past Oscillating Airfoils Including Nonlinear Thickness Effects," NACA TN-2982, July 1953.

18. Yates, E. C., Jr., and Bennett, R. M., "Analysis of Supersonic-Hypersonic Flutter of Lifting Surfaces at Angle of Attack," AIAA Paper No. 71-327, presented at AIAA/ASME 12th Structures, Structural Dynamics and Materials Conference, Anaheim, California, 19-21 April 1971.

19. Rodden, W. P., and Revell, J. D., "Oscillatory Aerodynamic Coefficients for a Unified Supersonic-Hypersonic Strip Theory," J. Aero. Sci., Vol. 27, No. 6, June 1960, pp 451 - 459.

APPENDIX H

APPENDIX H

A Transformation Method for Eigenvalue Extraction of Non-Hermitian Matrices

This appendix presents an outline of the algorithms required for extraction of eigenvalues and eigenvectors from general, real, or complex matrices. The techniques are obviously more general than those employed in the Tridiagonal Method for Real Symmetric Matrices in NASTRAN (Ref. 1, Sect. 10.2) but the NASTRAN Theoretical Manual is a basic reference in the context of extensions of its capability to include aeroelastic analyses. The fundamental reference is Wilkinson[2] for both the real and complex cases, but Parlett[3] has given a concise discussion of the algorithms for the real case, and the IBM 360 Library Subroutine Manual includes two important subroutines[4,5] for real matrices. The algorithms of Wilkinson for complex matrices have been automated by Funderlic and Rinzel in the subroutine ALLMAT[6].

The present outline is given because the available subroutines [4,5,6] must be reviewed for application to larger problems in NASTRAN and since the real subroutines [4,5] do not include the capability for eigenvector calculation. The outline is presented in the logical order of the calculations: Reduction to Upper Hessenberg Form, the QR Iteration, Convergence Criteria, Shifting, Deflation, and Eigenvectors. Emphasis is placed on the complex case since this is a primary requirement for flutter analysis.

Reduction to Upper Hessenberg Form

We denote the given matrix by [A] in the eigenvalue problem $[A-\lambda I]u = 0$. Reduction to this form requires decomposition of the mass matrix, see NASTRAN Theoretical Manual, Section 9.2. We reduce [A] to the upper Hessenberg matrix $[A_0]$ by using elementary stabilized transformations. The basic algorithm and two alternatives are given by Wilkinson[2] (pp. 354 - 355).

The subroutine ALLMAT [6] uses Eq. (9.1) of Ref. 2 (p. 355) in its reduction and it appears to be appropriate for the task. The total number of multiplications in the complete reduction is approximately $(5/6)n^3$ which is half the number in Householder's reduction and one-quarter the number in Givens' reduction.

The QR Iteration

The QR iteration of Francis [7] is defined by the relations (Wilkinson[2], p. 515)

$$\left[A_0^{(s)}\right] = \left[Q^{(s)}\right]\left[R^{(s)}\right] \tag{1}$$

$$\left[A_0^{(s+1)}\right] = \left[R^{(s)}\right]\left[Q^{(s)}\right] \tag{2}$$

where $\left[Q^{(s)}\right]$ is the product of the (n-1) elementary unitary transformations necessary to reduce $\left[A_0^{(s)}\right]$ to the upper triangular form $\left[R^{(s)}\right]$ with positive real diagonal elements,

$$\left[Q^{(s)}\right] = \left[T_s^{(1)}\right]\left[T_s^{(2)}\right] \ldots \left[T_s^{(n-2)}\right]\left[T_s^{(n-1)}\right] \tag{3}$$

so that

$$\left[R^{(s)}\right] = \left[Q^{(s)}\right]^H \left[A^{(s)}\right] \tag{4}$$

The transformation matrices $\left[T_s^{(j)}\right]$ are the Givens' rotations as discussed by Wilkinson [2] (p. 239-240) and in the NASTRAN Theoretical Manual [1] (Sect. 10.2) in real form for real matrices <u>but in complex form</u> for complex matrices. The iteration is continued until the nth diagonal element $|a^{(s)}_{n,n-1}| < \varepsilon$, the convergence test, at which point the smallest eigenvalue $\lambda_1 = a^{(s)}_{n,n}$; if the convergence proceeds so that $|a^{(s)}_{n-1,n-2}| < \varepsilon$ before $|a^{(s)}_{n,n-1}| < \varepsilon$, the two smallest eigenvalues are the roots of

$$\begin{vmatrix} a^{(s)}_{n-1,n-1} - \lambda & a^{(s)}_{n-1,n} \\ a^{(s)}_{n,n-1} & a^{(s)}_{n,n} - \lambda \end{vmatrix} = 0 \tag{5}$$

The roots will be complex for complex matrices, and either real or complex conjugates for real matrices. Before each iteration, the subdiagonal elements should be tested and if some $|a^{(s)}_{i,i-1}| < \varepsilon$, the matrix should be split according to this occurrence, and the iteration continued with the lower main submatrix only.

Convergence Criteria

The convergence criteria suggested by Wilkinson [2] (p.526) and Parlett [3] (p. 123) is based on the Euclidean norm of the matrix $||A_0||_E$ and is

$$\varepsilon = 2^{-t} \; ||A_0||_E \tag{6}$$

for floating-point calculations with mantissas of t binary bits. The Euclidean norm (Wilkinson [2], p. 57) is found from

$$||A||^2_E = \sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|^2 \tag{7}$$

Subroutines ATEIG [5] and ALLMAT [6] use decimal equivalents of Eq. (6).

Shifting

Since the QR iteration converges to the smallest eigenvalue, the convergence can be accelerated by shifting, i.e., by subtracting scalar matrices from the original matrix. The matrix $\left[A_0^{(s)}\right]$ is replaced by the difference $\left[A_0^{(s)}\right] - k_s [I]$ after each iteration, in which k_s is an estimate of the eigenvalue. The shift eigenvalue k_s is that root of Eq. (5), p_s or q_s, that makes $|a_{n,n}^{(s)} - p_s|$ or $|a_{n,n}^{(s)} - q_s|$ a minimum. The shifted algorithm then becomes (Wilkinson, p. 524)

$$\left[A_0^{(s)}\right] - k_s [I] = \left[Q^{(s)}\right]\left[R^{(s)}\right] \tag{8}$$

and

$$\left[A_0^{(s+1)}\right] = k_s [I] + \left[R^{(s)}\right]\left[Q^{(s)}\right] \tag{9}$$

Equations (8) and (9) represent the algorithm for a shift to a single eigenvalue and is appropriate for a complex matrix or a real matrix with all real eigenvalues. The possibility of complex conjugate eigenvalues in general unsymmetric real matrices suggests a shifting technique that will retain real arithmetic. An elegant procedure has been given by Francis [7] and is discussed by Wilkinson [2] (pp. 528-537) and Parlett [3] and utilized in subroutine ATEIG [5]; it is called the Double QR Tranformation. The algorithm permits a double shift after each iteration and the unitary transformation is that of Householder (Wilkinson [2], p. 533) rather than that of Givens'.

Deflation

When convergence to a single eigenvalue occurs, i.e., when $|a^{(s)}_{n,n-1}| < \varepsilon$, the Hessenberg matrix $[A_0]$ is deflated by elimination of its last row and column and the principal submatrix $[A_1]$ of order one less is the Hessenberg form for seeking the next eigenvalue. (Note: the subscript on A denotes the number of eigenvalues removed from $[A_0]$.) If convergence occurs to a pair of eigenvalues, i.e., $|a^{(s)}_{n-1,n-2}| < \varepsilon$, the matrix $[A_0]$ is deflated by deleting the last two rows and columns, and the principal submatrix $[A_2]$ of order two less becomes the basis for seeking the next eigenvalue. Each deflation removes either one or two eigenvalues depending on the two convergence tests; the Double QR Transformation always deflates two eigenvalues at a time.

Eigenvectors

The inverse power method with shifts (Wilkinson [2], pp. 626-628) converges rapidly to the eigenvector corresponding to each shift eigenvalue. This algorithm for real and complex matrices has been discussed throughly in the NASTRAN Theoretical Manual [1] (Sect. 10.4); the complex case is discussed in Sect. 10.4.4. No further discussion is required here.

REFERENCES

1. MacNeal, R.H., (ed.), "The NASTRAN Theoretical Manual," NASA SP-211, October 1969.

2. Wilkinson, J. H., The Algebraic Eigenvalue Problem, Oxford: Clarendon Press, 1965.

3. Ralston, A., and Wilf, H.S., (ed.) Mathematical Methods for Digital Computers, Vol. II, New York. John Wiley and Sons, Inc. 1968, pp. 116-130.

4. Anon., "Subroutine HSBG," IBM 360 Library Subroutine Manual, Form H20-0205-3, 14 February 1969, pp. 169-170.

5. Anon., "Subroutine ATEIG", IBM 360 Library Subroutine Manual, Form H20-0205-3, 14 February 1969, pp. 167-169.

6. Funderlic, R.E., and Rinzel, J., "ALLMAT, A Fortran IV Arbitrary Matrix Eigensystem Solver," SHARE Program Library, SDA. No. 3441, 2 March 1966.

7. Francis, J.G.F., "The QR Transformation - a Unitary Analogue to the LR Transformation, Parts 1 and 2, Computer Journal, Vol. 4, No. 3 (October 1961) and No. 4 (January 1962).

APPENDIX I

APPENDIX I

A Modified Power Method for Flutter Analysis by the P-K Method

The general aeroelastic eigenvalue problem including viscous damping terms may be solved by the p-k method, see Appendix F. The eigenvalue problem is stated as follows:

$$[Mp^2 + Bp + K + K^a]\{u\} = 0 \qquad (1)$$

where M, B, and K are structural and control system coefficients, K^a is the aerodynamic stiffness matrix and p is the complex eigenvalue parameter. K^a is a function of Mach number, m, and reduced frequency, k. The reduced frequency is related to the imaginary part of the eigenvalue parameter, ω, since

$$p = \alpha + i\omega \qquad (2)$$

and

$$k = \frac{b\omega}{V} \qquad (3)$$

During the course of the iteration procedure which produces the eigenvalue, $p_j = \alpha_j + i\omega_j$, the aerodynamic matrix is adjusted so that it is evaluated for $k = k_j = b\omega_j/V$. The general procedure is to select an initial estimate, λ_o, of p, where λ_o is an eigenvalue obtained from a previous solution with slightly different values of Mach number and/or air density, and to express K^a as the sum of two terms

$$K^a = K^{a0} + K^{a1} \qquad (4)$$

where $K^{a0} = -\frac{1}{2}\rho V^2[Q(m,k_o)]$ is the value of $[K^a]$ for the current values of ρ and m and the previously obtained eigenvalue

$$\lambda_o = \alpha_o + i\omega_o \tag{5}$$

such that

$$k_o = \frac{b\omega_o}{V} \tag{6}$$

The second term

$$K^{a1} = -\frac{1}{2}\rho V^2[Q(m,k) - Q(m,k_o)] \tag{7}$$

where k is the correct value of the reduced frequency. K^{ao} is a matrix of known coefficients whereas K^{a1} can only be estimated. The estimate is updated during the iteration.

The eigenvalue problem differs from that treated in the existing NASTRAN complex eigenvalue module only to the extent that K^{a1} is present. The proposed method of solution is a modification of the complex inverse power method described in Section 10.4.4 of the NASTRAN Theoretical Manual, Ref. 1. The iteration algorithm given by Eq. (14) of Section 10.4.4 will be replaced by the following:

$$\left[\lambda_o{}^2 M + \lambda_o B + K + \enclose{circle}{K^{a0}}\right]\{w_n\} = -\left[B + \lambda_o M + \frac{1}{\Lambda_{n-1}}\enclose{circle}{K^{a1}}\right]\{u_{n-1}\} - [M]\{V_{n-1}\} \tag{8a}$$

$$\{u_n\} = \frac{1}{c_n}\{w_n\} \tag{8b}$$

$$\{V_n\} = \lambda_o\{u_n\} + \frac{1}{c_n}\{u_{n-1}\} \tag{8c}$$

The circled terms are the only modifications. c_n is equal to the element of $\{w_n\}$ with largest magnitude. The current estimate of the eigenvalue is

$$p_n = \lambda_o + \Lambda_n \tag{9}$$

where

$$\Lambda_n = \frac{1}{c_n}\left[\frac{\{u^*_{n-1}\}^T[M]\{u_{n-1}\}}{\{u^*_n\}^T[M]\{u_n\}}\right]^{1/2} \tag{10}$$

$\{u^*_n\}^T$ is the transpose of the conjugate of $\{u_n\}$.

Only one eigenvalue will be extracted for each value of λ_o, so that the sweeping of previously extracted roots is not required. The starting vectors are $\{u_o\}$ and $\{v_o\} = \lambda_o\{u_o\}$ where $\{u_o\}$ is the eigenvector obtained in the previous analysis. In the event that no previous flutter case has been analyzed, $\{u_o\}$ will be taken from a free vibration analysis of the structure.

Since neither Λ nor K^{al} can be estimated before the first iteration, the initial value of the product $(1/\Lambda_{n-1})\, K^{al}$ which appears on the right of Eq. (8a) will be taken to be zero.

The convergence criteria provided with the current NASTRAN version of the inverse power method will remain unchanged.

In the iteration procedure, K^{al} will be estimated from a cubic spline fit to the values of the aerodynamic matrix [Q]. The coefficients in the spline fit will be kept in auxilliary storage. In any given interval of reduced frequency, $k_a < k < k_b$,

$$Q(k) = Q_a + (k-k_a)Q^1_a + (k-k_a)^2Q^2_a + (k-k_a)^3Q^3_a \tag{11}$$

The stored coefficients are Q_a, Q^1_a, Q^2_a and Q^3_a. As long as k_n, the current estimate of k, and k_o, the initial estimate of k, are in the same interval, k_a, k_b, only three sets of coefficients need be kept in high speed memory because

$$[K^{a1}] = \frac{1}{2}\rho V^2[Q(k_n) - Q(k_o)]$$

$$= \frac{1}{2}\rho V^2\Big[(k_n-k_o)Q_a^1 + \big((k_n-k_a)^2-(k_o-k_a)^2\big)Q_a^2 + \big((k_n-k_a)^3-(k_o-k_a)^3\big)Q_a^3\Big] \tag{12}$$

If k_n goes outside the range, for example by crossing the upper bound, k_b, four sets of coefficients should be stored in high speed memory because

$$[K^{a1}] = \frac{1}{2}\rho V^2\Big[\big(Q_b-Q(k_o)\big) + (k_n-k_b)Q_b^1 + (k_n-k_b)^2Q_b^2 + (k_n-k_b)^3Q_b^3\Big] \tag{13}$$

The algorithm given by Eq. (8) will also work for the special case [B] = 0. It can, however, be simplified and its convergence can, perhaps, be improved. Let

$$p^2 = \lambda_o + \Lambda \tag{14}$$

so that the eigenvalue problem may be stated as

$$[\lambda_o M + K + K^{a0}]\{u\} = -[K^{a1} + \Lambda M]\{u\} \tag{15}$$

The corresponding iteration algorithm is

$$[\lambda_o M + K + K^{a0}]\{w_n\} = -[M + \frac{1}{\Lambda_{n-1}} K^{a1}]\{u_{n-1}\} \tag{16a}$$

$$\{u_n\} = \frac{1}{c_n}\{w_n\} \tag{16b}$$

where c_n is equal to the element of $\{w_n\}$ with largest magnitude, and Λ_{n-1} is the current estimate of Λ, which is still computed by means of Eq. (10).

Convergence of algorithm given by Eq. (16) will now be explored. The orthogonality properties of the eigenvectors are

$$\{\bar{\phi}_j\}^T[M]\{\phi_i\} = \delta_i^j \tag{17}$$

$$\{\bar{\phi}_j\}^T[K + K^{a0} + K^{a1}]\{\phi_i\} = -p_j^2\ \delta_i^j \tag{18}$$

where $\{\phi_i\}$ is an eigenvector of Eq. (15), and $\{\bar{\phi}_j\}$ is a "left" eigenvector, i.e., an eigenvector of Eq. (15) with $[M]$ and $[K + K^{a0} + K^{a1}]$ replaced by their transposes. δ_i^j is the Kronecker delta. The trial vectors $\{u_n\}$ and $\{u_{n-1}\}$ can be expanded in terms of eigenvectors as follows:

$$\{u_n\} = \sum_i \alpha_{in}\{\phi_i\} \tag{19}$$

$$\{u_{n-1}\} = \sum_i \alpha_{i,n-1}\{\phi_i\} \tag{20}$$

Hence, upon substituting into Eq. (16), and premultiplying by $\{\bar{\phi}_j\}^T$,

$$c_n\{\bar{\phi}_j\}^T[\lambda_o\ M + K + K^{a0}] \sum_i \alpha_{in}\{\phi_i\} = -\ \{\bar{\phi}_j\}^T[M + \frac{1}{\Lambda_{n-1}}\ K^{a1}] \sum_i \alpha_{i,n-1}\{\phi_i\} \tag{21}$$

Invoking the orthogonality properties

$$c_n[(\lambda_o - p_j^2)\alpha_{jn}] = -\ \alpha_{j,n-1} + \bar{\phi}_j^T[K^{a1}]\Big\{c_n \sum_i \alpha_{i,n}\{\phi_i\} - \frac{1}{\Lambda_{n-1}} \sum_i \alpha_{i,n-1}\{\phi_i\}\Big\} \tag{22}$$

If $[K^{a1}]$ were null, then

$$\frac{\alpha_{jn}}{\alpha_{j,n-1}} = \frac{1}{c_n(p_j^2 - \lambda_o)} = \frac{1}{c_n\Lambda_j} \tag{23}$$

which shows that the trial vector would converge to the eigenvector with smallest Λ_j.

In the general case, assuming that the increment in reduced frequency is small, $[K^{a1}]$ can be assumed to be linearly proportional to the increment in reduced frequency, see Eq. (12). Thus,

$$[K^{a1}] \approx \frac{1}{2}\rho V^2(k_{n-1} - k_o)[Q_a^1] \tag{24}$$

In order to simplify the problem, assume that $k_n - k_o$ is proportional to $\Lambda_n = p_n^2 - \lambda_o = (p_n - p_o)(p_n + p_o)$ and that $[Q_a^1]$ is proportional to $[K + K^a]$. Thus, it is assumed that

$$[K^{a1}] = \eta \frac{\Lambda_{n-1}}{\lambda_o}[K + K^a] \tag{25}$$

where η is a dimensionless constant of proportionality.

Then, using Eq. (25) and the orthogonality conditions in Eq. (22),

$$-c_n \Lambda_j \alpha_{jn} = -\alpha_{j,n-1} - \eta\left(1 + \frac{\Lambda_j}{\lambda_o}\right)(c_n \Lambda_{n-1} \alpha_{jn} - \alpha_{jn-1}) \tag{26}$$

and solving for the ratio $\alpha_{jn}/\alpha_{j,n-1}$

$$\frac{\alpha_{jn}}{\alpha_{j,n-1}} = \frac{1 - \eta\left(1 + \frac{\Lambda_j}{\lambda_o}\right)}{c_n \Lambda_j - \eta c_n \Lambda_{n-1}\left(1 + \frac{\Lambda_j}{\lambda_o}\right)} \tag{27}$$

The product

$$c_n \Lambda_{n-1} \approx c_n \Lambda_1 \approx 1 \tag{28}$$

for a reasonably converged solution so that

$$\frac{\alpha_{jn}}{\alpha_{j,n-1}} \approx \frac{1 - \eta\left(1 + \frac{\Lambda_j}{\lambda_o}\right)}{\frac{\Lambda_j}{\Lambda_1} - \eta\left(1 + \frac{\Lambda_j}{\lambda_o}\right)} \tag{29}$$

A sufficient condition for <u>absolute convergence</u> of Eq. (29) is that

$$\left|\frac{\Lambda_j}{\Lambda_1}\right| > 1 + 2|\eta|\cdot\left|1 + \frac{\Lambda_j}{\lambda_o}\right| \tag{30}$$

for a small increment in the position of the lowest eigenvalue,

$$|\Lambda_1| << |\lambda_o|$$

and (31)

$$|\Lambda_1| << |\Lambda_j| \qquad j > 1$$

if the roots are well separated.

Thus, a convergent solution will result from a sufficiently small increment in the lowest eigenvalue. If $|\eta|$ is of the order of unity and the eigenvalues are well separated, the increment may actually be rather large, as is shown by Eq. (30).

APPENDIX J

APPENDIX J

Surface Splines for Aerodynamic Matrix Interpolation

Surface Spline Techniques

Surface spline equations could be used as an alternative to a grid of linear splines for the Aerodynamic Matrix Interpolation Module, (see Section 5.7). The characteristics of the surface spline method are:

Advantages

1. It does not require a rectangular grid of parameter values. This would be useful to check interpolation. For example, after a flutter condition is found, the resulting (m,k) parameter pair could be added to the calculation list, and then a new flutter point found. Any differences are presumably due to interpolation inaccuracies.

2. The mathematics would be easier to program, since there are less steps in the calculation.

Disadvantages

1. It would be slower in the final stage, i.e., calculation time in the inner loop of a looping procedure would increase.

2. It is poor in the one-parameter case.

The formal mathematics is the same as is used in the geometry interpolator, (see Section 5.4). One additional feature which is suggested is scaling the m's and k's. The equation of the surface is

$$F(m,k) = C_0 + C_1 x + C_2 y + \sum_{j=1}^{N} P_j \, r_j^2 \ln r_j^2 \qquad (1)$$

where $x = m/R_m$, R_m = scale factor

$y = k/R_k$, R_k = scale factor

$r_j^2 = (x-x_j)^2 + (y-y_i)^2$

The unknowns C_0, C_1, C_2, $P_i(i=1,N)$ are found from

$$0 = \sum P_i = \sum x_i P_i = \sum y_i P_i \tag{2}$$

$$F_i = F(m_i,k_i) = C_0 + C_1 x_i + C_2 y_i + \sum_{j=1}^{N} P_j r_{ij}^2 \ln r_{ij}^2 \text{ , } i=1,N \tag{3}$$

where $r_{ij}^2 = \left(\frac{m_i - m_j}{R_m}\right)^2 + \left(\frac{k_i - k_j}{R_k}\right)^2$

Using matrix notation

$$F = \left[R \;\vdots\; A \right] \left\{\frac{C}{P}\right\} \tag{4}$$

$$\left\{\begin{array}{c} 0 \\ \hline F_i \end{array}\right\} = \left[\begin{array}{c|c} 0 & R_i^T \\ \hline R_i & A_{ij} \end{array}\right] \left\{\begin{array}{c} C \\ \hline P \end{array}\right\} \tag{5}$$

where $\lfloor R \rfloor = \lfloor 1, x, y \rfloor$ (6)

$$A = [r_1^2 \ln r_1^2 , r_2^2 \ln r_2^2, \; . \; . \; . \; . \; r_N^2 \ln r_N^2] \tag{7}$$

$$\left[R_i\right] = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ & \vdots & \\ 1 & x_N & y_N \end{bmatrix} \tag{8}$$

$$A_{ij} = \begin{bmatrix} 0 & r^2_{12}\ln r^2_{12} \cdots & r^2_{1N}\ln r^2_{1N} \\ r^2_{12}\ln r^2_{12} & 0 & \cdots\cdots \\ r^2_{1N}\ln r^2_{1N} & \cdots\cdots & 0 \end{bmatrix} \quad (9)$$

The formal solution is

$$F = \lfloor R \mid A \rfloor \left[\begin{array}{c|c} 0 & R_i^T \\ \hline R_i & A_{ij} \end{array}\right]^{-1} \left\{\begin{array}{c} 0 \\ \hline F_i \end{array}\right\} \quad (10)$$

$$= \lfloor G \rfloor \left\{ F_i \right\} \quad (11)$$

where the interpolation vector G comes from the solution of

$$\left[\begin{array}{c|c} 0 & R_i^T \\ \hline R_i & A_{ij} \end{array}\right] \left\{\begin{array}{c} x^T \\ \hline G^T \end{array}\right\} = \left\{\begin{array}{c} R^T \\ \hline A^T \end{array}\right\} \quad (12)$$

The Aerodynamic Matrix Interpolation would be done in two stages. The first stage does not concern the values of the aerodynamic matrices, but merely the values of the parameters for which the matrix is known. The job is to set up the coefficient matrix of equation (12) and decompose it into triangular factors. The second phase, which occurs when interpolation is desired, is to evaluate the right hand side of equation (12), solve for the solution vector G of interpolation weights, and then compute (using equation (11) for all matrix terms) the weighted average of the known aerodynamic matrices.

The number of non-zero terms in the interpolation sums can be compared with the number using linear splines:

Surface spline : the number of (m,k) pairs
Linear spline : 4

The surface spline method could be used as an alternate method, or perhaps as the sole method of Aerodynamic Matrix Interpolation.

Mixed spline techniques

Another possibility for parametric Aerodynamic Matrix Interpolation is a mixture of surface splines and linear splines (instead of a lattice of linear splines). Define the following four formats for aerodynamic matrices.

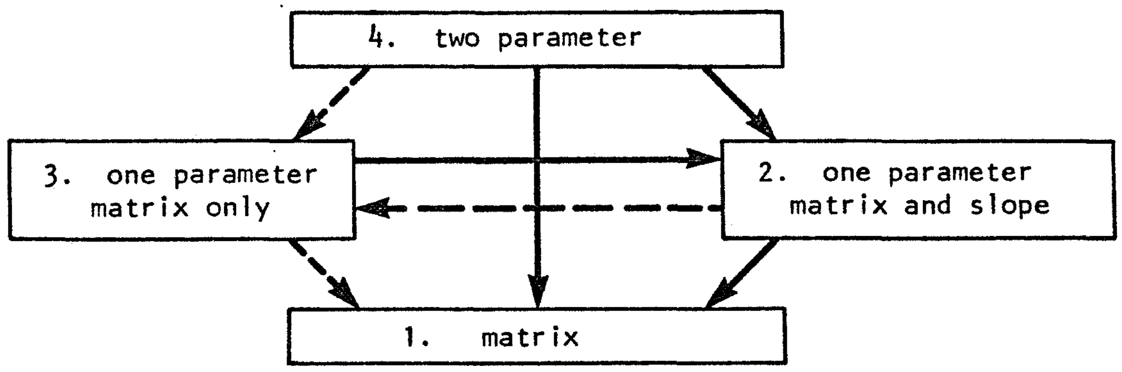
1. A single matrix. This is the format desired for output from an interpolation.

2. A single parameter list of values of the matrix and its first derivative for a set of parameters. This form is useful for linear interpolation.

3. A single parameter list of matrices. This is a type of data that may be computed in aero-statics for example, where Mach number is the only parameter.

4. A double parameter list of matrices. This is the type which would be generated in two-parameter theories, such as doublet lattice.

The sketch shows the different functions of a matrix interpolator.


4. two parameter
3. one parameter matrix only
2. one parameter matrix and slope
1. matrix

Each arrow represents a change of format which the interpolator can make. The solid lines are the most necessary, and will require spline calculations. The dotted lines are less necessary, and are intended to be used when the set of parameter values of the result is a subset of the parameters of the original format.

The calculations for the transitions are

4-1. Surface spline, as discussed in the first part of this appendix.

4-2. Surface spline, outputting the matrix and its slope at requested parameter values.

4-3. This might be used if one parameter of the parameter pair is frozen at one of the list of values. For example, if the matrix is known for m = 0, .1 and .3 for a set of values of k in format 4, it is easy to output a list of values for m = 0 and the same set of values of k in format 3.

3-2. This requires solving the linear spline equations to solve for the slopes. This is discussed in section 5.7.

3-1. When the matrix is desired to be chosen from a list, there is no need to use splines.

2-3. Of little value, but it is easy since this only requires partitioning out the slopes.

2-1. The final step in linear splining, as described in section 5.7. This is useful, since it should be fast, for use in inner loops of algorithms.

The following transitions would be used in the algorithms.

		with looping	
	no looping	outside loop	inside loop
1 parameter	3-2, 2-1	3-2	2-1
2 parameter	4-1	4-2	2-1

APPENDIX K

APPENDIX K

Data Block Descriptions

Due to the modular nature of the NASTRAN program, all data passed between modules must exist in data blocks or be in parameter lists. The data blocks may be saved for use in restart. There are two types of data blocks.

1. Matrices. Matrix data blocks are one or two dimensional arrays used for vectors and matrices.

2. Tables. All other data blocks are called tables.

The format of a matrix is standardized so that any of the arithmetic modules may use them. The format of the tables varies.

A new feature of the aerodynamic part of NASTRAN is the use of lists of matrices. These are used, for example, to generate a set of aerodynamic matrices for different values of reduced frequency and Mach number. The lists of matrices will be stored in table data blocks, so that no new data block types are needed. Within the tables, the matrices can be stored in the same packed format as is now used for matrices.

Each data block must have a unique MNEMONIC. A NASTRAN DICTIONARY (Section 7 of NASTRAN Users Manual) lists existing names. It also includes module names. Some sample additions needed for aerodynamics are shown in Figure 1.

Term	Code*	Definition
AEDECK	DBT	Aerodynamic Bulk Data Cards
AEG	FMA	Aerodynamic Element Generator
AELEM	DBT	Aerodynamic Element Definitions
AJJINV	DBM	$[A_{jj}]^{-1}$ Aerodynamic Influence Matrix
AJJT	DBT	List of A_{jj}^{-1} and A_{jj} Factors
APD	FMA	Aerodynamic Pool Distributor
ATDCN	DBT	Aerodynamic Data Table, Connectors
ATDLD	DBT	Aerodynamic Data Table, Loads
ATDMD	DBT	Aerodynamic Data Table, Methods
ATDPR	DBT	Aerodynamic Data Table, Properties
ATDSP	DBT	Aerodynamic Data Table, Splines
DJE	DBM	$[D_{je}]$ Aerodynamic Downwash Matrix
DJK	DBM	$[D_{jk}]$ Aerodynamic Downwash Matrix
GKA	DBM	$[G_{ka}]$ Interpolation Matrix
KADD	DBM	$[K^a_{dd}]$ Aerodynamic Stiffness Matrix
KAHH	DBM	$[K^a_{hh}]$ Aerodynamic Stiffness Matrix
KALE	DBM	$[K^a_{\ell e}]$ Aerodynamic Stiffness Matrix
KALL	DBM	$[K^a_{\ell\ell}]$ Aerodynamic Stiffness Matrix
LJJ	DBM	Lower Triangular Factor of $[A_{jj}]$
QHE	DBM	$[Q_{he}]$ Aerodynamic Matrix
QHET	DBT	List of $[Q_{he}]$ Matrices
QHH	DBM	$[Q_{hh}]$ Modal Aerodynamic Matrix
QHHT	DBT	List of $[Q_{hh}]$ Matrices
QIE	DBM	$[Q_{ie}]$ Aerodynamic Matrix Partition of $[Q_{hh}]$

Figure 1. Sample Mnemonic Dictionary Additions

* FMA - Functional Module - Aerodynamic; DBM - Data Block - Matrix; DBT - Data Block - Table

Term	Code*	Definition
QII	DBM	$[Q_{ii}]$ Aerodynamic Matrix Partition of $[Q_{hh}]$
QIJ	DBM	$[Q_{ij}]$ Modal Aerodynamic Matrix for Gusts
SKJ	DBM	$[S_{kj}]$ Aerodynamic Area Matrix
UJJ	DBM	Upper Triangular Factor of $[A_{jj}]$
USETKJ	DBT	Definitions of Sets k and j

* FMA - Functional Module - Aerodynamic
DBM - Data Block - Matrix
DBT - Data Block - Table

Figure 1 (Cont'd.)