NASA Technical Memorandum 78710

(NASA-TM-78710) DESCRIPTION OF A COMPUTER N78-29865
PROGRAM AND NUMERICAL TECHNIQUES FOR
DEVELOPING LINEAR PERTURBATION MODELS FROM
NONLINEAR SYSTEMS SIMULATIONS (NASA) 41 p Unclas
HC A03/MF A01 CSCL 12A G3/64 25913

Description of a Computer Program and Numerical Technique for Developing Linear Perturbation Models From Nonlinear Systems Simulations

James E. Dieudonne

JULY 1978

NASA

NASA Technical Memorandum 78710

Description of a Computer Program and Numerical Technique for Developing Linear Perturbation Models From Nonlinear Systems Simulations

James E. Dieudonne
Langley Research Center
Hampton, Virginia

NASA
National Aeronautics
and Space Administration

Scientific and Technical
Information Office

1978

SUMMARY

A numerical technique has been developed at Langley Research Center which generates linear perturbation models from nonlinear aircraft vehicle simulations. The technique is very general and can be applied to simulations of any system that is described by nonlinear differential equations. The computer program used to generate these models is discussed with emphasis placed on generation of the Jacobian matrices, calculation of the coefficients needed for solving the perturbation model, and generation of the solution of the linear differential equations. Included in the paper is an example application of the technique to a nonlinear model of the NASA Terminal Configured Vehicle.

INTRODUCTION

Linearized models of physical systems, when used either in conjunction with nonlinear simulations of the physical systems or in an independent mode, offer the research engineer many insights to his problem. Obviously, a nonlinear simulation will be a more valid model over a much larger range of the system's state variables, but with the present state of the art of mathematics and systems design techniques, linear models offer many advantages.

A linear model used to represent the system over some limited region has a known analytical solution which can be programmed on a digital computer and does not require standard numerical integration techniques (ref. 1). This property can result in a savings in both computation time requirements and computer memory allocations for the simulation of the system. Many available computer algorithms have been written which will identify the eigenvalues and eigenvectors of a linear model (ref. 2). This gives the researcher a quick look at the characteristic modes of the system. Once these modes have been identified, steps can be taken to eliminate undesirable characteristics by adding a feedback control system. At present, many books and articles have been written on the subject of linear feedback controls, and many computer programs are available which will provide such features as root placement and the solution to both the time-varying and steady-state optimal regulator problems (refs. 3 to 7).

This report will describe a computer program which was designed to obtain linear models about a nominal state and control vector from nonlinear real-time aircraft simulations. The program is very general in design and may be applied to any system that is described by a set of nonlinear differential equations about any trajectory in state space. The program uses various Lagrange interpolation formulas to obtain both the state and control Jacobian matrices. Once they are obtained, the linear differential equations are integrated by using the local linearization technique described in reference 1. Eigenvectors and eigenvalues are calculated using standard computer routines that are available at Langley Research Center.

SYMBOLS

$A(t)$ — $n \times n$ dimensional state Jacobian matrix, $A(t) = \left.\frac{\partial \bar{f}}{\partial \bar{x}}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}}$

a_{ij} — element of $A(t)$ located at intersection of ith row and jth column

$B(t)$ — $n \times k$ dimensional control Jacobian matrix, $B(t) = \left.\frac{\partial \bar{f}}{\partial \bar{u}}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}}$

$\bar{c}$ — mean aerodynamic chord, meters

$\bar{F}$ — vector of total aerodynamic forces

$\bar{f}(\bar{x},\bar{u})$ — n dimensional vector of general, nonlinear time-varying functions of state vector $\bar{x}$ and control vector $\bar{u}$

$f_i(\ )$ — ith component of $\bar{f}$

g — acceleration due to gravity, meters per second

h — integration interval step size, seconds

$\bar{h}(t)$ — n dimensional vector whose elements are residual higher order terms from Taylor series expansion

I — $n \times n$ identity matrix

$I_{XX}, I_{YY}, I_{ZZ}, I_{XZ}$ — moments of inertia, kilograms-meters^2

J_u — $n \times k$ Jacobian matrix of $\bar{f}$ with respect to $\bar{u}$

J_x — $n \times n$ Jacobian matrix of $\bar{f}$ with respect to $\bar{x}$

k — constant equal to number of elements in control vector $\bar{u}$

L/D — lift-drag ratio

$\ell_k(\ )$ — coefficients used in Lagrange interpolation formulas for approximation of f_i

$\ell_k'(\ )$ — coefficients used in Lagrange interpolation formulas for approximation of $\partial f_i/\partial x_j$

$\bar{M}$ — vector of total aerodynamic moments

M_{max} — maximum operating Mach number

n	constant equal to number of elements in state vector x
P	$n \times n$ dimensional matrix which is a Pade' approximation to $A^{-1}(e^{Ah} - I)B$
P_{sec}	period of characteristic mode, seconds
p_b	roll rate, radians per second
Q	$n \times k$ matrix which is a Pade' approximation to $A^{-2}(e^{Ah} - Ah - I)B$
q_b	pitch rate, radians per second
r_b	yaw rate, radians per second
S	similarity transformation matrix
T	thrust, newtons (ENG in computer-generated tables)
t	time, an independent variable, seconds
t_f	final time, seconds
t_i	starting time, seconds
$t_{1/2}$	time to damp to one-half amplitude, seconds
t_2	time to double amplitude, seconds
u_b	longitudinal translation velocity, meters per second
$u(t)$	k dimensional vector whose elements are control variables of system
V_c	calibrated airspeed, knots
V_{max}	maximum operating airspeed, knots
v_b	lateral translation velocity, meters per second
w_b	vertical translation velocity, meters per second
$x(t)$	n dimensional vector whose elements are state variables of system
x^j	nominal state vector with jth element possibly different from its nominal value
δ	small perturbation of variable away from its nominal value (DEL in computer-generated tables)
δ_a	aileron position, degrees
δ_e	elevator position, degrees

δ_j constant which is amount jth element of nominal state or control vector was perturbed

δ_r rudder position, degrees

δ_s stabilator position, degrees

$\delta_{sp,L}$ flight spoiler position, left side, degrees

$\delta_{sp,R}$ flight spoiler position, right side, degrees

θ pitch attitude, radians (THETA in computer-generated tables)

ξ_{DR} damping coefficient for Dutch roll mode of aircraft

ξ_p damping coefficient for phugoid mode of aircraft

ξ_{SP} damping coefficient for short period mode of aircraft

τ variable of integration

τ_{RS} time constant of roll subsidence mode

τ_{SD} time constant of spiral divergence mode

ϕ roll attitude, radians (PHI in computer-generated tables)

ψ yaw attitude, radians (PSI in computer-generated tables)

Subscripts:

o nominal values of variables

R rotor

A dot over a variable indicates a time (DOT in computer-generated tables).

PROBLEM DESCRIPTION

Aircraft simulated on Langley's real-time simulation system are described by a set of nonlinear simultaneous differential equations of the form

$$\dot{\bar{x}}(t) = \bar{f}\left[\bar{x}(t),\bar{u}(t),t\right] \tag{1}$$

where t represents time, $\bar{x}(t)$ is an n dimensional time-varying state vector, $\bar{u}(t)$ is a k dimensional time-varying control vector, and $\bar{f}$ is an n dimensional vector of general nonlinear functions. As shown by reference 3, if $\bar{u}_o(t)$ is a given input (control) to the system described by equation (1) and $\bar{x}_o(t)$ is a known solution of the state differential equation, one can find approximations to neighboring solutions for small deviations from the initial state and input vectors by using a linear state differential equation. Assume that $\bar{x}_o(t)$ satisfies

$$\dot{\bar{x}}_o(t) = \bar{f}\left[\bar{x}_o(t),\bar{u}_o(t),t\right] \qquad (t_i \geq t \geq t_f)$$

and that the system is operated close to nominal conditions. Therefore, one can write

$$\left.\begin{aligned} \bar{u}(t) &= \bar{u}_o(t) + \delta\bar{u}(t) \\ \\ \bar{x}(t) &= \bar{x}_o(t) + \delta\bar{x}(t) \end{aligned}\right\} \qquad (t_i \geq t \geq t_f) \tag{2}$$

Substituting equations (2) into the state differential equation (eq. (1)) and expanding in a Taylor series about $(\bar{x}_o(t),\bar{u}_o(t))$ yields

$$\dot{\bar{x}}_o(t) + \delta\dot{\bar{x}}(t) = \bar{f}\left[\bar{x}_o(t),\bar{u}_o(t),t\right] + J_x\left[\bar{x}_o(t),u_o(t),t\right]\,\delta\bar{x}(t)$$

$$+ J_u\left[\bar{x}_o(t),\bar{u}_o(t),t\right]\,\delta\bar{u}(t) + \bar{h}(t) \tag{3}$$

where J_x and J_u are the Jacobian matrices of $\bar{f}$ with respect to $\bar{x}$ and $\bar{u}$, respectively. They are given by

$$J_x = A \equiv \left.\frac{\partial\bar{f}}{\partial\bar{x}}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}} \qquad\qquad J_u = B \equiv \left.\frac{\partial\bar{f}}{\partial\bar{u}}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}}$$

The term $\bar{h}(t)$ is the sum of higher order terms from the Taylor expansion and should be "small" with respect to $\delta\bar{x}$ and $\delta\bar{u}$. Neglecting $\bar{h}(t)$, $\delta\bar{x}$ and $\delta\bar{u}$ approximately satisfy the "linear" equation

$$\delta\dot{\bar{x}}(t) = A(t)\ \delta\bar{x}(t) + B(t)\ \delta\bar{u}(t) \tag{4}$$

which is called the linearized state equation. For the particular applications of interest, only the time invariant system is considered in which $A(t)$ and $B(t)$ are constant matrices A and B. Therefore, the linearized state equation can be written as

$$\delta\dot{\bar{x}} = A\delta\bar{x} + B\delta\bar{u} \tag{5}$$

NUMERICAL LINEARIZATION TECHNIQUE

Computation of the A and B Matrices

Now by using equation (1) and by assuming that each component $(f_i(\bar{x},\bar{u},t)$ for $i = 1,n)$ is continuously differentiable m times and can be evaluated m times, the partials required for the A and B matrices can be approximated by using the Lagrange interpolation formulas (refs. 8 and 9). The components of $\bar{f}(\bar{x},\bar{u},t)$ are approximated by

$$f_i(\bar{x}^j, \bar{u}_o) = \sum_{k=1}^{m} \ell_k(x_j)\; f_i(\bar{x}_k{}^j, \bar{u}_o) \qquad (i = 1,n) \tag{6}$$

Due to notation complexity, this formula is explained by an example that uses the three-point Lagrange formula. First,

$$\bar{x}_k{}^j = \left(x_{o_1}, x_{o_2}, \ldots, x_j, \ldots, x_{o_n}\right)$$

which is the nominal state vector with the jth element allowed to vary from its nominal value while all the elements remain fixed. For the three-point formula ($m = 3$), these vectors are

$$\bar{x}_1{}^j = \left(x_{o_1}, x_{o_2}, \ldots, x_{o_j} - \delta_j, \ldots, x_{o_n}\right)$$

$$\bar{x}_2{}^j = \left(x_{o_1}, x_{o_2}, \ldots, x_{o_j}, \ldots, x_{o_n}\right)$$

$$\bar{x}_3{}^j = \left(x_{o_1}, x_{o_2}, \ldots, x_{o_j} + \delta_j, \ldots, x_{o_n}\right)$$

and

$$\ell_1(x_j) = \frac{\left(x_j - x_{o_j}\right)\left[x_j - \left(x_{o_j} + \delta_j\right)\right]}{\left[\left(x_{o_j} - \delta_j\right) - x_{o_j}\right]\left[\left(x_{o_j} - \delta_j\right) - \left(x_{o_j} + \delta_j\right)\right]}$$

$$= \frac{\left(x_j - x_{o_j}\right)\left(x_j - x_{o_j} - \delta_j\right)}{2\delta_j{}^2}$$

$$\ell_2(x_j) = \frac{\left[x_j - \left(x_{o_j} - \delta_j\right)\right]\left[x_j - \left(x_{o_j} + \delta_j\right)\right]}{\left[x_{o_j} - \left(x_{o_j} - \delta_j\right)\right]\left[x_{o_j} - \left(x_{o_j} + \delta_j\right)\right]}$$

$$= \frac{\left(x_j - x_{o_j} + \delta_j\right)\left(x_j - x_{o_j} - \delta_j\right)}{-\delta_j{}^2}$$

$$\ell_3(x_j) = \frac{\left[x_j - \left(x_{o_j} - \delta_j\right)\right]\left(x_j - x_{o_j}\right)}{\left[\left(x_{o_j} + \delta_j\right) - \left(x_{o_j} - \delta_j\right)\right]\left[\left(x_{o_j} + \delta_j\right) - x_{o_j}\right]}$$

$$= \frac{\left(x_j - x_{o_j} + \delta_j\right)\left(x_j - x_{o_j}\right)}{2\delta_j^2}$$

In order to compute $\left.\frac{\partial f_i}{\partial x_j}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}}$ equation (6) must be differentiated with respect to x_j and the resulting equation evaluated at $(\bar{x}_o, \bar{u}_o)$ as follows:

$$a_{ij} \equiv \left.\frac{\partial f_i}{\partial x_j}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}} = \sum_{k=1}^{m} \ell_k'(x_j)\ f_i(\bar{x}_k{}^j, \bar{u}_o) \tag{7}$$

Again using the three-point formula as an example yields

$$\ell_1'(x_j) = \frac{2\left(x_j - x_{o_j}\right) - \delta_j}{2\delta_j^2}$$

$$\ell_2'(x_j) = \frac{-2\left(x_j - x_{o_j}\right)}{\delta_j^2}$$

$$\ell_3'(x_j) = \frac{2\left(x_j - x_{o_j}\right) + \delta_j}{2\delta_j^2}$$

which, when evaluated at the three values of x_j and summed according to equation (7), results in

$$\left.\frac{\partial f_i}{\partial x_j}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}} = \frac{1}{2\delta_j}\left[-f_i\left(\bar{x}_1{}^j, \bar{u}_o\right) + f_i\left(\bar{x}_3{}^j, \bar{u}_o\right)\right] \tag{8}$$

An equivalent result for the five-point differentiation formula is

$$\left.\frac{\partial f_i}{\partial x_j}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}} = \frac{1}{12\delta_j}\left[f_i\left(\bar{x}_1{}^j,\bar{u}_o\right) - 8f_i\left(\bar{x}_2{}^j,\bar{u}_o\right) + 8f_i\left(\bar{x}_4{}^j,\bar{u}_o\right) - f_i\left(\bar{x}_5{}^j,\bar{u}_o\right)\right] \tag{9}$$

and that for the seven-point differentiation formula is

$$\left.\frac{\partial f_i}{\partial x_j}\right|_{\substack{\bar{x}=\bar{x}_o \\ \bar{u}=\bar{u}_o}} = \frac{1}{60\delta_j}\left[-f_i\left(\bar{x}_1{}^j,\bar{u}_o\right) + 9f_i\left(\bar{x}_2{}^j,\bar{u}_o\right) - 45f_i\left(\bar{x}_3{}^j,\bar{u}_o\right) + 45f_i\left(\bar{x}_5{}^j,\bar{u}_o\right)\right.$$

$$\left. - 9f_i\left(\bar{x}_6{}^j,\bar{u}_o\right) + f_i\left(\bar{x}_7{}^j,\bar{u}_o\right)\right] \tag{10}$$

The computation of the B matrix is identical to that of the A matrix except that $\bar{x}$ is held constant and $\bar{u}$ is varied.

Use of the Perturbation Model

Once the A and B matrices have been determined, the perturbation model defined by equation (5) is ready for use by the researcher. The two most common uses of this model are to use it in place of a nonlinear simulation in studies that will be very limited in their area of operation and to determine the eigenvalues of the model about the defined state trajectory. These eigenvalues in aerodynamic problems identify the basic modes of the aircraft; these are Dutch roll, short period, phugoid, spiral divergence, and roll subsidence. In this program, the eigenvalues of the perturbation model (characteristic roots of the A matrix) are determined by a standard Langley library routine.

If the researcher desires to use the perturbation model in place of, or to compare with, his nonlinear model, it will be necessary to integrate equation (5). The solution to equation (5) is

$$\delta\bar{x}(t) = e^{At}\,\delta\bar{x}(o) + \int_0^t e^{A(t-\tau)}\,B\bar{u}(\tau)\,d\tau \tag{11}$$

where the solution to the nonlinear system would be approximated by

$$\bar{x}(t) = \bar{x}(o) + \delta\bar{x}(t)$$

As shown by reference 1, a discrete approximation to equation (11) with local truncation error good to $O(h^3)$ is given by

$$\delta\bar{x}_{k+1} = e^{Ah}\delta\bar{x}_k + P\delta\bar{u}_k + Q\delta\dot{\bar{u}}_k \tag{12}$$

where

$$P = A^{-1}(e^{Ah} - I)B \tag{13}$$

$$Q = A^{-2}(e^{Ah} - Ah - I)B \tag{14}$$

$$\delta\dot{u}_k \equiv \frac{\delta u - \delta u_{k-1}}{h} \tag{15}$$

and as before

$$x_{k+1} = x_k + \delta\bar{x}_{k+1} \tag{16}$$

Equations (12) to (16) are solved by the program's integration subroutine.

PROGRAM USAGE AND LIMITATIONS

For normal application the following should be followed by the user (fig. 1):

(1) Trim the nonlinear aircraft model about the desired trajectory to obtain the nominal state vector x_o and the nominal control vector u_o. The trim algorithm used at Langley Research Center for most real-time simulations is described in reference 10.

(2) Compute the A matrix by using subroutine JACMAT (appendix A) with XNOM = $\bar{x}_o$ and M = N = n (number of states).

(3) Reset the states to their trim values.

(4) Compute the B matrix by using JACMAT with XNOM = $\bar{U}_o$, M = k (number of controls), and N = n.

(5) Reset controls to their trim values.

(6) If eigenvalues are required, the user must call a subroutine which generates eigenvalues. For the applications presented, subroutine REQR, a part of the Langley computer mathematical library, was used.

(7) If integration of the linear system is required, call subroutine COEFF (appendix A) with NDIMA = n and NCOLB = k for calculating the coefficients of δx_k, δu_k, and $\delta\dot{u}_k$.

(8) Obtain response of the linear system to a predetermined input sequence u_k by calling subroutine INTEGRT (appendix A).

When applying this technique to general nonlinear simulations, certain potential problem areas should be mentioned. First, all implicit loops in the nonlinear equations must be broken by substituting variables and by reformulating the equations. If this is not possible, an iterative technique may possibly be used to determine the approximate perturbed steady-state forces and moments. Second, the magnitudes of the perturbations used for the state and control variables need be chosen with care because if the perturbations are too small or too large, the derived linear model will not be a good approximation to the nonlinear analysis. The method used to choose perturbation magnitudes for the NASA Terminal Configured Vehicle (TCV) example is described in appendix B. And third, even though the linearization technique can be applied about any nominal state trajectory, the results are more meaningful when the vehicle is trimmed and the nominal trajectory is stable.

PROGRAM APPLICATION

As an example of the results obtained from a standard application, the TCV airplane, a Boeing 737-100, was chosen. The desired outputs of this application were (1) linearized models of the B-737 about various trim conditions, (2) identification of the basic modes of the aircraft (eigenvalues) at these trim conditions, and (3) time-history comparisons of the linear and nonlinear models for predetermined inputs.

The desired linearized models were of the form

$$\delta\dot{\bar{x}} = A\delta\bar{x} + B\delta\bar{u}$$

where the state vector was chosen to be in the body-axis system. The el of the body-axis state vector $\bar{x}_b$ are

$$\bar{x}_b = \begin{bmatrix} u_b \\ w_b \\ q_b \\ \theta \\ v_b \\ p_b \\ r_b \\ \phi \\ \psi \end{bmatrix}$$

and the elements of the control vector are

$$\bar{u} = \begin{bmatrix} T \\ \delta_s \\ \delta_r \\ \delta_e \\ \delta_a \\ \delta_{sp,L} \\ \delta_{sp,R} \end{bmatrix}$$

Tables I to V are example outputs and show the nominal state and control vectors, the body-axis A and B matrices, the eigenvalues, and the corresponding eigenvectors.

Linear models defined in other axis systems can be derived from this model by means of a similarity transformation S where

$$\bar{x}_b = S\bar{x}_D$$

and

$$\dot{\bar{x}}_b = S\dot{\bar{x}}_D$$

with $\bar{x}_D$ being the desired state vector defined in the new axis system and S being time invariant. Substituting into our linear model

$$\delta\dot{\bar{x}}_b = A\delta\bar{x}_b + B\delta\bar{u}$$

yields

$$S\delta\dot{\bar{x}}_D = AS\delta\bar{x}_D + B\delta\bar{u}$$

or

$$\delta\dot{\bar{x}}_D = S^{-1}AS\delta\bar{x}_D + S^{-1}B\delta\bar{u}$$

as our linear model in the desired axis system.

To further show the usefulness of these linear models as a simulation verification and validation tool for the various flight conditions shown in table VI, a comparison of independent Boeing data (unpublished) and the linear models generated from the nonlinear simulation is shown in tables VII and VIII. A review of these tables will show that good agreement exists between the simulation models and the independent data in almost all cases, excluding the spiral divergence mode. However, major disagreements do exist in the short period mode of condition V with the aft center of gravity and in the phugoid mode of condition VII with the forward center of gravity. The researcher should now take steps to resolve the reasons for the differences in these cases by first reverifying the implementation of the nonlinear simulations' aerodynamics data in these areas and by trying to obtain other independent data such as flight data.

Additional insights into the system being simulated can be gained by comparing the linear models generated by each Lagrange interpolation formula. A general indication of the linearity of the simulation about the nominal trajectory is obtained, as well as an indication of sensitive modes and parameters (nonlinearities) of the simulation. For example, a comparison of the models obtained for the maximum speed case (table VI, condition V) showed that the lateral and the short period modes were approximately linear, but for the phugoid mode, the damping ratio varied by 36 percent and the natural frequency by 3 percent. This information implies that the linear models obtained would not be suitable for studies requiring precise knowledge of the phugoid mode.

The numerical linearization technique has also been successfully applied to nonlinear simulations of other aircraft. Linear models of a fighter aircraft, a general aviation aircraft, a standard rotorcraft, and the rotor systems research aircraft (RSRA) developed by NASA and the U.S. Army (ref. 11) have been obtained about various trim conditions. The standard procedure as described was used in all cases except that of the RSRA aircraft which required procedural modifications since the nonlinear simulation included a dynamic rotor model which was continuously rotating during the linearization. Figure 2 outlines the iterative technique used to obtain the linear models for this vehicle. Basically, the approach taken was to allow integration of the rotor dynamics. However, a steady-state condition had to be obtained after each perturbation of a state or control before numerically calculating the Jacobians. Averaging the forces and moments over a number of rotor revolutions and at various points during each revolution is also done to enhance the credibility of the linear model obtained.

CONCLUDING REMARKS

The numerical linearization technique described in this paper has been successfully applied to nonlinear simulations of various aircraft. At this writing, linear models of the NASA Terminal Configured Vehicle, a fighter aircraft, a general aviation aircraft, a standard rotorcraft, and the RSRA have been obtained about various trim conditions. Linear models of aircraft with stability augmentation systems have also been obtained by augmenting the state vector with the associated automatic control system states and by proceeding in the manner described in the paper.

A modification of the technique for application to simulations of rotorcraft with dynamic rotor models has also been developed and described.

Langley Research Center
National Aeronautics and Space Administration
Hampton, VA 23665
May 18, 1978

APPENDIX A

DESCRIPTION AND LISTINGS OF SUBROUTINES JACMAT, COEFF, AND INTEGRT

The major portion of the linear analysis package consists of three subroutines, JACMAT, COEFF, and INTEGRT.

Subroutine JACMAT

Purpose: Calculates the Jacobian of the nonlinear system about a nominal point in vector space. This corresponds to the A matrix when the input argument is the state vector and the B matrix when the input argument is the control vector.

Use: CALL JACMAT (XNOM,F,FVAL,JACOBN,DELTA,N,M,EOM,IPNTS,MAXROW,MAXCOL) where

XNOM — An N dimensional input vector; this vector contains the nominal values of the independent variables about which the Jacobian is calculated

F — An N dimensional output vector; during computation of the partial derivatives, it contains the values of the dependent variables

FVAL — An N × M × IPNTS dimensional array used in calculating the partial derivatives; FVAL(I,J,K) is the Ith component of $\bar{F}$ evaluated at the Kth change in the Jth component of XNOM

JACOBN — An N × M dimensional output array which is the Jacobian matrix evaluated at XNOM; that is,

$$\text{JACOBN}(I,J) = \left.\frac{\partial f_i}{\partial x_j}\right|_{\bar{x}=\text{XNOM}}$$

DELTA — An N dimensional input vector to step sizes; DELTA(I) is the increment for XNOM(I)

N — An integer input specifying the number of equations

M — An integer input specifying the number of independent variables

EOM — A user-supplied subroutine which calculates the values of F used in computing FVAL; EOM is a subroutine in the parameter list of JACMAT; the statement

EXTERNAL EOM

must be included in the calling program of JACMAT; the calling statement for EOM is

APPENDIX A

CALL EOM (N,M,XNOM,F)

where N, M, and XNOM are inputs and F is the output

IPNTS An integer input which specifies the interpolation formula to be used:

INPTS = 2, three-point formula

INPTS = 4, five-point formula

INPTS = 6, seven-point formula

MAXROW An integer input specifying the maximum number of equations to be used

MAXCOL An integer input specifying the maximum number of independent variables to be used

The listing of subroutine JACMAT is as follows.

```
      SUBROUTINE JACMAT(XNOM,F,FVAL,JACOBN,DELTA,N,M,EQM,IPNTS,MAXROW,   JACMAT
     1                  MAXCOL)                                           JACMAT
                                                                          JACMAT
      DIMENSION DELTA(MAXCOL),F(MAXROW),FVAL(MAXROW,MAXCOL,IPNTS),R(6),   JACMAT
     1          S(6),XNOM(MAXCOL)                                         JACMAT
      REAL JACOBN(MAXROW,MAXCOL)                                          JACMAT
      DATA R,S/12*0./                                                     JACMAT
                                                                          JACMAT
C         THIS SUBROUTINE USES AN (IPNTS+1)-POINT FORMULA EVALUATED AT   JACMAT
C     THE CENTRAL POINT TO APPROXIMATE THE PARTIALS. THE PARAMETER,       JACMAT
C     IPNTS, MAY TAKE ON THE VALUES 2, 4, OR 6.                           JACMAT
C         DELTA IS AN ARRAY OF STEP SIZES WHICH ARE USED IN COMPUTING     JACMAT
C     THE PARTIALS.                                                       JACMAT
C         JACOBN IS THE JACOBIAN MATRIX.                                  JACMAT
C         XNOM  IS A VECTOR OF INDEPENDENT VARIABLES AND IS THE POINT     JACMAT
C     AT WHICH THE PARTIALS ARE CALCULATED.
C         N IS THE NUMBER OF EQUATIONS.
C         M IS THE NUMBER OF UNKNOWNS.
C         F IS AN ARRAY USED TO STORE CALCULATED VALUES OF THE FUNCTIONS
C         FVAL IS AN ARRAY USED IN COMPUTING THE PARTIALS FOR JACOBN.

      IF((IPNTS.LT.1).OR.(IPNTS.GT.6)) GO TO 70
      GO TO (70,2,70,4,70,6),IPNTS
    2 CONTINUE

C         CONSTANTS USED IN THE 3-POINT FORMULA

           DIV    = 2.
           R(1)   =-1.
           R(2)   = 2.
           S(1)   =-1.
           S(2)   = 1.
      GO TO 9
    4 CONTINUE
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      CO STANTS USED IN THE 5-POINT FORMULA

       DIV    = 12.
       R(1)   =-2.
       R(2)   = R(4) = 1.
       R(3)   = 2.
       S(1)   = 1.

       S(2)   =-8.
       S(3)   = 8.
       S(4)   =-1.
    GO TO 9
  6 CONTINUE

      CONSTANTS USED IN THE 7-POINT FORMULA

       DIV    = 60.
       R(1)   =-3.
       R(2)   = R(3) = R(5) = R(6) = 1.
       R(4)   = 2.
       S(1)   =-1.
       S(2)   = 9.
       S(3)   =-45.
       S(4)   = 45.
       S(5)   =-9.
       S(6)   = 1.
  9 CONTINUE

      HERE THE ARRAY FVAL IS COMPUTED.  FVAL(I,J,K) IS THE ITH
    FUNCTION EVALUATED AT THE KTH CHANGE IN THE JTH VARIABLE.

    DO 30 J=1,M
         XNMSAV = XNOM(J)
    DO 20 K=1,IPNTS
         XNOM(J) = XNOM(J) + R(K)*DELTA(J)
    CALL EOM(N,M,XNOM,F)
    DO 10 I=1,N
         FVAL(I,J,K) = F(I)/(DIV*DELTA(J))
 10 CONTINUE
```

```
20 CONTINUE
        XNOM(J) = XNMSAV
30 CONTINUE

      AT THIS POINT THE ARRAY JACOBN IS COMPUTED.

   DO 60 I=1.N
   DO 50 J=1.M
        JACOBN(I.J) = 0.
   DO 40 K=1.IPNTS
        JACOBN(I.J) = JACOBN(I.J) + S(K)*FVAL(I.J.K)
40 CONTINUE
50 CONTINUE
60 CONTINUE
   RETURN
70 CONTINUE
   RETURN
   END
```

APPENDIX A

Subroutine COEFF

Purpose: Computes the coefficients (e^{Ah}, P, and Q) required for calculation of the discrete approximation to the solution of perturbation model.

Use: CALL COEFF (A,NDIMA,B,NCOLB,H,EAH,P,Q,W,MAXDIMA,MAXCOLB), where

A	An NDIMA $\times$ NDIMA dimensional input array; this array is the A matrix of the perturbation model
NDIMA	An integer input specifying the dimension of A
B	An NDIMA $\times$ NCOLB dimensional input array; this array is the B matrix of the perturbation model
NCOLB	An integer input specifying the number of columns of B
H	Length of the integration interval
EAH	An NDIMA $\times$ NDIMA dimensional output array which approximates e^{Ah}
P	An NDIMA $\times$ NCOLB dimensional output array which approximates $A^{-1}(e^{Ah} - I)B$
Q	An NDIMA $\times$ NCOLB dimensional output array which approximates $A^{-2}(e^{Ah} - Ah - I)B$
W	An NDIMA $\times$ NDIMA dimensional working space array
MAXDIMA	An integer input specifying the maximum dimension of A
MAXCOLB	An integer input specifying the maximum number of columns of B

The listing of subroutine COEFF is as follows:

```
      SUBROUTINE COEFF(A,NDIMA,B,NCOLB,H,EAH,P,Q,W,MAXDIMA,MAXCOLB)



          THIS SUBROUTINE CALCULATES THE MATRIX COEFFICIENTS USED IN
      APPROXIMATING THE SOLUTION OF XDOT = AX + BU.  HERE EAH = E**(A*H)
      AND EAH IS APPROXIMATED BY (I + A*H/2)*((I - A*H/2)**(-1)).  HERE
      P = A**(-1)*(E**(A*H) - I)*B AND P IS APPROXIMATED BY
      H*((I - A*H/2)**(-1))*B.  HERE Q = A**(-2)*(E**(A*H) - A*H - I)*B
      AND Q IS APPROXIMATED BY (H**2/2)*(I + A*H/3)*B.  H IS THE LENGTH
      OF THE INTERVAL.

      DIMENSION A(MAXDIMA,MAXDIMA),B(MAXDIMA,MAXCOLB),
     1          EAH(MAXDIMA,MAXDIMA),P(MAXDIMA,MAXCOLB),
     2          Q(MAXDIMA,MAXCOLB),W(MAXDIMA,MAXDIMA),KARRAY(7)

          SET UP CONSTANTS.

            H2     = H*.5
            H3     = H/3.
            H22    = (H*H)*.5
            H36    = (H**3)/6.

      DO 10 I=1,NDIMA
      DO 10 J=1,NDIMA

          COMPUTE I+A*H/2 AND CALL THE RESULT EAH.

            EAH(I,J) = H2*A(I,J)
      IF(I.EQ.J) EAH(I,J) = 1. + EAH(I,J)

          COMPUTE I-A*H/2 AND CALL THE RESULT W.

            W(I,J) =-H2*A(I,J)
      IF(I.EQ.J) W(I,J) = 1. + W(I,J)
   10 CONTINUE
```

```
                                                                  COEFF
     INVERT I-A*H/2 AND CALL THE RESULT W.                        COEFF
                                                                  COEFF
       KARRAY(1) = 10                                             COEFF
       KARRAY(2) = KARRAY(3) = NDIMA                              COEFF
       KARRAY(5) = MAXDIMA                                        COEFF

       KARRAY(4) = KARRAY(6) = KARRAY(7) = 0                      COEFF
   CALL MATOPS(KARRAY,W,DET,DUM)                                  COEFF
                                                                  COEFF
     COMPUTF (I + A*H/2)*((I - A*H/2)**(-1)) AND CALL THE RFSULT  COEFF
   EAH.                                                           COEFF
                                                                  COEFF
       KARRAY(1) = 20                                             COEFF
       KARRAY(4) = NDIMA                                          COEFF
       KARRAY(6) = KARRAY(7) = MAXDIMA                            COEFF
   CALL MATOPS(KARRAY,EAH,W,EAH)                                  COEFF
                                                                  COEFF
   COMPUTE H*((I - A*H/2)**(-1))*B AND CALL THF RFSULT P.         COEFF
                                                                  COEFF
       KARRAY(4) = NCOLB                                          COEFF
   CALL MATOPS(KARRAY,W,B,P)                                      COEFF
   DO 20 I=1,NDIMA                                                COEFF
   DO 20 J=1,NCOLB                                                COEFF
20     P(I,J) = H*P(I,J)                                          COEFF
                                                                  COEFF
     COMPUTE (H**2/2)*(I + A*H/3)*B AND CALL THE RESULT Q.        COEFF
                                                                  COEFF
   DO 30 I=1,NDIMA                                                COEFF
   DO 30 J=1,NDIMA                                                COEFF
       W(I,J) = H36*A(I,J)                                        COEFF
   IF(I.EQ.J) W(I,J) = H22 + W(I,J)                               COEFF
30 CONTINUF                                                       COEFF
   CALL MATOPS(KARRAY,W,B,Q)                                      COEFF
   RETURN                                                         COEFF
   END                                                            COEFF
```

APPENDIX A

Subroutine INTEGRT

Purpose: Generates solutions to the linear differential equations obtained from the nonlinear simulation.

Use: CALL INTEGRT (N,X,XO,XODOTH,L,U,UO,UDOT,EAH,P,Q,W1,MAXN,MAXL), where

N — An integer input specifying the number of states being used; $N \leq MAXN$

X — A MAXN-dimensional input/output vector which contains the values of the states in its first N locations; on input, X contains the past values of the states, and on output, it contains the current values of the states

XO — A MAXN-dimensional input vector that contains the values of the states at which the A and B matrices were calculated in its first N locations; that is, XO contains $\bar{X}_O$

XODOTH — A MAXN-dimensional input vector which contains C*H in its first N locations where C is the value of $\bar{f}(\bar{x}_O,\bar{u}_O)$ and H is the same as in COEFF

L — An integer input specifying the number of controls being used; $L \leq MAXL$

U — A MAXL-dimensional input vector which contains the current values of the controls in its first L locations

UO — A MAXL-dimensional input vector that contains the values of the controls at which the A and B matrices were calculated in its first L locations; that is, UO contains $\bar{U}_O$

UDOT — A MAXL-dimensional input vector which contains the time derivatives of the controls in its first L locations

EAH — A MAXN × MAXN-dimensional input array; this is the same EAH as in COEFF

P — A MAXN × MAXL-dimensional input array; this is the same P as in COEFF

Q — A MAXN × MAXL-dimensional input array; this is the same Q as in COEFF

W1 — A MAXN-dimensional vector used for working space

MAXN — An integer input specifying the maximum number of states

MAXL — An integer input specifying the maximum number of controls

The listing of subroutine INTEGRT is as follows:

```
      SUBROUTINE INTEGRT(N,X,X0,X0DOTH,L,U,U0,UDOT,EAH,P,Q,W1,MAXN,MAXL)INTEGRT
C                                                                       INTEGRT
C         THIS SUBROUTINE APPROXIMATES THE SOLUTION TO THE DIFFERENTIAL INTEGRT
C     EQUATION XDOT = F(X,U). FIRST,THE DIFFERENTIAL EQUATION           INTEGRT
C     DELXDOT = A*DELX + B*DELU + F(X0,U0)  IS SOLVED.  THE SOLUTION TO INTEGRT
C     THIS DIFFERENTIAL EQUATION IS GIVEN BY  DELX(K+1) = EAH*DELX(K) + INTEGRT
C     P*DELU(K) + Q*UDOT(K) + XDOT*H. THEN, THE SOLUTION TO  XDOT =     INTEGRT
C     F(X,U)  IS GIVEN BY  X(K+1) = DELX(K+1) + X0.                     INTEGRT
C                                                                       INTEGRT
      DIMENSION EAH(MAXN,MAXN),P(MAXN,MAXL),Q(MAXN,MAXL),U(MAXL),       INTEGRT
     1          UDOT(MAXL),U0(MAXL),W1(MAXN),X(MAXN),X0(MAXN),          INTEGRT
     2          KARRAY(7),X0DOTH(MAXN)                                  INTEGRT
C                                                                       INTEGRT
C         CALCULATE  DELX = X - X0  AND CALL THE RESULT X.              INTEGRT
C                                                                       INTEGRT
      DO 1 I=1,N                                                        INTEGRT
    1    X(I)    = X(I) - X0(I)                                         INTEGRT
C                                                                       INTEGRT
C         CALCULATE  DELU = U - U0 AND CALL THE RESULT U.               INTEGRT
C                                                                       INTEGRT
      DO 2 I=1,L                                                        INTEGRT
    2    U(I)    = U(I) - U0(I)                                         INTEGRT
C                                                                       INTEGRT
C         CALCULATE  EAH*DELX  AND CALL THE RESULT X.                   INTEGRT
           KARRAY(1) = 20                                               INTEGRT
           KARRAY(4) = 1                                                INTEGRT
           KARRAY(2) = KARRAY(3) = N                                    INTEGRT
           KARRAY(5) = KARRAY(6) = KARRAY(7) = MAXN                     INTEGRT
      CALL MATOPS(KARRAY,EAH,X,X)                                       INTEGRT
C                                                                       INTEGRT
C         CALCULATE P*DELU  AND CALL THE RESULT W1.                     INTEGRT
C                                                                       INTEGRT
           KARRAY(3) = L                                                INTEGRT
           KARRAY(6) = MAXL                                             INTEGRT
      CALL MATOPS(KARRAY,P,U,W1)                                        INTEGRT
```

```
      DO 3 I=1,N                                                 INTEGRT  36
3        X(I)      = X(I) + W1(I)                                INTEGRT  37
                                                                 INTEGRT  38
C        CALCULATE  Q*UDOT  AND CALL THE RESULT W1.              INTEGRT  39
                                                                 INTEGRT  40
      CALL MATOPS(KARRAY,Q,UDOT,W1)                              INTEGRT  41
                                                                 INTEGRT  42
                                                                 INTEGRT  43
      DO 4 I=1,N                                                 INTEGRT  44
4        X(I)      = X(I) + W1(I) + X0(I) + X0DOTH(I)            INTEGRT  45
      DO 5 I=1,L                                                 INTEGRT  46
5        U(I)      = U(I) + U0(I)                                INTEGRT  47
                                                                 INTEGRT  48
       RETURN                                                    INTEGRT  49
       END                                                       INTEGRT  50
```

ORIGINAL PAGE IS
OF POOR QUALITY

APPENDIX B

METHOD OF CHOOSING PERTURBATION MAGNITUDES

The magnitudes of the perturbations used for the state variables in the TCV example were chosen as a function of aircraft states in which the aeronautical engineer would have some intuitive feel as to their desired range of variation. For this example, the variables used were total velocity V_T, angle of attack α, angle of sideslip β, roll attitude ϕ, pitch attitude θ, and yaw attitude ψ. The values of the variations in the body-axis state variables are given by

$$\Delta u_b = \Delta V_T \frac{u_b}{V_T} - w_b \ \Delta\alpha - v_b \cos\alpha \ \Delta\beta$$

$$\Delta w_b = \Delta V_T \frac{w_b}{V_T} + u_b \ \Delta\alpha - v_b \sin\alpha \ \Delta\beta$$

$$\Delta q_b = \Delta\hat{a} \sin\phi \cos\theta + r_b \ \Delta\phi + p_b \sin\phi \ \Delta\theta$$

$$\Delta v_b = \Delta V_T \frac{v_b}{V_T} + V_T \cos\beta \ \Delta\beta$$

$$\Delta p_b = -\Delta\hat{a} \sin\theta - \hat{a} \cos\theta \ \Delta\beta$$

$$\Delta r_b = \Delta\hat{a} \cos\phi \cos\theta - q_b \ \Delta\phi + p_b \cos\phi \ \Delta\theta$$

where

$$\hat{a} = \frac{g \tan\phi}{V_T}$$

$$\Delta\hat{a} = \frac{g \ \Delta\phi}{V_T \cos^2\phi} - \frac{a}{V_T} \Delta V_T$$

APPENDIX B

$\Delta V_T = 0.01 V_T$

$\Delta\alpha = 0.2/57.3$ rad

$\Delta\beta = 0.1/57.3$ rad

$\Delta\theta = 1./57.3$ rad

$\Delta\phi = 1./57.3$ rad

$\Delta\psi = 1./57.3$ rad

It should be noted that all variables except ϕ are at these trim values. The magnitude of ϕ was not allowed to be less than 2/57.3 rad so that a nonzero value for Δq_b would be calculated.

The variations in the control variables were 1 percent of the total range of each control variable.

REFERENCES

1. Barker, Lawrence E., Jr.; Bowles, Roland L.; and Williams, Louise H.: Development and Application of a Local Linearization Algorithm for the Integration of Quaternion Rate Equations in Real-Time Flight Simulation Problems. NASA TN D-7347, 1973.

2. Wilkinson, J. H.; and Reinsch, C.: Handbook for Automatic Computation. Volume II - Linear Algebra. Springer-Verlag, 1971.

3. Kwakernaak, Huibert; and Sivan, Raphael: Linear Optimal Control Systems. John Wiley & Sons, Inc., c.1972.

4. Bryson, Arthur E., Jr.; and Ho, Yu-Chi: Applied Optimal Control. Ginn and Co., c.1969.

5. White, John S.; and Lee, Homer Q.: Users Manual for the Variable Dimension Automatic Synthesis Program (VASP). NASA TM X-2417, 1971.

6. Edwards, John W.: A FORTRAN Program for the Analysis of Linear Continuous and Sampled-Data Systems. NASA TM X-56038, 1976.

7. Melsa, James L.; and Jones, Stephen K.: Computer Programs for Computational Assistance in the Study of Linear Control Theory. Second ed. McGraw-Hill Book Co., c.1973.

8. Hildebrand, F. B.: Introduction to Numerical Analysis. McGraw-Hill Book Co., Inc., 1956.

9. Fröberg, Carl-Erik: Introduction to Numerical Analysis. Addison-Wesley Pub. Co., Inc., c.1965.

10. Houck, Jacob A.; Gibson, Lucille H.; and Steinmetz, George G.: A Real-Time Digital Computer Program for the Simulation of a Single-Rotor Helicopter. NASA TM X-2872, 1974.

11. Houck, Jacob A.; Moore, Frederick L.; Howlett, James J.; Pollock, Kenneth S.; and Browne, Mary M.: Rotor Systems Research Aircraft Simulation Mathematical Model. NASA TM-78629, 1977.

TABLE I.- VALUES OF THE STATES, CONTROLS, AND STATE DERIVATIVES AT THE POINT AT WHICH THE PERTURBATION MODEL WAS GENERATED

[Aircraft weight, 36 287.4 kg; altitude, 457.2 m; airspeed, 63.09 m/sec; flap deflection, 40°; landing gear down]

STATE		STATE DERIVATIVE		CONTROL	
UB	= 63.04486	UBDOT	= 0.00001	ENG	= 35339.07894
WB	= 2.8632	WBDOT	= -0.00019	STAB	= 8.56032
QB	= 0.	QBDOT	= -0.00008	DELR	= 0.
THETA	= -0.00698	THETADOT	= 0.	DELE	= 2.66833
VB	= 0.	VBDOT	= 0.	DELA	= 0.
PB	= 0.	PBDOT	= 0.	SPL	= 0.
RB	= 0.	RBDOT	= 0.	SPR	= 0.
PHI	= 0.	PHIDOT	= 0.		
PSI	= 0.	PSIDOT	= 0.		

TABLE II.- THE A MATRIX

[Aircraft weight, 36 287.4 kg; altitude, 457.2 m; airspeed, 63.09 m/sec; flap deflection, 40°; landing gear down]

UBDOT m/sec/sec	-0.03781	0.11139	-2.86186	-9.80664	0.	0.	0.	0.	0.
WBDOT m/sec/sec	-0.28511	-0.72225	63.01435	0.07896	0.	0.	0.	0.	0.
QBDOT rad/sec/sec	-0.00055	-0.01971	-0.50162	-0.00032	0.	0.	0.	0.	0.
THETADOT rad/sec	0.	0.	1.0	0.	0.	0.	0.	0.	0.
VBDOT m/sec/sec	0.	0.	0.	0.	-0.14763	3.39181	-62.73394	9.80633	0.
PBDOT rad/sec/sec	0.	0.	0.	0.	-0.06835	-1.88160	0.99852	0.00003	0.
RBDOT rad/sec/sec	0.	0.	0.	0.	0.01064	-0.14972	-0.14574	-0.00452	0.
PHIDOT rad/sec	0.	0.	0.	0.	0.	1.0	-0.0069755	0.	0.
PSIDOT rad/sec	0.	0.	0.	0.	0.	0.	1.0	0.	0.
	UB m/sec	WB m/sec	QB rad/sec	THETA rad	VB m/sec	PB rad/sec	RB rad/sec	PHI rad	PSI rad

TABLE III.- THE B MATRIX

[Aircraft weight, 36 287.4 kg; altitude, 457.2 m; airspeed, 63.09 m/sec; flap deflection, 40°; landing gear down]

	THRUST N	STAB deg	DELR deg	DELE deg	DELA deg	SPL deg	SPR deg
UBDOT m/sec/sec	0.00003	0.00458	0.	0.00220	0.	-0.00252	-0.00252
WBDOT m/sec/sec	0.	-0.10089	0.	-0.04851	0.	0.02570	0.02570
QBDOT rad/sec/sec	0.00001	-0.04098	0.	-0.01972	0.	0.00097	0.00097
THETADOT rad/sec	0.	0.	0.	0.	0.	0.	0.
VBDOT m/sec/sec	0.	0.	0.04154	0.	0.00026	-0.00383	0.00383
PBDOT rad/sec/sec	0.	0.	0.01084	0.	0.01946	0.01156	-0.01156
RBDOT rad/sec/sec	0.	0.	-0.01102	0.	0.00163	0.00141	-0.00141
PHIDOT rad/sec	0.	0.	0.	0.	0.	0.	0.
PSIDOT rad/sec	0.	0.	0.	0.	0.	0.	0.

TABLE IV.- EIGENVALUES OF SYSTEM

[Aircraft weight, 36 287.4 kg; altitude, 457.2 m; airspeed, 63.09 m/sec; flap deflection, 40°; landing gear down]

EIGENVALUES	TIME CONSTANT	DAMPING RATIO	UNDAMPED NATURAL FREQUENCY	PERIOD	$t_{1/2}$
-.2016E+01 + 0. *I	.4960E+00				.3438E+00
-.5940E-02 + 0. *I	.1684E+03				.1167E+03
0. + 0. *I					
-.1635E-01 + .1778E+00*I		.9161E-01	.1785E+00	.3[illegible]35E+02	.4238E+02
-.1635E-01 + -.1778E+00*I					
-.7636E-01 + .1138E+01*I		.6694E-01	.1141E+01	.5520E+01	.9077E+01
-.7636E-01 + -.1138E+01*I					
-.6145E+00 + .1110E+01*I		.4845E+00	.1268E+01	.5663E+01	.1128E+01
-.6145E+00 + -.1110E+01*I					

ERROR RETURN FROM REQR = 0

TABLE V.- THE A MATRIX EIGENVECTORS

[Aircraft weight, 36 287.4 kg; altitude, 457.2 m; airspeed, 63.09 m/sec; flap deflection, 40°; landing gear down]

.92388E-13	-.77217E-11	0.	-.33790E+00	-.93419E+00	-.66972E-12	-.27383E-12	-.16303E-01	.15728E-01
-.31828E-11	.30459E-11	0.	.22215E-01	.11216E+00	-.81036E-11	.50657E-11	-.87648E+00	.48086E+00
.25839E-13	-.10793E-15	0.	-.48066E-03	-.89016E-03	-.54591E-13	-.29485E-13	-.30577E-02	-.44365E-02
-.12815E-13	.18171E-13	0.	-.47189E-02	.31381E-02	-.22585E-13	.49478E-13	-.18919E-02	.38035E-02
-.99308E+00	.28896E+00	0.	.31229E-15	-.26995E-14	-.65558E+00	-.75499E+00	-.23761E-15	-.12345E-14
-.10499E+00	-.18419E-03	0.	-.10451E-16	.24668E-16	.89103E-02	.46858E-02	-.10877E-16	.59110E-16
-.65566E-02	.56815E-02	0.	-.33435E-16	-.70731E-16	-.29104E-02	.28763E-02	-.10735E-16	-.75311E-18
.52050E-01	.37683E-01	0.	.76340E-16	.85714E-16	.35568E-02	-.80849E-02	.45508E-16	-.14068E-16
.32519E-02	-.95658E+00	.10000E+01	-.37741E-15	.22281E-15	.26866E-02	.23768E-02	.35810E-17	.76917E-17

TABLE VI.- BASIC FLIGHT CONDITIONS

Condition	Description	Weight, kg	Center of gravity		Flap deflection	Gear	Altitude, m	Mach number	V_c, knots
			Forward	Aft					
I	Approach	[a]40 816.3	$0.10\bar{c}$	$0.31\bar{c}$	40°	Down	0	0.185	122
II	Holding[b]				Up	Up	1 254	.382	230
III	Maximum dynamic pressure						3 962	.831	440
IV	Climb[c]						3 048	.622	340
V	Maximum speed and Mach number						6 096	.900	403
VI	Cruise						6 096	.735	330
VII	V_{max}/M_{max} corner						6 102	.840	350
VIII	Maximum altitude cruise						10 058	.742	250
IX	Lightweight V_{max}/M_{max} corner	[d]31 746.0		$0.33\bar{c}$			7 102	.789	330

[a]For weight of 40 823 kg

I_{XX} = 508 432 kg-m^2

I_{YY} = 1 186 340 kg-m^2

I_{ZZ} = 1 626 981 kg-m^2

I_{XZ} = 105 754 kg-m^2

[b]Chosen as being 10 percent above maximum L/D speed.

[c]Minimum cost climb.

[d]For weight of 31 751 kg.

TABLE VII.- CHARACTERISTIC MODES FROM BOEING DATA

Condition	Center of gravity	Short period			Phugoid			Dutch roll		
		ξ_{SP}	P_{sec}	$t_{1/2}$	ξ_p	P_{sec}	$t_{1/2}$	ξ_{DR}	P_{sec}	$t_{1/2}$
I	0.1	0.418	5.39	1.29	0.084	34	45	0.057	4.81	9.31
	0.3	0.583	8.58	1.32	0.130	48	40	0.047	5.07	11.99
II	0.1	0.361	2.86	0.82	0.030	62	225	0.117	3.34	3.12
	0.3	0.490	4.22	0.[illegible]3	0.028	71	273	0.109	3.58	3.59
III	0.1	0.426	1.96	0.46	0.640	118	16	0.125	1.85	1.62
	0.3	0.743	4.67	0.46	0.263	44	18	0.121	2.00	1.82
IV	0.1	0.354	1.97	0.57	0.065	109	185	0.102	2.49	2.68
	0.3	0.499	3.04	0.58	0.086	147	188	0.093	2.70	3.20
V	0.1	0.419	2.40	0.57	$t_{1/2} = 5.1$; $t_2 = 24.9$[a]			0.134	2.12	1.73
	0.3	0.924	12.70	0.58	$t_{1/2} = 4.4$; $t_2 = 17.1$[a]			0.132	2.30	1.91
VI	0.1	0.333	1.86	0.58	0.134	155	127	0.089	2.32	2.86
	0.3	0.484	2.89	0.59	$t_{1/2} = 80.3$; $t_2 = 358.1$[a]			0.086	2.51	3.19
VII	0.1	0.366	2.28	0.64	0.925	273	12	0.097	2.21	2.49
	0.3	0.645	5.00	0.65	0.482	60	12	0.092	2.37	2.83
VIII	0.1	0.285	2.45	1.01	0.103	86	92	0.083	3.08	4.10
	0.3	0.371	3.76	1.04	0.099	87	96	0.079	3.30	4.60
IX	0.1	0.401	2.18	0.55	$t_{1/2} = 17.3$; $t_2 = 7.2$[a]			0.109	2.10	2.11
	0.3	0.739	5.54	0.56	0.451	50	11	0.097	2.25	2.56

[a]Complex conjugate pair splits into two simple poles.

TABLE VII.- Concluded

Condition	Center of gravity	Spiral divergence			Roll subsidence		
		Root	τ_{SD}	$t_{1/2}$ or t_2	Root	τ_{RS}	$t_{1/2}$ or t_2
I	0.1	---	---	19	---	0.531	---
	0.3	---	---	24	---	0.531	---
II	0.1	---	---	124	---	0.433	---
	0.3	---	---	182	---	0.431	---
III	0.1	---	---	131	---	0.346	---
	0.3	---	---	131	---	0.345	---
IV	0.1	---	---	81	---	0.326	---
	0.3	---	---	94	---	0.324	---
V	0.1	---	---	829	---	0.381	---
	0.3	---	---	659	---	0.380	---
VI	0.1	---	---	19	---	0.361	---
	0.3	---	---	20	---	0.359	---
VII	0.1	---	---	57	---	0.454	---
	0.3	---	---	56	---	0.452	---
VIII	0.1	---	---	11	---	0.573	---
	0.3	---	---	11	---	0.572	---
IX	0.1	---	---	91	---	0.358	---
	0.3	---	---	95	---	0.356	---

TABLE VIII.- CHARACTERISTIC MODES FROM LINEAR ANALYSIS OUTPUT

Condition	Center of gravity	Short period			Phugoid			Dutch roll		
		ξ_{SP}	P_{sec}	$t_{1/2}$	ξ_p	P_{sec}	$t_{1/2}$	ξ_{DR}	P_{sec}	$t_{1/2}$
I	0.1	0.399	5.17	1.31	0.071	34	52	0.053	4.82	10.06
	0.3	0.585	11.73	1.79	0.117	30	19	0.040	5.06	14.11
II	0.1	0.368	2.90	0.81	0.036	71	219	0.163	3.65	2.44
	0.3	0.494	4.26	0.83	0.038	84	241	0.160	3.93	2.68
III	0.1	0.430	2.02	0.47	0.418	63	15	0.121	1.85	1.67
	0.3	0.742	4.77	0.47	0.274	37	14	0.129	1.93	1.64
IV	0.1	0.357	2.00	0.58	0.081	117	160	0.136	2.51	2.01
	0.3	0.495	3.04	0.59	0.097	144	163	0.134	2.69	2.21
V	0.1	0.419	2.41	0.57	$t_{1/2}$ = 13.7; t_2 = 117[a]			0.135	2.13	1.73
	0.3	0.695	4.74	0.54	-0.024	28	t_2 = 1263	0.099	2.72	2.56
VI	0.1	0.323	1.98	0.64	0.100	146	161	0.117	2.48	2.33
	0.3	0.455	3.03	0.65	0.218	334	166	0.114	2.65	2.55
VII	0.1	0.361	2.28	0.65	0.326	56	18	0.113	2.21	2.14
	0.3	0.597	4.51	0.67	0.180	36	15	0.109	2.35	2.35
VIII	0.1	0.264	2.51	1.01	0.094	117	137	0.100	1.99	3.49
	0.3	0.374	3.76	1.03	0.109	150	152	0.097	3.38	3.84
IX	0.1	0.358	1.99	0.571	$t_{1/2}$ = 19.2; t_2 = 188[a]			0.074	2.50	3.71
	0.3	0.547	3.46	0.584	$t_{1/2}$ = 6.35; $t_{1/2}$ = 7.9[a]			0.059	2.70	5.01

[a]Complex conjugate pair splits into two simple poles.

TABLE VIII.- Concluded

Condition	Center of gravity	Spiral divergence			Roll subsidence		
		Root	τ_{SD}	$t_{1/2}$ or t_2	Root	τ_{RS}	$t_{1/2}$ or t_2
I	0.1	-0.003	312.5	216.6	-1.918	0.521	0.361
	0.3	0.001	-1429.0	990.2	-1.943	0.515	0.357
II	0.1	0.027	-37.0	25.7	-2.195	0.456	0.316
	0.3	0.026	-38.5	26.7	-2.198	0.455	0.315
III	0.1	-0.003	322.6	223.6	-2.933	0.341	0.236
	0.3	0.009	-111.11	77.0	-2.849	0.351	0.243
IV	0.1	0.019	-52.63	36.5	-2.933	0.341	0.236
	0.3	0.019	-52.63	36.5	-2.931	0.341	0.236
V	0.1	0.003	-321.6	222.9	-2.621	0.382	0.264
	0.3	-0.002	572.3	396.7	-2.09	0.479	0.332
VI	0.1	0.027	-36.77	25.48	-2.542	0.393	0.273
	0.3	0.027	-37.59	26.06	-2.538	0.394	0.273
VII	0.1	0.014	-71.94	49.87	-2.218	0.451	0.313
	0.3	0.014	-73.53	50.97	-2.217	0.451	0.313
VIII	0.1	0.037	-27.32	18.94	-1.710	0.585	0.405
	0.3	0.036	-28.09	19.47	-1.707	0.586	0.406
IX	0.1	0.030	-33.11	22.95	-2.601	0.384	0.266
	0.3	0.088	-11.42	7.91	-2.638	0.379	0.263

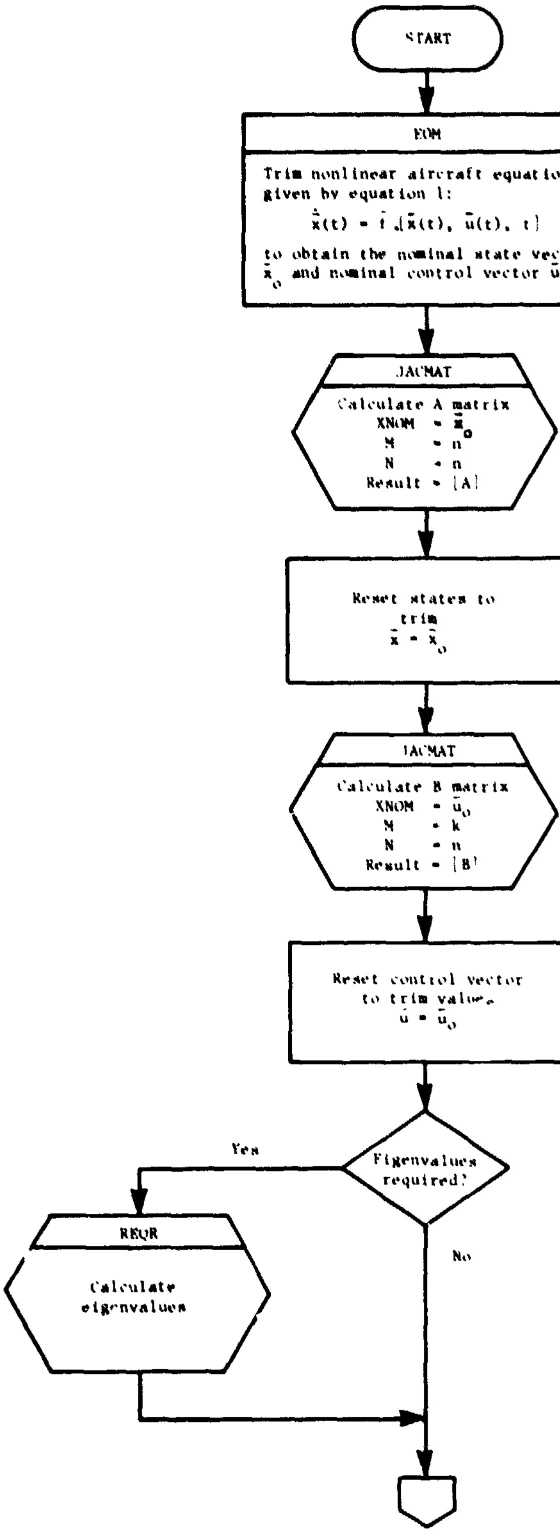

Figure 1.- Program usage flow chart.

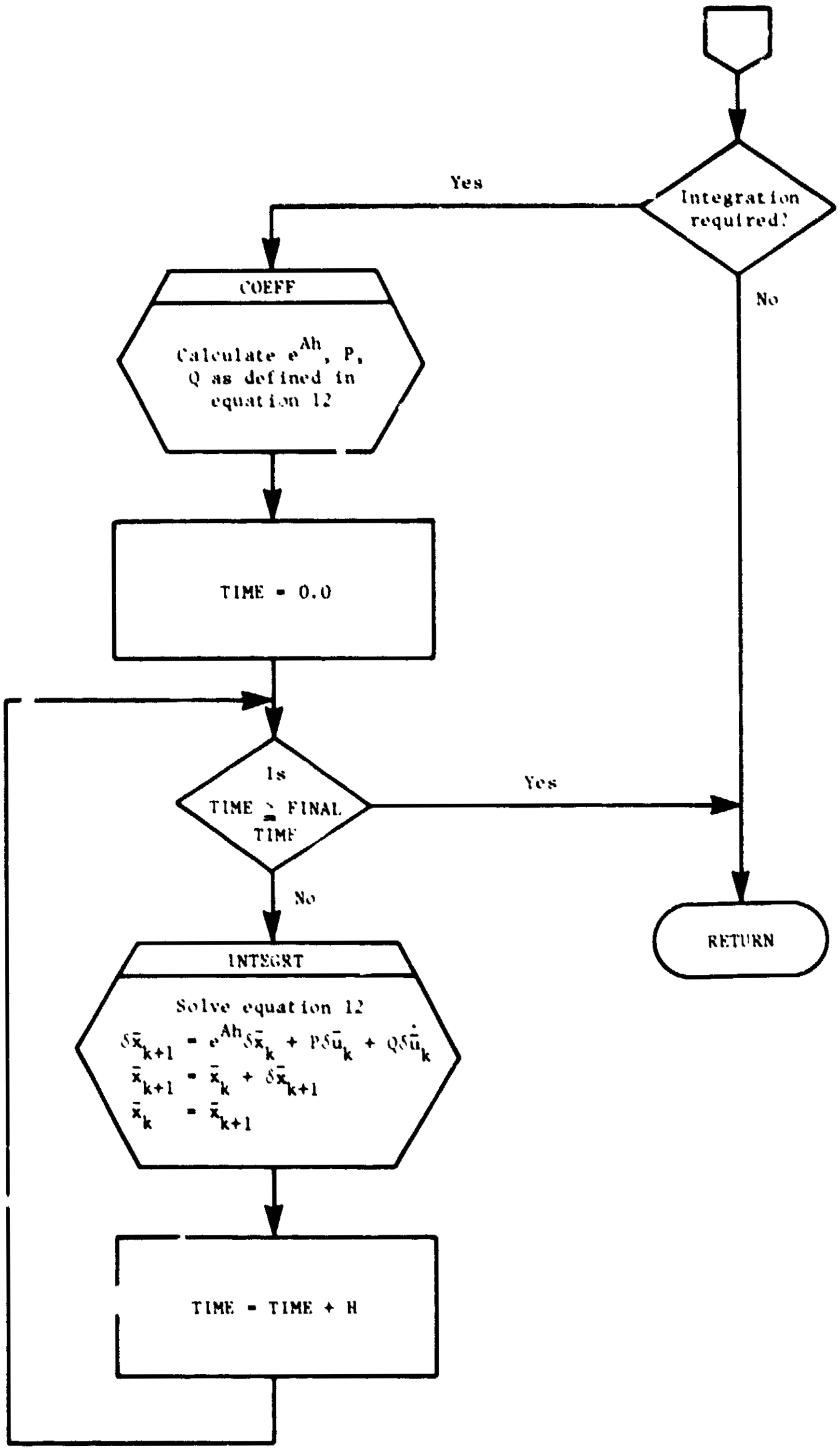

Figure 1.- Concluded.

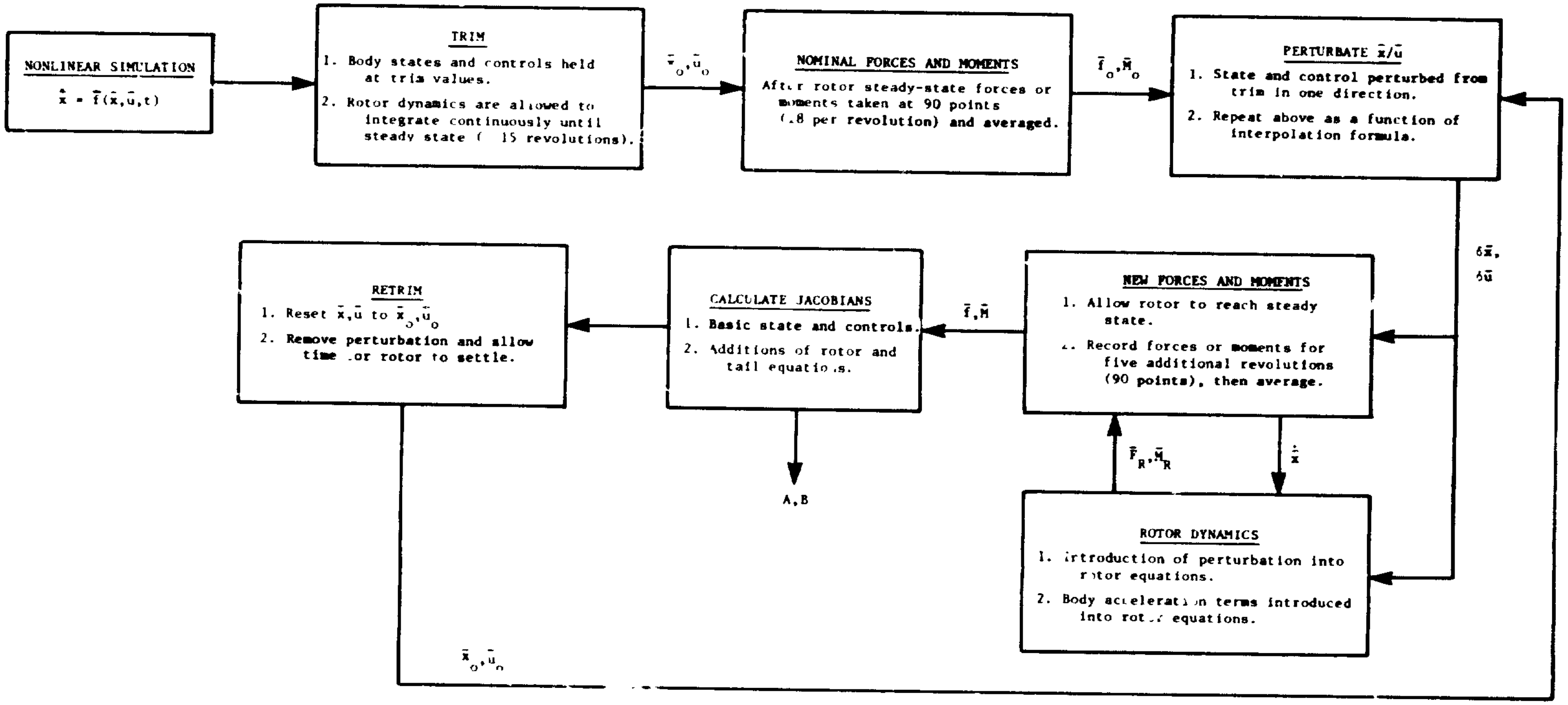

Figure 2.- Procedural modification for system with rotor dynamics.