THESIS

LINEAR MODELING OF ROTORCRAFT FOR STABILITY ANALYSIS AND PRELIMINARY DESIGN

by

Walter M. Wirth, Jr.

September, 1993

Thesis Advisor: E. Roberts Wood

Approved for public release; distribution is unlimited.
This thesis investigates linear state space modeling of single main rotor helicopters culminating in a computer program that can be used for 1) stability and control analysis for any single main rotor helicopter or 2) preliminary design of a helicopter. The trim solution for a flight condition is found, the aircraft is perturbed about the nominal point, and the stability and control derivatives are determined. State space models and analysis tools are provided by the program. A notional attack helicopter designed for the 1993 American Helicopter Society Design Competition and a notional utility helicopter are used as examples.
Linear Modeling of Rotorcraft
for
Stability Analysis and Preliminary Design

by

Walter M. Wirth, Jr.
Major, United States Army
B.S., United States Military Academy, 1981

Submitted in partial fulfillment
of the requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 1993

Author:

Walter M. Wirth, Jr.

Approved by:

E. Roberts Wood, Thesis Advisor
Richard M. Howard, Second Reader
Daniel J. Collins, Chairman

Department of Aeronautical Engineering
ABSTRACT

This thesis investigates linear state space modeling of single main rotor helicopters culminating in a computer program that can be used for 1) stability and control analysis for any single main rotor helicopter or 2) preliminary design of a helicopter. The trim solution for a flight condition is found, the aircraft is perturbed about the nominal point, and the stability and control derivatives are determined. State space models and analysis tools are provided by the program. A notional attack helicopter designed for the 1993 American Helicopter Society Design Competition and a notional utility helicopter are used as examples.
J. HMRGRP.M ................................................. 154
K. HTRGRP.M ................................................. 156
L. STABOUT.M ................................................. 158
M. CMDBWPLH.M ............................................. 172
N. CMDBWPLC.M ............................................. 176
O. DERIV2.M ................................................. 180

LIST OF REFERENCES ............................................. 181
INITIAL DISTRIBUTION LIST .................................... 182
I. INTRODUCTION

A. NATURE OF THE PROBLEM

"Life is too short to worry about helicopter dynamics."

Louis V. Schmidt
Professor Emeritus
Naval Postgraduate School

Helicopter dynamics, because of the complexity added by the high-mass, rotational components of the aircraft, namely the main and tail rotors, is often shunned by the aerodynamicist. The fact that main and tail rotors are the dominant contributors to flight dynamics makes the situation even worse. The understanding of flight mechanics, stability, and control is difficult enough without this added twist. As a result, courses in flight mechanics focus on solutions for fixed wing aircraft.

As will be shown, the flight mechanics of a helicopter are essentially no different than those of a fixed wing aircraft. The analysis of a helicopter in flight depends on solving the equations of motion, just like for a fixed wing aircraft. The key difference is the realization of a trimmed solution for the helicopter, which is an iterative process vice a straightforward, closed-form solution as for the fixed wing aircraft. Once the trim solution is found, the evaluation of stability and control derivatives for the helicopter is done the same as for a fixed wing aircraft except that the derivatives for the main and tail rotors must also be evaluated.

An attempt was made in the preparation of this paper to be as consistent with fixed wing stability notation as possible. The large number of helicopter parameters which are unique or differ from fixed wing notation are listed in Appendix A. Additionally, the symbol $R$ is used for rolling moment instead of the more common $L$ to be consistent with Ray Prouty’s notation, common to helicopter literature.
B. BACKGROUND

This project is the result of two needs at the Naval Postgraduate School.

1. MODELING OF ROTORCRAFT

Modeling of rotorcraft needs to be performed because the school is pursuing the acquisition of aircraft for flight testing of an auto-land helicopter system. The key deficiency in the use of simple, off-the-shelf rotorcraft is the lack of design data and theoretical analysis available for control system design. The modeling presented in this thesis is linear. Non-linear modeling can be performed on Flightlab which is available at the school.

2. SOFTWARE AVAILABILITY

Software available for preliminary helicopter design, such as HESCOMP or CAMRAD, is too complicated for students to use during a one quarter Helicopter Design course. The time required for a student to learn to use available software is prohibitive. The need for user friendly software, based on easily accessible programs such as MATLAB, culminated in two software packages being prepared for preliminary helicopter design. The first package JANRAD Performance [Ref. 1] solves the helicopter trim solution and is used to perform aerodynamic analyses. The second package comes from this study and is used to perform stability and control analysis, but uses the trim solution from JANRAD. Both packages require a working knowledge of MATLAB. MATLAB was used as the base program for these software packages because it is readily available and familiar to students because of its wide use at Naval Postgraduate School.

3. ASSUMPTIONS

The following assumptions were made in the foregoing analysis.

- A person using the program has a working knowledge of MATLAB.
- The aircraft trim condition is in level flight with climb angle \((\gamma_c)\) equal to zero and no sideslip.
The main and tail rotors provide the only significant contribution to stability and control at a hover. Contributions of all other airframe components can be neglected at a hover.

Effects of the separate components of the helicopter contribute linearly to stability and control.

A single main rotor helicopter is used with a counter-clockwise (as viewed from above) rotating main rotor system.

The main rotor system is articulated, bearingless or rigid in plane.

Rotor blades are rigid.

The tail rotor rotates counter-clockwise as viewed from the left side of the helicopter.

The tail rotor is a teetering rotor with a delta-3 (pitch-flap coupling) hinge.

The tail rotor mast is perpendicular to the direction of flight.

Tail rotor coning can be approximated by using only tail rotor thrust and centrifugal forces.

The NOTAR™ boom is circular.

Inflow is uniform.

The fuselage of the helicopter is a rigid body and has aerodynamic characteristics similar to the Prouty example helicopter [Ref. 2].
II. SOFTWARE OVERVIEW

A. SOFTWARE ARCHITECTURE

The Joint Army/Navy Rotorcraft Analysis and Design program, or JANRAD, consists of two major subroutines with others under development. The first routine, JANRAD Performance, calculates the trim solution and various performance parameters of a helicopter. It is described in detail in Ref. 1.

The second major component is JANRAD Stability, written by the author, which 1) calculates all the stability derivatives of each component of the helicopter, 2) determines the linear state space model at any trimmable point in the forward flight regime, 3) provides the eigenvalues of the plant matrix and plots them in the Argand plane, and 4) determines the open loop transfer functions from control inputs to state outputs of the aircraft.

The basic architecture of the program is in Figure 2.1. The opening menu has the user enter data for performance calculations, then branches to either the performance or stability and control routines as desired. When the user selects the stability routines, additional input data is requested by the program for analysis. Data required for each major aircraft component is delineated under the chapters describing the stability and control solution for the component, and is listed in Appendix F.

Upon data entry, the program performs either a hovering or forward flight stability analysis of the helicopter. For calculational purposes, hovering flight is any velocity less than 20 ft/sec or 12 knots. Each analysis requires a number of perturbations about the nominal flight condition, which is done by performing a perturbation, then by calling the TRIM subroutine to determine the new trim condition. This procedure is described in more detail in following sections.
Figure 2.1. Program Architecture.
After the stability and control analysis is performed, the user can select from five different screen outputs depending on his/her needs. Information not available from the output screens can be called from the workspace as described in section III.A.

The linear models provided by the software are saved in a MATLAB data file and can be used to perform simulations of aircraft response due to particular control inputs. This is described in the Software Use, and Examples chapters.

B. MODELLING CONVENTIONS

The coordinate system used in the modelling of the helicopter is the standard set of stability and control axes (Fig. 2.2). The x axis passes through the longitudinal axis of the helicopter with the positive direction to the front of the aircraft. The y axis passes through the lateral axis of the helicopter with the positive direction to the right of the aircraft. The z axis passes through the vertical axis with the positive direction to the bottom of the aircraft, or toward the earth if the aircraft is upright. This set of coordinates is standard in flight mechanics, though it may be a bit confusing initially.

The coordinate system describing the locations of various aircraft components is the same coordinate system used to compute weight and balance for an aircraft. The fuselage station (x axis) determines the longitudinal position of the component, the buttline (y axis) determines the lateral position, and the waterline (z axis) determines the vertical component. The axes are oriented with the x axis parallel to the longitudinal axis of the aircraft with positive direction to the rear, the y axis is parallel to the lateral axis with the positive direction to the right side of the aircraft. The z direction is orthogonal to the x and y axis with the positive direction upward (Fig. 2.2).

Modelling of the helicopter was done in standard state-space format, where the dynamics of the aircraft are described by a system of differential equations, written as the following system of matrix equations.
\[ \dot{X} = [A] X + [B] u \]
\[ y = [C] X + [D] u \]

\( X \) is the state vector which contains the current state of the system. States are selected which are the outputs of integrators and that describe the motion of the aircraft. For the complete coupled system, the following states were selected per Ref. 3.

- \( u \) - velocity of the aircraft in the positive \( x \) direction
- \( w \) - velocity of the aircraft in the positive \( z \) direction
- \( q \) - pitch rate of the aircraft
- \( \theta \) - pitch angle of the aircraft
- \( v \) - velocity of the aircraft in the positive \( y \) direction
- \( p \) - roll rate of the aircraft
The vector $u$ contains the inputs to the system. In the case of the helicopter, the inputs are

$\delta B_1$ - change in longitudinal cyclic or elevator

$\delta \theta_{\text{col}}$ - change in collective

$\delta A_1$ - change in lateral cyclic or aileron

$\delta p$ - change in anti-torque pedals or rudder

$[A]$ is the plant matrix and describes the dynamics of the aircraft. The plant matrix is determined by using Newton's laws to sum forces and moments about the helicopter at an equilibrium condition. This yields a system of differential equations. The system of equations is written in matrix form where the matrix relating the states $X$ with the derivatives of the states is the plant matrix $[A]$. The plant matrix used is described in NASA Technical Memorandum 84281 [Ref. 3].

The $[B]$ matrix shows the contributions of the inputs $u$ to the system. $[B]$ multiplies the input vector $u$, and the result is added to the system. Each element of $[B]$ is the partial derivative of the corresponding state with respect to the input. For example $B(1,2)$ would be the partial derivative of $u$ with respect to collective, $\partial u/\partial \theta_{\text{col}}$.

The vector $y$ is the output of the system. The desired output is described by using the $[C]$ and $[D]$ matrices. The matrix $[C]$ is used to determine the output of the system from the states $X$. If the values of the state vector are the desired output, as in the case of this paper, then the matrix is an identity matrix the size of the plant matrix; however, the $[C]$ matrix does not have to be the identity
matrix and can be set up to provide a wide variety of output. For example, if the desired output is to be lateral acceleration, or $\ddot{v}$, then the fifth row of the plant matrix, which corresponds to the equation
\[
\ddot{v} = \frac{\partial v}{\partial u} \dot{u} + \frac{\partial v}{\partial w} \dot{w} + \frac{\partial v}{\partial \theta} \dot{\theta} + \frac{\partial v}{\partial q} \dot{q} + \frac{\partial v}{\partial p} \dot{p} + \frac{\partial v}{\partial \psi} \dot{\psi}
\]
would be used in the $[C]$ matrix to provide the output.

The final matrix, $[D]$, is the direct transition matrix for the input. It can be used to provide a direct feed from the input vector $u$ to the output vector $y$.

C. THE TRIM SOLUTION

Before a stability and control analysis can be performed on a helicopter, the trim solution for the main and tail rotors must be obtained. The trim solution solves for the collective pitch ($\theta_{cm}$), the thrust moment of the blade ($T_m$), the first lateral harmonic of blade feathering ($A_1$), the first longitudinal harmonic of blade feathering ($B_1$), the rotor coning angle ($\alpha_{cm}$) and the tip path plane angle ($\alpha_{qcm}$) of the main rotor. When these values are solved for, the first coefficient of longitudinal blade flapping ($a_{1cm}$), and the first coefficient of lateral blade flapping ($b_{1cm}$), are then computed. Additionally, the trim solution for the tail rotor is needed to provide tail rotor collective ($\theta_{ct}$), the first coefficient of longitudinal blade flapping ($a_{1ct}$), the first coefficient of lateral blade flapping ($b_{1ct}$), and the tail rotor coning angle ($\alpha_{ct}$) for stability analysis.

The trim solution for the main rotor is solved by the TRIM routine in JANRAD. The values $a_{1cm}$, $b_{1cm}$ are then computed in the Stability and Control Routine. The tail rotor solution is approximated using closed form solutions in the Stability and Control Routine.

D. GENERAL PROCEDURE TO EVALUATE STABILITY DERIVATIVES

Stability derivatives are determined by using closed form solutions whenever possible, and by solving multiple trim solutions about a nominal point to solve for unknowns.
By using the TRIM routine in the JANRAD program, the trim solution for the desired flight condition is determined. The helicopter is then perturbed about this nominal condition and retrimmed. From these additional solutions, the basic stability derivatives for the main and tail rotors are solved for.

By solving for the basic stability derivatives first, the helicopter can be described by using the same states as in the standard linear model of a fixed wing aircraft \([u \, v \, w \, p \, q \, r \, \psi \, \theta \, \Phi]'\). Without solving for the intermediate derivatives first, the helicopter would have to be described using additional states for the main rotor, \([a_{iM} \, b_{iM} \, C_{rM}/\sigma_{M} \, C_{PM}/\sigma_{M} \, C_{QM}/\sigma_{M} \, \mu_{M} \, \lambda_{M}]'\), and the tail rotor, \([a_{iT} \, b_{iT} \, C_{rT}/\sigma_{T} \, C_{PM}/\sigma_{T} \, C_{QT}/\sigma_{T} \, \mu_{T} \, \lambda_{T}]'\). Aside from the fact that a model described in this fashion is inconsistent with the standard flight mechanic description, the size of the system model becomes large and intimidating. This is part of the reason that helicopter flight mechanics are generally avoided.

After the basic stability derivatives are determined, intermediate derivatives are evaluated. Closed form solutions are used whenever possible to solve intermediate derivatives. After an intermediate derivative is determined, the basic derivative and the intermediate derivative are combined using the chain rule to form the complete stability derivative. An example solution of a main rotor derivative follows.

\[
\text{Wanted: } \left( \frac{\partial X}{\partial \theta_0} \right)_M
\]

\text{Find from rotor plots or numerically: } \left( \frac{\partial C_{rM}/\sigma}{\partial \theta_{0M}} \right)_M \left( \frac{\partial C_{\muM}}{\partial \theta_{0M}} \right)_M \left( \frac{\frac{\partial a_{iM}}{a}}{\partial \theta_{0M}} \right)_M

\text{Compute:}

\[
\left( \frac{\partial C_{rM}/\sigma}{\partial a_{iM}} \right)_M = C_{rM}/\sigma + \lambda/\delta
\]

\[
\left( \frac{\partial X}{\partial \theta_0} \right)_M = -p_{A}(\Omega R)^2 \left[ \frac{\partial C_{rM}/\sigma}{\partial \theta_{0M}} + \frac{\partial C_{\muM}}{\partial \theta_{0M}} \left( \frac{\partial a_{iM}}{\partial \theta_{0M}} + (a_{iM} + \mu) \frac{\partial C_{rM}/\sigma}{\partial \theta_{0M}} \right) \right]
\]
III. STABILITY AND CONTROL DERIVATIVES

A. MAIN ROTOR CONTRIBUTIONS

The main rotor is the key contributor to helicopter flight dynamics. The strong coupling between the states of the rotor produce complicated responses in many of the aircraft states at the same time. This can be visualized through the following example.

If collective pitch is increased in forward flight, the increased drag in the rotor system increases the amount of torque required from the engine; this results in a yaw of the helicopter. Simultaneously, since the tip path plane is tilted forward, the forward thrust increases which increases the advanced ratio. This increase in advance ratio results in differing amounts of lift on the advancing and retreating sides of the rotor disk which results in a pitching moment because of the phase lag of the blades. At the same time, the pitching moment changes the angle of attack on the fore and aft blades, which changes lateral flapping, which changes the rolling moment. Because of the cross coupling between lateral and longitudinal flapping, there is a change in longitudinal flapping. And so on.

The basic stability derivatives of the main rotor are described in terms of the partial derivatives of the standard helicopter coefficients, $a_{1M}$, $b_{1M}$, $C_{\gamma}/\sigma_M$, $C_{\mu}/\sigma_M$, $C_{\lambda}/\sigma_M$, $\mu_M$, $\lambda'$, $A_{1M}$, $B_{1M}$, and the main rotor blade stiffness

$$\left( \frac{dM}{da_i} \right)_M \cdot \left( \frac{dR}{db_i} \right)_M = \frac{3 \epsilon \rho R (\Omega R)^2 a}{4 R} \gamma$$
These basic main rotor derivatives are then combined as described in the prior section to form the main rotor stability derivatives. A detailed list of the basic main rotor derivatives and the stability derivatives is available in Appendix C.

To solve for the derivatives, JANRAD uses input data from the performance menu, and output computations from the TRIM section of the program. The input required for the trim solution is found in the JANRAD Performance thesis [Ref. 1]. Output from the performance routine and input from the Stability and Control Routine New File menu (Fig. 3.1) are used to solve for $a_{im}$, $b_{im}$, $C_{r/\sigma_M}$, $C_{p/\sigma_M}$, $C_{q/\sigma_M}$, $\mu_M$, $\lambda'$, $A_1$, $B_1$. Input data can be changed in the Stability and Control Additional Parameters Menu screen 1 (Fig. 3.2).

Blade flapping moment of inertia (slug ft^2):
Hub height above reference datum/waterline (ft):
Hub fuselage station (ft):
Hub position right of buttline (ft):
Mast incidence (negative forward - deg):

Figure 3.1. Main Rotor New File Menu.

The perturbation solution in conjunction with the equations in Appendix C solve for the partial derivatives of $a_{im}$, $b_{im}$, $C_{r/\sigma_M}$, $C_{p/\sigma_M}$, $C_{q/\sigma_M}$, $\mu_M$, $\lambda'$, $A_1$, $B_1$ with respect to each other, thus completing the evaluation of the basic main rotor derivatives. The CRUISE.M and HOVER.M routines then combine the basic and intermediate derivatives to give a complete set of main rotor derivatives.

Main rotor output is displayed in output screen 2 (Fig. 3.3). Derivatives are stored in the workspace. Because of the large number of derivatives, it is impractical to display all of them on the screen. Instead, key derivatives and control power are displayed in the Key Control Parameters output screen (Fig. 3.4) AFTER all contributions from other aircraft components are determined.
*** STABILITY AND CONTROL MENU ***
*** ADDITIONAL PARAMETERS (1 of 3) ***

Main Rotor
1. flapping mom of inertia 2. hub height above waterline
3. hub fuselage station  4. hub posn right of buttline
5. mast incidence

Tail Rotor (enter zeros (0) if using NOTAR)
6. height above waterline 7. hub fuselage station
8. posn right of buttline 9. number of blades
10. blade chord 11. blade radius
12. lift curve slope  13. rotational velocity
14. flap mom of inertia 15. delta-3 angle
16. blade twist

Vertical Fin
17. height above waterline 18. fuselage station
19. posn right of buttline 20. alpha zero lift
21. CL max 22. dynamic pressure ratio
23. lift curve slope

0. NO CHANGES
Input the parameter to change:

Figure 3.2. Additional Parameters Menu, Screen 1.

*** INPUT DATA CONTINUED (screen 2 of 8) ***
filename

Main Rotor

advance ratio = mu
inflow parameter wrt TPP = lamp
Tip path angle = altpp*57.3 dege
Rotor coning angle = ao*57.3 dege
1st lat cyclic term-A1 = A1*57.3 dege
1st long cyclic term-B1 = B1*57.3 dege
lateral flapping = bls*57.3 dege
longitudinal flapping = als*57.3 dege
Lock number = lockno

Figure 3.3. Main Rotor Calculated Output.

All derivatives can be called from the workspace. This is described in the Output Data Instructions screen 1 (Fig. 3.5). The file VARLIST.TXT is found in Appendix D.
*** KEY CONTROL PARAMETERS ***

cross coupling = xcouple

Designed damping
pitch = desm/dq ft-lbs/(rad/sec)
roll = desd/dr ft-lbs/(rad/sec)
yaw = desd/dr ft-lbs/(rad/sec)

Control Power
pitch = ccppitch ft-lbs/in
roll = ccpproll ft-lbs/in
yaw = ccppyaw ft-lbs/in

Cooper Harper Pilot Ratings
damping/moment of inertia
pitch (dM/dq)/Iyy = prpitch [ft-lbs/(rad/sec)]/(slug ft^2)
roll (dR/dp)/Ixx = prroll [ft-lbs/(rad/sec)]/(slug ft^2)
yaw (dN/dr)/Izz = pryaw [ft-lbs/(rad/sec)]/(slug ft^2)

control power/moment of inertia
pitch (dM/in)/Iyy = ccppitch (ft-lbs/in)/(slug ft^2)
roll (dR/in)/Ixx = ccpproll (ft-lbs/in)/(slug ft^2)
yaw (dN/in)/Izz = ccppyaw (ft-lbs/in)/(slug ft^2)

---

Figure 3.4. Key Control Parameters Output Screen.

*** OUTPUT DATA INSTRUCTIONS (screen 1 of 3) ***

Because this subroutine generates a large number of single value data not shown on the output screen, a text file VARLIST.TXT is on this disk which lists the variable names for all the stability derivatives. Stability derivative contributions for all major aircraft components can be found by reading the text file VARLIST.TXT, then asking MATLAB the variable name corresponding to the derivative.

Press any key to continue

Figure 3.5. Output Data Instructions, Screen 1.

B. TAIL ROTOR CONTRIBUTIONS

Before stability contributions of the tail rotor can be determined, the tail rotor trim solution must be derived. Since only the main rotor trim solution is computed in the performance section of JANRAD, the tail rotor trim solution is solved in the following manner.
By assuming that the shaft axis of the tail rotor is perpendicular to the flight path of the helicopter, the tip path plane angle (\( \alpha_{\text{tp}} \)) is equal to longitudinal flapping (\( \alpha_{\text{f}} \)). Additionally, no longitudinal or lateral cyclic pitch is introduced by the control system, hence \( B_{\text{rr}} = 0 \) and \( A_{\text{rr}} = 0 \). This reduces the inflow parameter with respect to the tip path plane from [Ref. 2:p. 189]

\[
\lambda' = \mu \left[ \alpha_{\text{tp}} - (B_{\text{rr}} + a_{\text{f}}) \right] - \frac{v_1}{\Omega R}
\]

This yields

\[
\lambda = - \frac{v_1}{\Omega R} = -C_{\text{q/f}} \frac{\sigma}{2\mu}
\]

Tail rotor thrust \( T_r \) is computed by using the main rotor torque from the main rotor trim solution, length of the moment arm to the tail rotor, lift of the vertical tail, and length of the moment arm to the vertical tail. The moment contributions of the tail rotor and vertical tail must equal the torque of the main rotor. This yields

\[
T_r = \frac{C_{\text{Q/R}} \rho A_{\text{m}} (\Omega R)^2 - L_v l_v}{l_T}
\]

Tail rotor coning is arrived at by using centrifugal forces and tail rotor thrust. This gives the following, where \( \gamma \) is the Lock number of the tail rotor blades [Ref. 2:p 171].

\[
a_0 = \frac{2}{3} \gamma \frac{C_{\text{q/f}} \sigma}{a} - \frac{3}{2} \frac{g R^2}{(\Omega R)^2}
\]

Longitudinal flapping is determined by writing the flapping equations for the tail rotor in terms of \( a_0, C_{\text{q/f}}, \mu, \) and the tail rotor pitch-flap coupling angle \( \delta \). Furthermore, this allows the lateral flapping to be determined in terms of the above values [Ref. 2:p 189].
\[
\begin{align*}
\frac{4}{3} \mu a_0 + C_d/\sigma - a_1 \tan \delta &= 2 \left[\frac{4}{3} \mu a_0 + C_d/\sigma \right] - 2 \mu \left[1 - \left(-\frac{4}{3} \mu \right)^2 \right] \tan \delta \\
a_1 &= \frac{2 - \mu^2}{2 + \mu^2} \left[1 - \left(-\frac{4}{3} \mu \right)^2 \right] \tan \delta \\
\end{align*}
\]

To arrive at $\theta_{or}$, $C_d/\sigma$, $\mu$, tail rotor twist $\Theta_{or}$, and $\lambda$ are used yielding [Ref. 2:p 187].

\[
\theta_{or} = \frac{4}{3} \frac{C_d/\sigma}{\sigma} \tan \delta - \frac{1}{2} \frac{\mu^2}{\sigma} \theta_1 - \mu b_1 \tan \delta - \lambda
\]

At a hover, there is no longitudinal or lateral flapping in the tail rotor. Because of this, the tail rotor equations can be resolved in terms $a_0$, $\theta_{or}$, $\Theta_{or}$, $\Theta_{or}$, $h_i$, $l_i$, and the main rotor trim conditions.

This results in the following set of equations for the tail rotor at a hover.

\[
\begin{align*}
a_0 &= 0 \\
b_0 &= 0 \\
\frac{C_T}{\sigma} &= \frac{a}{4} \left( \frac{v_i}{\Omega R} \sqrt{\frac{\sigma_M (C_d/\sigma)_M}{2}} \right) \\
\frac{C_Q}{\sigma} &= \frac{C_T}{2} \frac{c_d}{8}
\end{align*}
\]

where $c_d$ is approximated from

\[
c_d = c_d a \left( \frac{6C_d/\sigma}{\pi e c_i R} \right)
\]
Once a trim solution has been determined, stability derivatives are found in the same manner as for the main rotor, by perturbing about a nominal point. At a hover, the stability derivatives are computed in terms of the main rotor stability derivatives as in Ref. 2.

These equations may produce flapping magnitudes that are lower than more exact solutions, but since the trends of the perturbations are the same in both cases, we can use these trim equations as the basis for solving the basic derivatives of the tail rotor [Ref. 2:p 189].

In addition to the tail rotor parameters input from the JANRAD opening menu, additional parameters are required for the stability and control analysis. Tail rotor inputs are entered in the Stability and Control Routine New File menu (Fig. 3.6) and changed in the Stability and Control Menu Additional Parameters menu, screen 1 (Fig. 3.7).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tail rotor height above reference datum/waterline (ft):</td>
<td></td>
</tr>
<tr>
<td>Tail rotor fuselage station (ft):</td>
<td></td>
</tr>
<tr>
<td>Tail rotor position right of buttline (ft):</td>
<td></td>
</tr>
<tr>
<td>Number of tail rotor blades:</td>
<td></td>
</tr>
<tr>
<td>Blade chord (ft):</td>
<td></td>
</tr>
<tr>
<td>Tail rotor blade radius (ft):</td>
<td></td>
</tr>
<tr>
<td>Average lift curve slope of tail rotor:</td>
<td></td>
</tr>
<tr>
<td>Rotational velocity of tail rotor (rad/sec):</td>
<td></td>
</tr>
<tr>
<td>Blade flapping moment of inertia (slug ft^2):</td>
<td></td>
</tr>
<tr>
<td>Delta-3 angle (deg):</td>
<td></td>
</tr>
<tr>
<td>Blade twist (deg):</td>
<td></td>
</tr>
</tbody>
</table>

Figure 3.6. Tail Rotor New File Menu.

The program tests to see if an anti-torque device is installed by checking the number of tail rotor blades and the NOTAR™ fan diameter. If both are zero, an error message is displayed (Fig. 3.8).
*** STABILITY AND CONTROL MENU ***
*** ADDITIONAL PARAMETERS (1 of 3) ***

Main Rotor
1. flapping mom of inertia  2. hub height above waterline
3. hub fuselage station  4. hub posn right of buttline
5. mast incidence

Tail Rotor (enter zeros (0) if using NOTAR)
6. height above waterline  7. hub fuselage station
8. posn right of buttline  9. number of blades
10. blade chord  11. blade radius
12. lift curve slope  13. rotational velocity
14. flap mom of inertia  15. delta-3 angle
16. blade twist

Vertical Fin
17. height above waterline  18. fuselage station
19. posn right of buttline  20. alpha zero lift
21. CL max  22. dynamic pressure ratio
23. lift curve slope'

0. NO CHANGES
Input the parameter to change:

Figure 3.7. Additional Parameters Menu, Screen 1.

You must have a tail rotor or NOTAR/thruster!

Figure 3.8. Missing Component Warning.

Tail rotor output is displayed in Input Variable output screen 3 and in the Tail Rotor Calculated Data screen (Fig 3.9).

Tail rotor derivatives are solved for in the same manner as for the main rotor. A detailed listing of tail rotor derivatives is in Appendix C.

C. FUSELAGE CONTRIBUTIONS

The fuselage contributions to stability and control are typically arrived at through empirical methods, so are best estimated initially by referring to characteristics of similar helicopters.
Lift and drag moments of the fuselage depend on the angle of attack $\alpha_F$, and the sideslip angle $\beta_F$ of the fuselage. Because of the downwash induced by the main rotor, $\alpha_F$ is estimated by

$$\alpha_F = \theta - \gamma - \varepsilon_M$$

where $\theta$ is the fuselage pitch attitude, $\gamma$ is the climb angle, and

$$\varepsilon_M = \frac{v_F}{v_i} \frac{T_M}{4 q A_M}$$

The value $v_F/v_i$, the fuselage downwash ratio, is the only fuselage stability parameter required to be input into the stability and control program. It can be estimated by methods outlined in Ref. 2. A typical value of 1.5 is recommended otherwise.
The other parameters input with fuselage data are the cg location, and moments of inertia of the aircraft.

Input for the fuselage is made in the Stability and Control Routine New File menu (Fig. 3.10) and changed in the Stability and Control Menu Additional Parameters menu, screen 3 (Fig. 3.11).

CG height above reference datum/waterline (ft):
CG Fuselage station (ft):
CG position right of buttline (ft):
lxx (slug ft\(^2\)):
lyy (slug ft\(^2\)):
lzz (slug ft\(^2\)):
lxz (slug ft\(^2\)):

Downwash ratio for fuselage (page 513 Prouty):

Figure 3.10. Fuselage New File Menu.

*** STABILITY AND CONTROL MENU ***
*** ADDITIONAL PARAMETERS (3 of 3) ***

CG location and Inertias/fuselage parameters
1. cg ht. above waterline  2. cg fuselage station
3. cg posn rt of buttline   4. lxx
5. lyy                      6. lzz
7. lxz                      8. fuselage downwash ratio

NOTAR if available (enter zeros if using tail rotor)
9. height above waterline   10. boom fuselage station
11. boom position left ref  12. NOTAR diameter
13. swirl angle at boom     14. NOTAR max force
15. thruster fuselage station

Rigging
16. B1 main/in defl (del e) 17. A1 main/in defl (del a)
18. theta0m/in defl (del c) 19. theta0t/ pedal defl (del r or p)
20. NOTAR sleeve twist/defl 21. max rudder defl

0. NO CHANGES
Input the parameter to change:

Figure 3.11. Additional Parameters Menu, Screen 3.

Stability derivatives from the fuselage are combined with the stability derivatives of other components for display in the Key Control Parameters output screen. The individual contributions
of the fuselage to stability can be pulled from the workspace by finding the variable name in VARLIST.TXT and calling the value as described in the Main Rotor section.

The values of the basic fuselage stability contributions are set in the program to typical values found in Ref. 2. If one desires to change these values, edit the subroutine CBODYGRP.M lines 17 through 22, substituting desired values for the defaults.

D. HORIZONTAL STABILIZER CONTRIBUTIONS

Contributions of the horizontal tail are computed as for a fixed wing aircraft with one exception. In a helicopter, the downwash created by the rotor is very large, and in nearly all modes of flight, the horizontal tail is in the downwash and is affected.

The effects of this are accounted for by adjusting the dynamic pressure on the horizontal tail by determining the dynamic pressure ratio $q_H/q$, and downwash angles due to the main rotor $\epsilon_m$, and fuselage $\epsilon_f$. These effects are generally evaluated in some empirical fashion, but some general guidelines follow.

The dynamic pressure at the horizontal and vertical tail is generally lower than free stream due to momentum loss from the fuselage and main rotors. A first guess for $q_H/q$ can be made by using Figure 8.9 Ref. 2. If no information is available, the author recommends a typical value of 0.6 be used.

The main rotor downwash ratio for the horizontal tail has the form

$$\epsilon_m = \frac{v_H v_1}{v_1 V} = \frac{v_H D.L.}{v_1 4q}$$

The ratio $v_H/v_1$ has a maximum theoretical value of two because of the maximum theoretical downwash velocity achieved by the main rotor using momentum theory, but empirical measurements
indicate that higher ratios are obtainable [Ref. 2]. The term \( v_r/v \), can be estimated by using Fig 8.11 Ref. 2. The author recommends a typical value of 1.5 in the absence of more precise information.

The fuselage downwash ratio has the form

\[
\epsilon_{r_w} = \epsilon_{r_w} + \left( \frac{dx_{r_w}}{d\alpha} \right) \alpha
\]

The fuselage downwash ratio can be estimated using Fig 8.14 Ref. 2. The fuselage downwash ratio includes the effects of any wings since most wings installed on a helicopter are small and do not warrant a separate term for downwash effects. A typical value of 0.25 is recommended when no other information is available. Detailed discussion on determination of these effects are contained in Ref. 2 page 489.

The final effect to account for is the time lag for change in rotor downwash to reach the horizontal tail. This effect is accounted for through the time lag

\[
\Delta t = \frac{l_H}{V}
\]

yielding the following equation for the angle of attack for the horizontal tail

\[
\alpha_H = \theta + i_H - (e_{M_H} + e_{r_w}) - \gamma_H \left( \frac{\partial M_H}{\partial \alpha} \right) \frac{l_H}{V}
\]

These effects, when combined with typical computations for horizontal stabilizer stability analysis, yield the equations found in Appendix C.

Input for the horizontal tail is made in the JANRAD Performance input screen with additional input required for stability analysis entered in Stability and Control Routine New File menu (Fig.
Height above reference datum/waterline (ft):
Fuselage station (ft):
Position right of buttline:
Zero lift angle for horizontal tail (deg):
Angle of incidence of horizontal tail (deg):
Lift curve slope of horizontal tail:
Dynamic pressure ratio (pg 489 Prouty):
Rotor downwash ratio (pg 489 Prouty):
Fuselage downwash ratio (pg 489 Prouty):

Figure 3.12. Horizontal Tail New File Menu.

3.12) and changed in the Stability and Control Menu Additional Parameters menu, screen 2 (Fig. 3.13).

*** STABILITY AND CONTROL MENU  ***
*** ADDITIONAL PARAMETERS (2 of 3) ***

Horizontal Tail
1. height above waterline  2. fuselage station
3. posn right of buttline  4. alpha zero lift
5. angle of incidence      6. lift curve slope
7. dynamic pressure ratio  8. rotor downwash ratio
9. fuselage downwash ratio

Wing
10. height above waterline 11. fuselage station
12. posn right of buttline 13. alpha zero lift
14. angle of incidence     15. lift curve slope
16. tip cord               17. root cord
18. rotor downwash ratio  19. fuselage downwash ratio

0. NO CHANGES
Input the parameter to change:

Figure 3.13. Additional Parameters Menu, Screen 2.

Stability derivatives from the horizontal tail are combined with the stability derivatives of all other components for display in the Key Control Parameters output screen. The individual horizontal tail stability derivatives can be pulled from the workspace by finding the variable name in VARLIST.TXT and calling the value from the workspace.
E. VERTICAL STABILIZER CONTRIBUTIONS

The stability derivatives are evaluated in the same manner as the horizontal tail. The only differences are that the swirl from the main rotor downwash, the induced flow from the tail rotor, induced velocities from the fuselage and aircraft sideslip all create an effective sideslip (sidewash) angle on the vertical fin.

If the tail rotor is mounted close to the vertical fin, the form for the downwash effect is the same as for the main rotor downwash on the fuselage

\[ \eta_T = \left[ \frac{D.L.}{4 \frac{q'}{q} g} \right], \]

where

\[ D.L. = \frac{1}{A_T} \frac{Q_{H} - L_{n} \dot{i}}{l_{T}} \]

The parameter \( q'/q \) can be estimated using Figure 8.9 in Ref. 2. A typical value of 0.6 is recommended otherwise.

The effect of aircraft sideslip on the vertical fin is negligible when the tail rotor is in close proximity to the vertical fin because the induced velocities of the tail rotor are much higher than the sideslip velocities typical in the linear region of this analysis. In the case where the tail rotor is far away from the vertical fin, or when a NOTAR™ is used, a typical value of sideslip effect is assigned by the program in the form
The induced sidewash from the main rotor and other factors affecting the tail vertical fin are evaluated by Stability and Control section of JANRAD and do not need to be input by the user. A detailed explanation of these effects in available in Ref. 2.

Input for the vertical tail is made in the JANRAD Performance input screen with additional input required for stability analysis entered in Stability and Control Routine New File menu (Fig. 3.14)

Height above reference datum/waterline (ft):
Fuselage station (ft):
Position right of buttline (ft):
Zero lift angle for vertical tail (deg):
Maximum Cl for vertical tail:
Dynamic pressure ratio (pg 489 Prouty):
Lift curve slope of vertical tail:

Figure 3.14. Vertical Tail New File Menu.

and changed in the Stability and Control Menu Additional Parameters menu, screen 1 (Fig. 3.15).

As with the horizontal tail, results for the vertical tail are combined with other stability derivatives for final output, but can be called from the workspace individually as outlined previously.

F. WING CONTRIBUTIONS

Most helicopters do not have wings, and the ones that do usually have wings only in the sense that the wings are aerodynamic hard points for weapon mounting. But compound helicopters do have wings that are designed to provide a large amount of lift for the helicopter.

Stability and control analysis for a wing is the same as for a fixed wing aircraft, except that large amounts of rotor downwash must be accounted for. Rotor wash is accounted for in the same fashion as for the horizontal tail and the fuselage. The dynamic pressure at the wing is adjusted by
using a dynamic pressure ratio, $q_w/q_1$. The angle of attack is modified by down wash terms.

$$\epsilon_M = \frac{v_w v_i}{v_i V} = \frac{v_w D.L}{v_i 4q}$$

$$\epsilon_{\alpha_w} = \epsilon_{\alpha_w} + \left( \frac{d\epsilon_{\alpha_w}}{d\alpha_F} \right) \alpha_F$$

$$\alpha_w = \theta + \epsilon_{\alpha_w} + \alpha_{o_w} - \gamma_c - \epsilon_M - \epsilon_{\alpha_w}$$

Input for the wing is made in the JANRAD Performance input screen with additional input required for stability analysis entered in Stability and Control Routine New File menu (Fig. 3.16) and changed in the Stability and Control Menu Additional Parameters menu, screen 2 (Fig. 3.17).
Height above reference datum/waterline (ft):
Fuselage station (ft):
Position right of buttline (ft):
Zero lift angle for wing (deg):
Angle of incidence of wing (deg):
Lift curve slope of wing:
Tip chord (ft):
Root chord (ft):
Rotor downwash ratio (pg 489 Prouty):
Fuselage downwash ratio (pg 489 Prouty):

Figure 3.16. Wing New File Menu.

*** STABILITY AND CONTROL MENU ***
*** ADDITIONAL PARAMETERS (2 of 3) ***

Horizontal Tail
1. height above waterline 2. fuselage station
3. posn right of buttline 4. alpha zero lift
5. angle of incidence 6. lift curve slope
7. dynamic pressure ratio 8. rotor downwash ratio
9. fuselage downwash ratio

Wing
10. height above waterline 11. fuselage station
12. posn right of buttline 13. alpha zero lift
14. angle of incidence 15. lift curve slope
16. tip chord 17. root chord
18. rotor downwash ratio 19. fuselage downwash ratio

0. NO CHANGES
Input the parameter to change:

Figure 3.17. Additional Parameters Menu, Screen 2.

As with the horizontal tail, results are combined with other stability derivatives for final output, but can be called from the workspace individually as outlined previously.
G. NOTAR™

NOTAR™ derivatives were obtained from proprietary information and cannot be developed here. The inputs for NOTAR™ are entered in the Stability and Control Routine New File menu (Fig. 3.18) and can be changed in the Stability and Control Additional Parameters Menu screen 3 (Fig. 3.19). NOTAR™ input is echoed in the 'Input Variable' output screen 4, but derivatives must be called from the workspace as described in prior sections.

<table>
<thead>
<tr>
<th>Height above reference datum/waterline (ft):</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fuselage station (ft):</td>
</tr>
<tr>
<td>Position right of buttline (ft):</td>
</tr>
<tr>
<td>NOTAR boom diameter (ft):</td>
</tr>
<tr>
<td>Swirl angle at boom (deg):</td>
</tr>
<tr>
<td>Maximum thruster force of NOTAR (lbs):</td>
</tr>
<tr>
<td>Thruster fuselage station (ft):</td>
</tr>
</tbody>
</table>

Figure 3.18. NOTAR New File Menu.

*** STABILITY AND CONTROL MENU ***
*** ADDITIONAL PARAMETERS (3 of 3) ***

CG location and Inertias/fuselage parameters
1. cg ht. above waterline 2. cg fuselage station
3. cg posn rt of buttline 4. Ixx
5. Iyy 6. Izz
7. Ixz 8. fuselage downwash ratio

NOTAR if available (enter zeros if using tail rotor)
9. height above waterline 10. boom fuselage station
11. boom position left ref 12. NOTAR diameter
13. swirl angle at boom 14. NOTAR max force
15. thruster fuselage station

Rigging
16. B1 main/in defl (del e) 17. A1 main/in defl (del a)
18. theta0m/in defl (del c) 19. theta0t/pedal defl (del r or p)
20. NOTAR sleeve twist/defl 21. max rudder defl

0. NO CHANGES
Input the parameter to change:

Figure 3.19. Additional Parameters Menu, Screen 3.
H. RIGGING

The control input matrix \([B]\) is automatically computed by the program, but the user must establish control rigging slopes and control limits. User inputs are entered in the Stability and Control Routine New File menu (Fig. 3.20) and can be changed in the Stability and Control Additional Parameters Menu screen 3 (Fig. 3.18).

Rigging inputs are echoed in the output listings, but the effects of the rigging are best seen in the Key Control Parameters screen, screen view selection 5, and State Matrices Representations, screen view selection 3.

| Long cyclic pitch per inch defl (deg/in): |
| Lateral cyclic pitch per inch defl (deg/in): |
| Collective pitch per inch defl (deg/in): |
| Tail rotor pitch change per inch defl or percentage of twist |
| Enter 0 (zero) if using NOTAR (deg/in or deg/deg of twist): |
| Maximum deflection of anti-torque from neutral for NOTAR, Enter 0 (zero) if using tail rotor (deg or inch travel): |
| Displacement of anti-torque control until full rudder deflection. Enter 0 (zero) if rudder is fixed (deg or inch travel): |

Figure 3.20. Rigging New File Menu.
IV. SOFTWARE USE WITH EXAMPLES

A. THE PROUTY HELICOPTER

The program is initiated by a program call from the JANRAD program. After entering MATLAB, the command JANRAD is entered and the opening menu appears (Fig. 4.1).

Do you want to edit an existing file or create a new one?
1. edit existing file  2. create new file  

Figure 4.1. Opening Menu.

If entering a new file, JANRAD will ask for the required input. When basic data entry is complete, a screen appears requesting a filename for the data. Enter a six letter filename. This allows JANRAD to concatenate additional letters to the filename for specific output files (Fig. 4.2).

If editing an old file, loading instructions appear (Fig. 4.3). The filename is entered as instructed. Data entry is performed by selecting the variable to change, then making the appropriate

*** SAVE INSTRUCTIONS ***

A. Save the new data to a specified file name.
B. Do not use an extension or quotations.
C. Use letter/number combinations of 6 characters or less.
D. The file will be saved with a "mat" extension.

ex: dsgn_2

E. If you made no changes, press < Enter >, the file will be saved with the original name.

save file as:

Figure 4.2. Save Screen.
entry. If <ENTER> is pressed without entering a value, the last number assigned to the variable is retained. This value is displayed on the screen during the procedure.

### *** LOAD INSTRUCTIONS ***

A. Input the name of the file to edit.
B. The file was saved in your previous session with a "mat" extension.
C. Do not include the extension or quotations.

ex: dsgn1

name of input file:

Figure 4.3. Load Instructions.

After pressing <ENTER>, a menu appears asking which routine to run (Fig. 4.4). When Stability and Control is selected, three additional input screens appear for further data entry. When this is complete, another save menu appears. A new filename can be entered, or <ENTER> can be pressed to save data under the last filename.

### *** EXECUTION MENU ***

1. Rotor Performance Analysis
2. Stability and Control Analysis
3. Rotor Dynamics Analysis
4. Change data
5. Quit

Enter a 1, 2, 3, 4, or 5 >>

Figure 4.4. Execution Menu.

After data entry is complete, the program performs a stability and control analysis then prompts the user for desired output format (Fig. 4.5). If the user wants hard copies of the root loci or the control Bode plots, the plots must be viewed on the screen first to allow MATLAB to create the graph META files for printing.
Do you want the results displayed on screen?

NOTE: if you want a hard copy of the plots, you must select (1) and view them on the screen first.

1. yes  2. no  >>

Figure 4.5. Data Display.

Output from the program is in six basic categories as shown in Fig. 4.6. To demonstrate the output features, the Prouty sample helicopter at a hover [Ref. 2] will be used as an example.

*** STABILITY AND CONTROL PROGRAM ***
*** SCREEN VIEW MENU ***

What do you want to see?

1. Input data.
2. Calculated data.
3. State Matrices.
4. Eigenvalues of the plants and plots of the roots.
5. Key control parameters.
6. Open loop transfer plots.

0. Exit screen view.

Enter a number;

Figure 4.6. Screen View Menu.

Selection 1, Input Data, is self explanatory, and will be skipped in the interest of brevity.

Selection 2, Calculated Data, has two screens. The first screen (Fig. 4.7) gives the calculated main rotor parameters of interest. Note that since the helicopter is hovering, many of the quantities are equal to or near zero. The second screen has calculated data for the anti torque system (Fig. 4.8). Again note that many parameters are zero since the aircraft has no forward velocity.

Selection 3 provides the state matrix representations of the longitudinal, lateral and coupled plants as well as the input matrices (Figs. 4.9, 4.10, and 4.11). The states are listed under each heading for convenience. These matrices can be used after exiting the program to simulate aircraft response to impulses, steps or any other variety of input. These matrices can also be used to develop
response to impulses, steps or any other variety of input. These matrices can also be used to develop state feedback control for the aircraft.

Selection 4 displays the eigenvalues and the root loci of the uncoupled and coupled plants. Before doing this, an instruction screen is displayed to assist in understanding the format of the output (Fig. 4.12).

The second screen displays the longitudinal and the lateral uncoupled eigenvalues, then immediately plots the roots (Fig. 4.13). The plots are META filed by MATLAB after display. To recover them on a hard copy, one must use the Graphics Post Processing (GPP) utility after exiting.
response to impulses, steps or any other variety of input. These matrices can also be used to develop state feedback control for the aircraft.

Selection 4 displays the eigenvalues and the root loci of the uncoupled and coupled plants. Before doing this, an instruction screen is displayed to assist in understanding the format of the output (Fig. 4.12).

The second screen displays the longitudinal and the lateral uncoupled eigenvalues, then immediately plots the roots (Fig. 4.13). The plots are META filed by MATLAB after display. To recover them on a hard copy, one must use the Graphics Post Processing (GPP) utility after exiting.
### Longitudinal uncoupled plant (A or F depending on notation)

**States are** \([u \ w \ q \ theta]\)

\[
\begin{array}{cccc}
-0.0106 & 0 & 2.5068 & -32.2000 \\
0 & -0.2698 & 0 & 0.0002 \\
0.0038 & 0 & -0.8878 & 0 \\
0 & 0 & 0.9977 & 0
\end{array}
\]

### Longitudinal uncoupled input matrix (B or G depending on notation)

**Inputs are** \([\text{longitudinal cyclic, collective, lateral cyclic, pedals}]\)

\[
\begin{array}{cccc}
1.2248 & -0.0000 & -0.0738 & 0 \\
0 & -5.8773 & 0 & 0 \\
-0.4338 & 0.0000 & 0.0261 & -0.0238 \\
0 & 0 & 0 & 0
\end{array}
\]

Press any key to continue...

**Figure 4.9. Longitudinal State Matrix Representation.**

### Lateral/directional uncoupled plant (A or F depending on notation)

**States are** \([v \ p \ phi \ r \ psi]\)

\[
\begin{array}{cccccc}
-0.0305 & -2.6260 & 32.1267 & 0.7353 & 0 \\
-0.0064 & -1.0274 & 0 & 0.0783 & 0 \\
0 & 1.0000 & 0 & -0.0000 & 0 \\
0.0130 & 0.0783 & 0 & -3.3360 & 0 \\
0 & 0 & 0 & 1.0000 & 0
\end{array}
\]

### Lateral/directional uncoupled input matrix (B or G depending on notation)

**Inputs are** \([\text{longitudinal cyclic, collective, lateral cyclic, pedals}]\)

\[
\begin{array}{cccccc}
0.1200 & 0.0000 & 0.7533 & -1.9736 \\
0.0486 & 0.0000 & 0.3049 & -0.2101 \\
0 & 0 & 0 & 0 \\
0 & 0.3001 & 0 & 1.2959 \\
0 & 0 & 0 & 0
\end{array}
\]

Press any key to continue...

**Figure 4.10. Lateral/Directional State Matrix Representation.**

The first root in the lateral directional plant is an artifact of the system representation and has no physical meaning. The other modes are the Dutch-roll and spiral modes. The longitudinal modes shown are the short period and phugoid.
**Coupled plant (A or F depending on notation)**

States are \([u \ w \ q \ \theta \ v \ \phi \ r \ \text{psl}]\)

<table>
<thead>
<tr>
<th>Columns 1 through 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.0106 0 2.5068 -32.2000 -0.0057 -0.7469 0</td>
</tr>
<tr>
<td>0 -0.2698 0 0.0002 0 0 2.1710</td>
</tr>
<tr>
<td>0 0 -0.8878 0 0.0013 0.2645 0</td>
</tr>
<tr>
<td>0 0 0.9977 0 0 0 0</td>
</tr>
<tr>
<td>0.0057 0 -0.7469 -0.0000 -0.0305 -2.6260 32.1267</td>
</tr>
<tr>
<td>0.0023 0 -0.3023 0 -0.0064 -1.0274 0</td>
</tr>
<tr>
<td>0 0 0.0000 0 0 1.0000 0</td>
</tr>
<tr>
<td>0 -0.0198 0 0 0.0130 0.0783 0</td>
</tr>
<tr>
<td>0 0 0 0 0 0 0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Columns 8 through 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
</tr>
<tr>
<td>0 0</td>
</tr>
<tr>
<td>0.0266 0</td>
</tr>
<tr>
<td>0.0674 0</td>
</tr>
<tr>
<td>0.7353 0</td>
</tr>
<tr>
<td>0.0783 0</td>
</tr>
<tr>
<td>-0.0000 0</td>
</tr>
<tr>
<td>-3.3360 0</td>
</tr>
<tr>
<td>1.0000 0</td>
</tr>
</tbody>
</table>

**Coupled input matrix (B or G depending on notation)**

Inputs are \([\text{longitudinal cyclic, collective, lateral cyclic, pedals}]\)

<table>
<thead>
<tr>
<th>1.2248 -0.0000 -0.0738 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 -5.8773 0 0</td>
</tr>
<tr>
<td>-0.4338 0.0000 0.0261 -0.0238</td>
</tr>
<tr>
<td>0 0 0 0</td>
</tr>
<tr>
<td>0.1200 0.0000 0.7533 -1.9736</td>
</tr>
<tr>
<td>0.0486 0.0000 0.3049 -0.2101</td>
</tr>
<tr>
<td>0 0 0 0</td>
</tr>
<tr>
<td>0 0.3001 0 1.2959</td>
</tr>
<tr>
<td>0 0 0 0</td>
</tr>
</tbody>
</table>

**Figure 4.11. Coupled State Matrix Representation.**

Comparing the root estimates of the longitudinal and lateral plants with the roots of the coupled plant in the third screen shows close correlation.

The plots of the eigenvalues are in the Argand plane (Figs 4.15, 4.16, and 4.17). Note that the plant has some unstable roots. This is typical of helicopters. Very rarely are all the modes stable without some form of stability augmentation.
After you view the root loci plot, a meta file is made. When you are done a screen will tell you the file names of the meta files. To get a hard copy of the plots, you must graphics post process (GPP) the files for your particular printer set-up then, print.

NOTE: If ALL roots are real, MATLAB will NOT plot them in the Argand plane, but will plot the root against its position in the vector (e.g. the first root would be plotted as (1,root))

press any key to continue...

Figure 4.3. Eigenvalue and Root Loci Instruction Screen.

*** EIGENVALUES ***

prouth

Uncoupled

Longitudinal plant

<table>
<thead>
<tr>
<th>Root</th>
<th>wn</th>
<th>damping</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3.0155</td>
<td>0.0155</td>
<td>1.0000</td>
</tr>
<tr>
<td>0.0585 + 0.3398i</td>
<td>0.3448</td>
<td>-0.1698</td>
</tr>
<tr>
<td>0.0585 - 0.3398i</td>
<td>0.3448</td>
<td>-0.1698</td>
</tr>
<tr>
<td>-0.2698</td>
<td>0.2698</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Lateral/Directional plant

<table>
<thead>
<tr>
<th>Root</th>
<th>wn</th>
<th>damping</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-1.0000</td>
</tr>
<tr>
<td>-3.3433</td>
<td>3.3433</td>
<td>1.0000</td>
</tr>
<tr>
<td>-1.1784</td>
<td>1.1784</td>
<td>1.0000</td>
</tr>
<tr>
<td>0.0639 + 0.4025i</td>
<td>0.4075</td>
<td>-0.1568</td>
</tr>
<tr>
<td>0.0639 - 0.4025i</td>
<td>0.4075</td>
<td>-0.1568</td>
</tr>
</tbody>
</table>

press any key to continue...

Figure 4.13. Uncoupled Eigenvalues.

The final screen for this section (Fig. 4.18) tells the user the filenames for the graphs. The user can write these down, or refer to Appendix E for the filenames when hard copies are desired.

The Key Control Parameters screen 2 (Figs 4.19 and 4.20), selection 5, provides numbers that can be used to determine Cooper-Harper pilot ratings and compliance with MIL-H-8501. The screen also tells the amount of cross coupling between flapping modes of the main rotor system. Though
Table with Eigenvalues:

<table>
<thead>
<tr>
<th>Root</th>
<th>wn</th>
<th>damping</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-1.0000</td>
</tr>
<tr>
<td>-3.3432</td>
<td>3.3432</td>
<td>1.0000</td>
</tr>
<tr>
<td>-1.0970 + 0.2583i</td>
<td>1.1270</td>
<td>0.9734</td>
</tr>
<tr>
<td>-1.0970 - 0.2583i</td>
<td>1.1270</td>
<td>0.9734</td>
</tr>
<tr>
<td>0.0760 + 0.3983i</td>
<td>0.4054</td>
<td>-0.1873</td>
</tr>
<tr>
<td>0.0760 - 0.3983i</td>
<td>0.4054</td>
<td>-0.1873</td>
</tr>
<tr>
<td>0.0461 + 0.3582i</td>
<td>0.3611</td>
<td>-0.1276</td>
</tr>
<tr>
<td>0.0461 - 0.3582i</td>
<td>0.3611</td>
<td>-0.1276</td>
</tr>
<tr>
<td>-0.2689</td>
<td>0.2689</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

Figure 4.14. Coupled Eigenvalues.

Figure 4.15. Longitudinal Roots.

Figure 4.18. Eigenvalues Closing Screen.

Plots are saved under the following filenames:

Longitudinal roots - rootlon.met
Lateral/Directional roots - rootlat.met
Coupled roots - rootcoup.met

press any key to continue...
MIL-H-8501 has been superseded by ADS-33C, the numbers still provide information on the damping characteristics and control power of the helicopter.

The last selection, open loop transfer plots, provides the transfer function from a control input to a state output. The purpose of these plots is to allow the designer to design automatic flight control systems and analyze the best techniques for controlling unstable modes. For example, to see the

---

Figure 4.16. Lateral Directional Roots.

Figure 4.17. Coupled Roots.
The key control parameters for pitch, roll, and yaw are as follows:

**Designed damping**
- Pitch: \(-35513.5\) ft-lbs/(rad/sec)
- Roll: \(-35957.8\) ft-lbs/(rad/sec)
- Yaw: \(-116759.1\) ft-lbs/(rad/sec)

**Control Power**
- Pitch: \(-17351.2\) ft-lbs/in
- Roll: \(10672.0\) ft-lbs/in
- Yaw: \(45355.3\) ft-lbs/in

Press any key to continue...

---

**Effects of Collective Pitch on Forward Velocity, u.** Select the longitudinal set of plots from the screen (Fig. 4.21) and page through the plots until you get to "Collective Pitch to U." It can be seen as shown in Fig. 4.22, that there is little control gain from collective to forward velocity at a hover. Obviously, collective would be a poor choice for controlling fore and aft motion at a hover.

If one looks at "Longitudinal Cyclic to U," it can be seen that there is high gain from input to output, especially at the short period frequency (Fig. 4.23). This shows that longitudinal cyclic is probably a good choice for controlling fore and aft motion.
After you view a bode plot of the transfer function from input to state output, a meta file is made. When you exit, a screen will tell you the file names of the meta files. To get a hard copy of the plots, you must graphics post process (GPP) the files for your particular printer set-up then print.

While viewing a plot, press any key to go to the next plot

Do you want to see longitudinal or lateral/directional plots?

1. Longitudinal (eight plots total).
2. Lateral Directional (ten plots total).

Enter a number:

Figure 4.21. Open Loop Transfer Plots, Screen 1.

Figure 4.22. Collective to Forward Velocity.

Another observation from this plot is that at low frequencies (e.g. when you hold the cyclic steady) there is good control response. This is consistent with flight experience. When the cyclic is displaced forward, the aircraft pitches forward as the cyclic is moved, until some equilibrium pitch attitude is reached. When the cyclic is stopped, the attitude remains fixed and the aircraft moves forward. Likewise, when the cyclic is moved back and forth very rapidly, the aircraft may vibrate, but the pitch attitude remains essentially fixed because the aircraft cannot move as quickly as the
control input, hence, the aircraft does not change its average velocity.

These effects can be seen in the control responses of all the controls. In the lateral-directional plant, the response from lateral cyclic to sideslip, $v$, has the same type of peak near the Dutch-roll mode, but response above the Dutch-roll frequency is reduced (Fig. 4.24).

Figure 4.23. Longitudinal Cyclic to Forward Velocity.

Figure 4.24. Lateral Cyclic to Lateral Velocity (Sideslip).
A complete set of Bode plots for the prouty helicopter can be found in Appendix G. The shapes of these curves is typical of helicopters with single main rotors and a tail rotor.

Hard copies of the plots are obtained in the same manner as the root locus plots. A complete list of command response META files is located in Appendix E.

The output from the screens is written to a diary after exiting the Screen View menu. The Output Data Instruction menus in Figs. 4.25 through 4.27 explain the data recovery procedures.

### OUTPUT DATA INSTRUCTIONS (screen 1 of 3)

Because this subroutine generates a large number of single value data not shown on the output screen, a text file VARLIST.TXT is on this disk which lists the variable names for all the stability derivatives. Stability derivative contributions for all major aircraft components can be found by reading the text file VARLIST.TXT, then asking MATLAB the variable name corresponding to the derivative.

Figure 4.25. Output Data Instructions, Screen 1.

### OUTPUT DATA INSTRUCTIONS (screen 2 of 3)

A. Data from the output screen saved to a file named: "filenamel.stb"

B. This is a text file, use the TYPE command to view the file or use a text editor to view/print the file.

C. Matrix and vector data saved to a default file named: "mstabdat.mat"

D. This is a ".mat" binary file, use the LOAD command to retrieve the data for plotting.

E. Rename "mstabdat.mat" to another ".mat" file. The file "mstabdat.mat" will be overwritten when the program is executed.

F. Do not rename the file as "filenamel.mat" The file "filenamel.mat" is already on disk and used for future editing.

Figure 4.26. Output Data Instructions, Screen 2.
A. Single value data saved to a default file named:
   "vstabdat.mat"

B. This is a ".mat" binary file, use the LOAD command to retrieve the data for plotting.

C. Rename "vstabdat.mat" to another ".mat" file.
   The file "vstabdat.mat" will be overwritten when the program is executed.

D. Do not rename the file as "filename1.mat"
   The file "filename1.mat" is already on disk and used for future editing.

*** END STABILITY AND CONTROL ROUTINE ***

Figure 4.27. Output Data Instructions, Screen 3.

B. A HELICOPTER WITH NOTAR™

A final example will be a notional helicopter designed for the 1993 annual American Helicopter Society (AHS) Design Competition, the Arapaho. The forward flight mode of the helicopter will be used. Since the procedures for entering and displaying data have been covered, only the salient points of the design will be addressed.

The helicopter incorporated a NOTAR™ tailboom. As a result, the tail rotor derivatives are unusual as are the command transfer Bode plots.

The Output file will have a zero displayed for all tail rotor computations except the tail rotor thrust. The thrust displayed in this case will be the NOTAR™ thruster force (Fig 4.28).

By comparing the longitudinal control matrices with and without a tail rotor (Fig. 4.10 and 4.29) it can be seen that a change in tail rotor thrust has a pitching moment contribution (look at the B(3,4) term). Because the tail rotor rotates about an axis that is parallel to the Y control axis, this contribution is to be expected with a change in tail rotor thrust requirement. In contrast, the NOTAR™ does not have this contribution because the thruster fan rotates on an axis parallel to the X control axis.
**CALCULATED DATA (screen 2 of 2)**

*arapac*

Tail Rotor (zero if NOTAR)

- tail rotor thrust = 264.2 lbs
- advance ratio = 0.0
- inflow parameter = 0.000
- Rotor coning angle = 0.0 degs
- lateral flapping = 0.00 degs
- longitudinal flapping = 0.00 degs
- Lock number = 0.0

press any key to continue...

Figure 4.28. Tail Rotor Output.

<table>
<thead>
<tr>
<th>States are ([u \ w \ q \ \theta] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-0.0363 \quad 0.0567 \quad 3.3538 \quad -32.1459)</td>
</tr>
<tr>
<td>(0.0685 \quad 0.3972 \quad 84.6890 \quad -1.8657)</td>
</tr>
<tr>
<td>(-0.0098 \quad -0.0204 \quad -0.5489 \quad 0)</td>
</tr>
<tr>
<td>(0 \quad 0 \quad 0.9999 \quad 0)</td>
</tr>
</tbody>
</table>

Longitudinal uncoupled input matrix (B or G depending on notation)

<table>
<thead>
<tr>
<th>Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]</th>
</tr>
</thead>
<tbody>
<tr>
<td>(7.0005 \quad -0.5855 \quad -2.0435 \quad 0)</td>
</tr>
<tr>
<td>(26.0454 \quad -10.9228 \quad 0 \quad 0)</td>
</tr>
<tr>
<td>(-1.2115 \quad 0.5179 \quad 0.3537 \quad 0)</td>
</tr>
<tr>
<td>(0 \quad 0 \quad 0 \quad 0)</td>
</tr>
</tbody>
</table>

press any key to continue...

Figure 4.29. Arapaho Control Matrix.

Figure 4.30 demonstrates the difference in flight control response with airspeed. In the hover case, longitudinal cyclic has little direct effect on vertical velocity. But notice that in the forward flight regime, a longitudinal deflection has a great effect on vertical velocity. A zoom climb can be done in forward flight but not at a hover.
Open loop response Longitudinal Cyclic to $W$, Cruise

Figure 4.30. Longitudinal Cyclic to Vertical Velocity.
V. CONCLUSIONS AND RECOMMENDATIONS.

A. CONCLUSIONS

The aerodynamicist should not be intimidated by the complexity of helicopters. The complexities of determining the stability characteristics of a helicopter are formidable, but helicopter flight dynamics are essentially no different than fixed wing flight dynamics. Modern software reduces helicopter analysis to a routine procedure, making the analysis of these characteristics much less time consuming.

B. RECOMMENDATIONS

To improve the fidelity of JANRAD Stability and Control, the following actions are recommended.

a) Expand the state matrix to include $a_{iH}, b_{iM}, C_I/\sigma_M, Ch/\sigma_M, Cq/\sigma_M, \mu, \lambda^z a_{iH}, b_{iH}, C_I/\sigma_I, Ch/\sigma_I, Cq/\sigma_I, \mu_I$ and $\lambda_I$ for the tail and main rotors. This representation would allow students to run simulations on MATLAB that could show rotor flapping, cyclic pitch changes, and fluctuations in the coefficients of torque, thrust and H-force.

b) Expand the input screens to allow selecting different lift and drag characteristics for the fuselage. Even though this type of information is derived empirically, studies of existing helicopter with appropriate flight test data could be done without the added artificiality of a particular fuselage configuration. JANRAD currently uses fuselage aerodynamic characteristics of the Prouty sample helicopter as outlined in section III. C.

c) Improve the fidelity of the tail rotor trim solution. The method used to derive the tail rotor trim solution is extremely rough compared to the trim solution of the main rotor.
d) Expand the wing inputs to include dihedrally mounted wings and wing sweep. Even though most helicopters do not use wings for lift or control, compound helicopters are likely to require a higher fidelity wing solution.
APPENDIX A. NOTATION.

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Definition</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Lift curve slope</td>
<td>1/rad</td>
</tr>
<tr>
<td>a₀</td>
<td>Coning angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>a₁ₘ</td>
<td>First harmonic of longitudinal flapping wrt shaft axis</td>
<td>rad, deg</td>
</tr>
<tr>
<td>b</td>
<td>Number of blades</td>
<td></td>
</tr>
<tr>
<td>b₁ₘ</td>
<td>First harmonic of lateral flapping wrt shaft axis</td>
<td>rad, deg</td>
</tr>
<tr>
<td>c</td>
<td>Chord</td>
<td>ft</td>
</tr>
<tr>
<td>cₐ</td>
<td>Coefficient of drag</td>
<td></td>
</tr>
<tr>
<td>cₙ</td>
<td>Center of gravity</td>
<td></td>
</tr>
<tr>
<td>cₗ</td>
<td>Coefficient of lift</td>
<td></td>
</tr>
<tr>
<td>cₘₙ</td>
<td>Coefficient of moment</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>Hinge offset</td>
<td>ft</td>
</tr>
<tr>
<td>f</td>
<td>Equivalent flat plate area</td>
<td>ft²</td>
</tr>
<tr>
<td>g</td>
<td>Acceleration due to gravity</td>
<td>ft/sec²</td>
</tr>
<tr>
<td>i</td>
<td>Incidence</td>
<td>rad, deg</td>
</tr>
<tr>
<td>l</td>
<td>Fuselage station</td>
<td>ft</td>
</tr>
<tr>
<td>p</td>
<td>Roll rate</td>
<td>rad/sec</td>
</tr>
<tr>
<td>q</td>
<td>Pitch rate</td>
<td>rad/sec</td>
</tr>
<tr>
<td>q</td>
<td>Dynamic pressure</td>
<td>lb/ft</td>
</tr>
<tr>
<td>r</td>
<td>Blade element radius</td>
<td>ft</td>
</tr>
<tr>
<td>r</td>
<td>Yaw rate</td>
<td>rad/sec</td>
</tr>
<tr>
<td>u</td>
<td>Forward velocity</td>
<td>ft/sec</td>
</tr>
<tr>
<td>u</td>
<td>Input vector</td>
<td></td>
</tr>
<tr>
<td>v</td>
<td>Induced velocity</td>
<td>ft/sec</td>
</tr>
<tr>
<td>v</td>
<td>Lateral velocity</td>
<td>ft/sec</td>
</tr>
<tr>
<td>w</td>
<td>Vertical velocity</td>
<td>ft/sec</td>
</tr>
<tr>
<td>y</td>
<td>Output vector</td>
<td></td>
</tr>
<tr>
<td>y</td>
<td>Buttline position</td>
<td>ft</td>
</tr>
<tr>
<td>z</td>
<td>Waterline position</td>
<td>ft</td>
</tr>
<tr>
<td>A</td>
<td>Plant matrix</td>
<td></td>
</tr>
<tr>
<td>A</td>
<td>Area of disk</td>
<td>ft²</td>
</tr>
<tr>
<td>Aₐ</td>
<td>Blade area</td>
<td>ft²</td>
</tr>
<tr>
<td>A₁ₘ</td>
<td>First lateral harmonic of blade feathering</td>
<td>rad, deg</td>
</tr>
<tr>
<td>B</td>
<td>Input matrix</td>
<td></td>
</tr>
<tr>
<td>B₁ₘ</td>
<td>First longitudinal harmonic of blade feathering</td>
<td>rad, deg</td>
</tr>
<tr>
<td>C</td>
<td>Output matrix</td>
<td></td>
</tr>
<tr>
<td>Cₐ</td>
<td>Coefficient of horizontal force</td>
<td></td>
</tr>
<tr>
<td>Cₙ</td>
<td>Coefficient of power</td>
<td></td>
</tr>
<tr>
<td>Symbol</td>
<td>Description</td>
<td>Unit</td>
</tr>
<tr>
<td>--------</td>
<td>-------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>$C_t$</td>
<td>Coefficient of torque</td>
<td></td>
</tr>
<tr>
<td>$C_i$</td>
<td>Coefficient of thrust</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>Direct transition matrix</td>
<td></td>
</tr>
<tr>
<td>D.L.</td>
<td>Disk loading</td>
<td>lb/ft²</td>
</tr>
<tr>
<td>D</td>
<td>Drag</td>
<td>lb</td>
</tr>
<tr>
<td>GW</td>
<td>Gross weight</td>
<td>lb</td>
</tr>
<tr>
<td>H</td>
<td>Horizontal force on rotor</td>
<td>lb</td>
</tr>
<tr>
<td>$l_b$</td>
<td>Flapping moment of inertia</td>
<td>slug ft²</td>
</tr>
<tr>
<td>$l_m$</td>
<td>Rolling moment of inertia</td>
<td>slug ft²</td>
</tr>
<tr>
<td>$l_y$</td>
<td>Pitching moment of inertia</td>
<td>slug ft²</td>
</tr>
<tr>
<td>L</td>
<td>Lift</td>
<td>lb</td>
</tr>
<tr>
<td>M</td>
<td>Pitching moment</td>
<td>ft-lb</td>
</tr>
<tr>
<td>N</td>
<td>Yawing moment</td>
<td>ft-lb</td>
</tr>
<tr>
<td>Q</td>
<td>Torque</td>
<td>ft-lb</td>
</tr>
<tr>
<td>R</td>
<td>Blade radius</td>
<td>ft</td>
</tr>
<tr>
<td>R</td>
<td>Rolling moment</td>
<td>ft-lb</td>
</tr>
<tr>
<td>T</td>
<td>Thrust</td>
<td>lb</td>
</tr>
<tr>
<td>X</td>
<td>State vector</td>
<td></td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Angle of attack</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\beta$</td>
<td>Blade flapping angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Sideslip angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\gamma_c$</td>
<td>Lock number</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\delta$</td>
<td>Pitch-flap coupling angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\epsilon$</td>
<td>Downwash angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\eta$</td>
<td>Sidewash angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Pitch angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\Theta$</td>
<td>Blade pitch</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\Theta_i$</td>
<td>Blade twist</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>Inflow ratio wrt swashplate</td>
<td></td>
</tr>
<tr>
<td>$\lambda'$</td>
<td>Inflow ratio wrt tip path plane</td>
<td></td>
</tr>
<tr>
<td>$\mu$</td>
<td>Tip speed ratio</td>
<td></td>
</tr>
<tr>
<td>$\rho$</td>
<td>Density of air</td>
<td>slug/ft³</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Rotor Solidity</td>
<td></td>
</tr>
<tr>
<td>$\Phi$</td>
<td>Roll angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\phi$</td>
<td>Inflow angle</td>
<td>rad</td>
</tr>
<tr>
<td>$\Psi$</td>
<td>Yaw angle</td>
<td>rad, deg</td>
</tr>
<tr>
<td>$\psi$</td>
<td>Rotational velocity of rotor</td>
<td>rad/sec</td>
</tr>
<tr>
<td>$\Omega$</td>
<td>Rotor tip speed</td>
<td>ft/sec</td>
</tr>
<tr>
<td>Subscript</td>
<td>Definition</td>
<td></td>
</tr>
<tr>
<td>-----------</td>
<td>---------------------------------</td>
<td></td>
</tr>
<tr>
<td>0 thru 1</td>
<td>Blade position in r/R</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Nominal</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>Reference</td>
<td></td>
</tr>
<tr>
<td>b</td>
<td>Per blade</td>
<td></td>
</tr>
<tr>
<td>i</td>
<td>Induced</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>Fuselage</td>
<td></td>
</tr>
<tr>
<td>H</td>
<td>Horizontal stabilizer</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>Main rotor</td>
<td></td>
</tr>
<tr>
<td>N</td>
<td>NOTAR™</td>
<td></td>
</tr>
<tr>
<td>Q</td>
<td>Torque</td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>Tail rotor</td>
<td></td>
</tr>
<tr>
<td>T</td>
<td>Thrust</td>
<td></td>
</tr>
<tr>
<td>TPP</td>
<td>Tip path plane</td>
<td></td>
</tr>
<tr>
<td>V</td>
<td>Vertical stabilizer</td>
<td></td>
</tr>
</tbody>
</table>
APPENDIX B. NASA STATE SPACE REPRESENTATION.

NASA state space representation from Ref. 3.

\[
X = (\text{deg}, \text{deg}, \text{deg}, \text{deg}, \text{deg}, \text{deg})^T
\]

\[
\dot{X} = (\ddot{X}, \dddot{X}, \ldots)^T
\]

\[
P = \begin{bmatrix}
    0 & 1 & 0 & 0 & 0 & 0 \\
    -\sin \phi & \cos \phi & 0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 1 & 0 & 0 \\
    0 & 0 & -\sin \phi & \cos \phi & 0 & 0 \\
    0 & 0 & 0 & 0 & 1 & 0 \\
    0 & 0 & 0 & 0 & 0 & 1 \\
\end{bmatrix}
\]

\[
l_c = (l_{c1}, l_{c2})
\]

\[
\begin{bmatrix}
l_{c1} \\
l_{c2} \\
l_{c1} \\
l_{c2} \\
l_{c1} \\
l_{c2}
\end{bmatrix}
= \begin{bmatrix}
    l_{c1} \\
l_{c1} \\
l_{c1} \\
l_{c2} \\
l_{c2} \\
l_{c2}
\end{bmatrix}
\]
APPENDIX C. SUMMARY OF STABILITY DERIVATIVES.

A summary of stability derivatives extracted from Ref. 2.

### Basic Rotor Derivatives near Hover

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$(\frac{\partial \mu}{\partial \lambda})<em>\alpha, (\frac{\partial \gamma}{\partial \lambda})</em>\alpha, (\frac{1}{\Omega R})_\alpha$</td>
<td>$(\frac{\partial C_{\mu}/\alpha}{\partial \lambda}) = \frac{-2 \frac{\mu}{\Omega R}}{4 \left( 1 - \frac{\mu}{\Omega R} \right) \left( 1 - \frac{\epsilon}{\Omega R} \right)}$</td>
</tr>
<tr>
<td>$(\frac{\partial \delta_{A1}}{\partial \mu})_0, (\frac{1}{\Omega R})_0$</td>
<td>$(\frac{\partial \delta_{A1}}{\partial \lambda}, \frac{\partial \delta_{A1}}{\partial \mu}) = \frac{-12 \frac{\mu}{\Omega R}}{\gamma \left( 1 - \frac{\epsilon}{\Omega R} \right)}$</td>
</tr>
<tr>
<td>$(\frac{\partial \delta_{B1}}{\partial \mu})_0, (\frac{1}{\Omega R})_0$</td>
<td>$(\frac{\partial \delta_{B1}}{\partial \lambda}, \frac{\partial \delta_{B1}}{\partial \mu}) = \frac{-144 \frac{\mu}{\Omega R}}{\gamma \left( 1 - \frac{\epsilon}{\Omega R} \right)^4}$</td>
</tr>
<tr>
<td>$(\frac{\partial \delta_{M}}{\partial \lambda})<em>\gamma, (\frac{\partial \delta</em>{M}}{\partial \mu})<em>\gamma, (\frac{\partial \delta</em>{N}}{\partial \lambda})<em>\gamma, (\frac{\partial \delta</em>{N}}{\partial \mu})<em>\gamma, (\frac{\partial \delta</em>{R}}{\partial \lambda})<em>\gamma, (\frac{\partial \delta</em>{R}}{\partial \mu})<em>\gamma, (\frac{\partial \delta</em>{T}}{\partial \lambda})<em>\gamma, (\frac{\partial \delta</em>{T}}{\partial \mu})_\gamma$</td>
<td>$(\frac{\partial \delta_{A1}}{\partial \lambda}, \frac{\partial \delta_{A1}}{\partial \mu}) = \frac{-3 \frac{\mu}{\Omega R}}{2 \left( 1 - \frac{\mu}{\Omega R} \right) \left( 1 - \frac{\epsilon}{\Omega R} \right)}$</td>
</tr>
</tbody>
</table>

### Other Derivatives

- Take from hover charts of Chapter 1
- $\frac{\partial C_{\mu}/\alpha}{\partial \theta_0}$
- $\frac{\partial C_{\mu}/\alpha}{\partial \theta_0}$

---

53
<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial x}{\partial i} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{i}}{\partial i} )</td>
</tr>
<tr>
<td>( \frac{\partial y}{\partial i} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{y}}{\partial i} )</td>
</tr>
<tr>
<td>( \frac{\partial z}{\partial i} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{z}}{\partial i} )</td>
</tr>
<tr>
<td>( \frac{\partial x}{\partial j} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{x}}{\partial j} )</td>
</tr>
<tr>
<td>( \frac{\partial y}{\partial j} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{y}}{\partial j} )</td>
</tr>
<tr>
<td>( \frac{\partial z}{\partial j} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{z}}{\partial j} )</td>
</tr>
<tr>
<td>( \frac{\partial x}{\partial \lambda_{1}} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{x}}{\partial \lambda_{1}} )</td>
</tr>
<tr>
<td>( \frac{\partial y}{\partial \lambda_{1}} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{y}}{\partial \lambda_{1}} )</td>
</tr>
<tr>
<td>( \frac{\partial z}{\partial \lambda_{1}} )</td>
<td>( -p_{s}^{1}(\Omega) \cdot \frac{\partial c_{z}}{\partial \lambda_{1}} )</td>
</tr>
</tbody>
</table>

---

**Main Rotor Derivatives near Hover**
<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial R}{\partial A_1} )</td>
<td>( \frac{\partial R}{\partial A_1} ) ( \frac{\partial B_1}{\partial A_1} ) ( \frac{\partial C_1}{\partial A_1} ) ( \frac{\partial D_1}{\partial A_1} ) ( \frac{\partial E_1}{\partial A_1} ) ( \frac{\partial F_1}{\partial A_1} ) ( \frac{\partial G_1}{\partial A_1} ) ( \frac{\partial H_1}{\partial A_1} ) ( \frac{\partial I_1}{\partial A_1} ) ( \frac{\partial J_1}{\partial A_1} ) ( \frac{\partial K_1}{\partial A_1} ) ( \frac{\partial L_1}{\partial A_1} ) ( \frac{\partial M_1}{\partial A_1} ) ( \frac{\partial N_1}{\partial A_1} ) ( \frac{\partial O_1}{\partial A_1} ) ( \frac{\partial P_1}{\partial A_1} ) ( \frac{\partial Q_1}{\partial A_1} ) ( \frac{\partial R_1}{\partial A_1} ) ( \frac{\partial S_1}{\partial A_1} ) ( \frac{\partial T_1}{\partial A_1} ) ( \frac{\partial U_1}{\partial A_1} ) ( \frac{\partial V_1}{\partial A_1} ) ( \frac{\partial W_1}{\partial A_1} ) ( \frac{\partial X_1}{\partial A_1} ) ( \frac{\partial Y_1}{\partial A_1} ) ( \frac{\partial Z_1}{\partial A_1} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial A_2} )</td>
<td>( \frac{\partial R}{\partial A_2} ) ( \frac{\partial B_2}{\partial A_2} ) ( \frac{\partial C_2}{\partial A_2} ) ( \frac{\partial D_2}{\partial A_2} ) ( \frac{\partial E_2}{\partial A_2} ) ( \frac{\partial F_2}{\partial A_2} ) ( \frac{\partial G_2}{\partial A_2} ) ( \frac{\partial H_2}{\partial A_2} ) ( \frac{\partial I_2}{\partial A_2} ) ( \frac{\partial J_2}{\partial A_2} ) ( \frac{\partial K_2}{\partial A_2} ) ( \frac{\partial L_2}{\partial A_2} ) ( \frac{\partial M_2}{\partial A_2} ) ( \frac{\partial N_2}{\partial A_2} ) ( \frac{\partial O_2}{\partial A_2} ) ( \frac{\partial P_2}{\partial A_2} ) ( \frac{\partial Q_2}{\partial A_2} ) ( \frac{\partial R_2}{\partial A_2} ) ( \frac{\partial S_2}{\partial A_2} ) ( \frac{\partial T_2}{\partial A_2} ) ( \frac{\partial U_2}{\partial A_2} ) ( \frac{\partial V_2}{\partial A_2} ) ( \frac{\partial W_2}{\partial A_2} ) ( \frac{\partial X_2}{\partial A_2} ) ( \frac{\partial Y_2}{\partial A_2} ) ( \frac{\partial Z_2}{\partial A_2} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial A_3} )</td>
<td>( \frac{\partial R}{\partial A_3} ) ( \frac{\partial B_3}{\partial A_3} ) ( \frac{\partial C_3}{\partial A_3} ) ( \frac{\partial D_3}{\partial A_3} ) ( \frac{\partial E_3}{\partial A_3} ) ( \frac{\partial F_3}{\partial A_3} ) ( \frac{\partial G_3}{\partial A_3} ) ( \frac{\partial H_3}{\partial A_3} ) ( \frac{\partial I_3}{\partial A_3} ) ( \frac{\partial J_3}{\partial A_3} ) ( \frac{\partial K_3}{\partial A_3} ) ( \frac{\partial L_3}{\partial A_3} ) ( \frac{\partial M_3}{\partial A_3} ) ( \frac{\partial N_3}{\partial A_3} ) ( \frac{\partial O_3}{\partial A_3} ) ( \frac{\partial P_3}{\partial A_3} ) ( \frac{\partial Q_3}{\partial A_3} ) ( \frac{\partial R_3}{\partial A_3} ) ( \frac{\partial S_3}{\partial A_3} ) ( \frac{\partial T_3}{\partial A_3} ) ( \frac{\partial U_3}{\partial A_3} ) ( \frac{\partial V_3}{\partial A_3} ) ( \frac{\partial W_3}{\partial A_3} ) ( \frac{\partial X_3}{\partial A_3} ) ( \frac{\partial Y_3}{\partial A_3} ) ( \frac{\partial Z_3}{\partial A_3} )</td>
</tr>
</tbody>
</table>

\( \frac{\partial N}{\partial r} \) \( \frac{\partial N}{\partial \theta_0} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \) \( \frac{\partial N}{\partial \phi} \) \( \frac{\partial N}{\partial \theta} \)
### Tall Rotor Derivatives near Hover

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial Y}{\partial \theta} )</td>
<td>( p_{\lambda_{a}}(\Omega R) \frac{\partial C_{t}/\sigma}{\partial \lambda} )</td>
</tr>
<tr>
<td>( \frac{\partial Y}{\partial \phi} )</td>
<td>( \frac{\partial Y}{\partial \phi} )</td>
</tr>
<tr>
<td>( \frac{\partial Y}{\partial r} )</td>
<td>( \frac{\partial Y}{\partial r} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial N}{\partial \theta} )</td>
<td>( \frac{\partial N}{\partial \theta} )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial \phi} )</td>
<td>( \frac{\partial N}{\partial \phi} )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial r} )</td>
<td>( \frac{\partial N}{\partial r} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial M}{\partial \theta} )</td>
<td>( \frac{\partial M}{\partial \phi} )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial r} )</td>
<td>( \frac{\partial M}{\partial r} )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial \phi} )</td>
<td>( \frac{\partial M}{\partial \phi} )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial r} )</td>
<td>( \frac{\partial M}{\partial r} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial R}{\partial \theta} )</td>
<td>( \frac{\partial R}{\partial \phi} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial r} )</td>
<td>( \frac{\partial R}{\partial r} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial \phi} )</td>
<td>( \frac{\partial R}{\partial \phi} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial r} )</td>
<td>( \frac{\partial R}{\partial r} )</td>
</tr>
</tbody>
</table>

56
## Basic Main Rotor Derivatives in Forward Flight

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{\partial \mu}{\partial \lambda}$</td>
<td>$\frac{1}{\Omega R}$</td>
</tr>
<tr>
<td>$\frac{\partial \lambda'}{\partial \lambda}$</td>
<td>$\Omega R \left[ \frac{\partial C_T/\sigma}{\partial \mu} \right]$</td>
</tr>
<tr>
<td>$\frac{\partial \lambda'}{\partial \lambda}$</td>
<td>$\Omega R \left( 1 + \frac{\partial C_T/\sigma}{\partial \lambda} \right)$</td>
</tr>
<tr>
<td>$\frac{\partial \beta}{\partial \lambda}$</td>
<td>$\frac{1}{\nu}$</td>
</tr>
<tr>
<td>$\frac{\partial C_{n}/\sigma}{\partial \lambda}$, $\frac{\partial C_{T}/\sigma}{\partial \lambda}$</td>
<td>$\frac{C_{T}/\sigma + \frac{e}{R}}{\lambda'}$</td>
</tr>
<tr>
<td>$\frac{\partial s_1}{\partial \gamma}$</td>
<td>$\gamma \Omega \left( 1 - \frac{\epsilon}{R} \right)^2 \left( 1 - \frac{\mu}{2} \right)$</td>
</tr>
<tr>
<td>$\frac{\partial s_1}{\partial \mu}$</td>
<td>$\Omega \left( 1 - \frac{\mu}{2} \right)$</td>
</tr>
<tr>
<td>$\frac{\partial s_1}{\partial A_t}$</td>
<td>$\gamma \frac{12}{R} \left( 1 - \frac{\mu}{4} \right)$</td>
</tr>
<tr>
<td>$\frac{\partial s_1}{\partial B_t}$</td>
<td>$\left( 1 - \frac{\mu}{2} \right)$</td>
</tr>
</tbody>
</table>
### Main Rotor Derivatives in Forward Flight

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial b_1}{\partial \eta} )</td>
<td>( -\frac{1}{\Omega} \left( 1 + \frac{\mu'}{2} \right) + \frac{192 \frac{\lambda}{R}}{\gamma \Omega \left( 1 - \frac{\mu'}{R} \right)^4 \left( 1 - \frac{\mu'}{4} \right)} )</td>
</tr>
<tr>
<td>( \frac{\partial b_1}{\partial \rho} )</td>
<td>( \frac{16}{\gamma \Omega \left( 1 - \frac{\mu'}{R} \right)^4 \left( 1 - \frac{\mu'}{4} \right)} - \frac{12 \frac{\lambda}{R}}{\gamma \Omega \left( 1 - \frac{\mu'}{R} \right) \left( 1 - \frac{\mu'}{4} \right)} )</td>
</tr>
<tr>
<td>( \frac{\partial b_1}{\partial \lambda} )</td>
<td>( \frac{12 \frac{\lambda}{R}}{\gamma \left( 1 - \frac{\mu'}{R} \right)} )</td>
</tr>
<tr>
<td>( \frac{\partial b_1}{\partial B_1} )</td>
<td>( \frac{\frac{3 \frac{\lambda}{R}}{4 \rho R} - \frac{\lambda}{R} \rho R \Omega (\Omega R)^2}{\gamma} )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial a_1}, \frac{\partial R}{\partial b_1} )</td>
<td>( \frac{3 \frac{\lambda}{R}}{4 \rho R} - \frac{\lambda}{R} \rho R \Omega (\Omega R)^2 )</td>
</tr>
</tbody>
</table>

Where:
- \( a_1 \) is the angle of attack
- \( b_1 \) is the pitch angle
- \( \mu' \) is the reduced frequency
- \( \Omega \) is the angular velocity
- \( R \) is the radius
- \( \rho \) is the density
- \( \gamma \) is the specific heat ratio
- \( \lambda \) is the lift coefficient
- \( \lambda' \) is the lift coefficient
- \( \eta \) is the proximity factor
- \( \rho \) is the density
- \( \rho R \Omega (\Omega R)^2 \) is the reduced frequency

---

The formulas above represent the derivatives of the main rotor in forward flight, capturing the effects of changes in various parameters on the rotor's performance. Each derivative is expressed in terms of the derivatives of the angle of attack \( a_1 \) and the pitch angle \( b_1 \).
<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial X}{\partial q} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial X}{\partial \hat{d}} ) \hat{k}_{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial X}{\partial p} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial X}{\partial \hat{p}} ) \hat{k}_{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial X}{\partial \theta_{o}} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial \dot{X}}{\partial \dot{d}<em>{i}} + \frac{\partial \dot{X}}{\partial \dot{d}</em>{i}} \cdot \frac{\partial \dot{X}}{\partial \dot{d}<em>{i}} ) \hat{k}</em>{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial X}{\partial \lambda_{i}} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial X}{\partial \hat{\lambda}<em>{i}} ) \hat{k}</em>{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial X}{\partial \eta_{i}} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial X}{\partial \hat{\eta}<em>{i}} ) \hat{k}</em>{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial Y}{\partial \hat{d}} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial \dot{Y}}{\partial \dot{d}<em>{i}} + \frac{\partial \dot{Y}}{\partial \dot{d}</em>{i}} \cdot \frac{\partial \dot{Y}}{\partial \dot{d}<em>{i}} ) \hat{k}</em>{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial Y}{\partial \hat{p}} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial \dot{Y}}{\partial \dot{d}<em>{i}} \cdot \frac{\partial \dot{Y}}{\partial \dot{d}</em>{i}} ) \hat{k}_{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial Y}{\partial \hat{\lambda}_{i}} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial \dot{Y}}{\partial \dot{d}<em>{i}} \cdot \frac{\partial \dot{Y}}{\partial \dot{d}</em>{i}} ) \hat{k}_{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial Y}{\partial \hat{\eta}_{i}} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial \dot{Y}}{\partial \dot{d}<em>{i}} \cdot \frac{\partial \dot{Y}}{\partial \dot{d}</em>{i}} ) \hat{k}_{\mu} )</td>
</tr>
<tr>
<td>( \frac{\partial \dot{Y}}{\partial \dot{d}_{i}} )</td>
<td>(-pA_{L}(\Omega R)^{T} \frac{\partial C_{W}/\sigma}{\partial d_{i}} ) + ( \frac{\partial \dot{Y}}{\partial \dot{d}<em>{i}} \cdot \frac{\partial \dot{Y}}{\partial \dot{d}</em>{i}} ) \hat{k}_{\mu} )</td>
</tr>
</tbody>
</table>

59
<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial z}{\partial i} )</td>
<td>(- \rho A_i (\Omega R) \left[ \frac{\partial C_i / \sigma}{\partial i} + \frac{\partial C_i / \sigma}{\partial l} \frac{\partial l}{\partial i} \right] )</td>
</tr>
<tr>
<td>( \frac{\partial z}{\partial e} )</td>
<td>(- \rho A_i (\Omega R) \frac{\partial C_i / \sigma}{\partial \lambda i} )</td>
</tr>
<tr>
<td>( \frac{\partial z}{\partial \theta_0} )</td>
<td>(- \rho A_i (\Omega R) \frac{\partial C_i / \sigma}{\partial \theta_0} )</td>
</tr>
<tr>
<td>( \frac{\partial z}{\partial \Pi} )</td>
<td>(- \rho A_i (\Omega R) \frac{\partial C_i / \sigma}{\partial \Pi} \frac{\partial l}{\partial \Pi} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial l} )</td>
<td>(\left( \frac{dR}{d\theta_i} \right) \left( \frac{\partial l}{\partial \theta_i} \right) \left( \frac{\partial l}{\partial \Pi} \right) \left( \frac{\partial l}{\partial \Pi} \right) )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial \tau} )</td>
<td>(- \left( \frac{dR}{d\theta_i} \right) \left( \frac{\partial l}{\partial \theta_i} \right) \frac{\partial \theta}{\partial \Pi} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial \delta} )</td>
<td>(- \left( \frac{dR}{d\theta_i} \right) \left( \frac{\partial l}{\partial \theta_i} \right) \frac{\partial \delta}{\partial \Pi} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial \phi} )</td>
<td>(- \left( \frac{dR}{d\theta_i} \right) \left( \frac{\partial l}{\partial \theta_i} \right) \frac{\partial \phi}{\partial \Pi} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial \theta_1} )</td>
<td>(- \left( \frac{dR}{d\theta_i} \right) \left( \frac{\partial l}{\partial \theta_i} \right) \frac{\partial \theta_1}{\partial \Pi} )</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial \Lambda_1} )</td>
<td>(- \left( \frac{dR}{d\theta_i} \right) \left( \frac{\partial l}{\partial \theta_i} \right) \frac{\partial \Lambda_1}{\partial \Pi} )</td>
</tr>
<tr>
<td>Derivative</td>
<td>Equation</td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
</tr>
<tr>
<td>( \frac{\partial R}{\partial i} )</td>
<td>( \frac{\partial R}{\partial j} \left( \frac{\partial i}{\partial k} + \frac{\partial j}{\partial k} \right) b_k )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial z} )</td>
<td>( \frac{\partial M}{\partial x} \left[ \frac{\partial x}{\partial y} + \frac{\partial y}{\partial z} \right] - \left( \frac{\partial X}{\partial z} \right) b_z + \left( \frac{\partial Z}{\partial z} \right) l_z )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial y} )</td>
<td>( \left( \frac{\partial X}{\partial y} \right) \left( \frac{\partial y}{\partial X} \right) + \left( \frac{\partial Z}{\partial y} \right) \left( \frac{\partial y}{\partial Z} \right) )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial q} )</td>
<td>( \frac{\partial M}{\partial x} \left( \frac{\partial x}{\partial y} \right) - \left( \frac{\partial X}{\partial y} \right) b_y )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial p} )</td>
<td>( \frac{\partial M}{\partial x} \left( \frac{\partial x}{\partial y} \right) - \left( \frac{\partial X}{\partial y} \right) b_y )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial \theta} )</td>
<td>( \frac{\partial M}{\partial x} \left( \frac{\partial x}{\partial y} \right) - \left( \frac{\partial X}{\partial y} \right) b_y )</td>
</tr>
<tr>
<td>( \frac{\partial M}{\partial \phi} )</td>
<td>( \frac{\partial M}{\partial x} \left( \frac{\partial x}{\partial y} \right) - \left( \frac{\partial X}{\partial y} \right) b_y )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial i} )</td>
<td>( \frac{\partial A_i(O,R)^T}{\partial i} \left[ \frac{\partial C_i/O}{\partial \sigma} \frac{\partial \mu}{\partial x} + \frac{\partial C_i/O}{\partial \lambda} \frac{\partial \lambda}{\partial x} \right] )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial j} )</td>
<td>( \frac{\partial A_i(O,R)^T}{\partial j} \left( \frac{\partial C_i/O}{\partial \lambda} \frac{\partial \lambda}{\partial j} \right) )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial r} )</td>
<td>( \frac{\partial A_i(O,R)^T}{\partial r} \left( \frac{\partial C_i/O}{\partial \theta} \right) )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial \theta} )</td>
<td>( \frac{\partial A_i(O,R)^T}{\partial \theta} \left( \frac{\partial C_i/O}{\partial \theta} \right) )</td>
</tr>
</tbody>
</table>

61
### Basic Tail Rotor Derivatives in Forward Flight

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial \mu}{\partial \xi} )</td>
<td>( \frac{1}{\Omega R} )</td>
</tr>
<tr>
<td>( \frac{\partial \lambda^\prime}{\partial \xi} )</td>
<td>( \frac{1}{\Omega R} \left[ \frac{1}{2 \mu} \left( \frac{\partial C_1/\alpha}{\partial \mu} \frac{\partial C_1/\alpha}{\partial \mu} \right) \right] )</td>
</tr>
<tr>
<td>( \frac{\partial \lambda^\prime}{\partial \gamma} )</td>
<td>( \frac{1}{\Omega R} \left( \frac{\partial C_1/\alpha}{\partial \gamma} + \frac{\partial \lambda^\prime}{\partial \gamma} \right) )</td>
</tr>
<tr>
<td>( \frac{\partial \beta}{\partial \gamma} )</td>
<td>( \frac{1}{\Gamma} )</td>
</tr>
</tbody>
</table>

### Derivative of Tail Rotor Derivatives in Forward Flight

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial \gamma}{\partial \xi} )</td>
<td>( \frac{\partial \gamma}{\partial \xi} )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \eta} )</td>
<td>( \frac{\partial \gamma}{\partial \eta} )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \phi} )</td>
<td>( \frac{\partial \gamma}{\partial \phi} )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \theta} )</td>
<td>( \frac{\partial \gamma}{\partial \theta} )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \theta_\alpha} )</td>
<td>( \frac{\partial \gamma}{\partial \theta_\alpha} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial N}{\partial \xi} )</td>
<td>( \frac{\partial N}{\partial \xi} )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial \eta} )</td>
<td>( \frac{\partial N}{\partial \eta} )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial \phi} )</td>
<td>( \frac{\partial N}{\partial \phi} )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial \theta} )</td>
<td>( \frac{\partial N}{\partial \theta} )</td>
</tr>
<tr>
<td>( \frac{\partial N}{\partial \theta_\alpha} )</td>
<td>( \frac{\partial N}{\partial \theta_\alpha} )</td>
</tr>
</tbody>
</table>
**Nondimensional Horizontal Stabilizer Derivatives**

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial \gamma}{\partial \xi} )</td>
<td>( \frac{1}{V} )</td>
</tr>
<tr>
<td>( \frac{\partial e_{MH}}{\partial \xi} )</td>
<td>( \frac{r_H}{r_1} \frac{1}{4qA_M^2} \left[ \left( \frac{\partial Z}{\partial \xi} \right)_M - \frac{2Z_M}{V} \right] )</td>
</tr>
<tr>
<td>( \frac{\partial e_{SH}}{\partial \xi} )</td>
<td>( \frac{r_H}{r_1} \frac{1}{4qA_M} \left( \frac{\partial Z}{\partial \xi} \right)_M )</td>
</tr>
<tr>
<td>( \frac{\partial e_{IH}}{\partial \xi} )</td>
<td>( \frac{\partial e_{MH}}{\partial \xi} ) ( \frac{\partial e_{SH}}{\partial \xi} ) ( \frac{\partial \gamma}{\partial \xi} )</td>
</tr>
<tr>
<td>( \frac{\partial a_H}{\partial \xi} )</td>
<td>( \frac{\partial c_{MH}}{\partial \xi} ) ( \frac{\partial c_{SH}}{\partial \xi} )</td>
</tr>
<tr>
<td>( \frac{\partial a_H}{\partial \xi} )</td>
<td>( \frac{\partial c_{MH}}{\partial \xi} ) ( \frac{\partial e_{IH}}{\partial \xi} ) ( \frac{i_H}{V} )</td>
</tr>
</tbody>
</table>

**Horizontal Stabilizer Derivatives in Forward Flight**

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial X}{\partial \xi} )</td>
<td>( \frac{2}{V} X_H + \frac{q_H}{q} A_M e_H \left{ \left( \bar{a}<em>H - a</em>{10} \right) \left[ 1 - \frac{2a_H(1 + 6)}{nA R.} \right] + \left( a_H - i_H \right) \right} \frac{\partial a_H}{\partial x} )</td>
</tr>
<tr>
<td>( \frac{\partial X}{\partial \xi} )</td>
<td>( \frac{q_H}{q} A_M e_H \left{ \left( \bar{a}<em>H - a</em>{10} \right) \left[ 1 - \frac{2a_H(1 + 6)}{nA R.} \right] + \left( a_H - i_H \right) \right} \frac{\partial a_H}{\partial i} )</td>
</tr>
<tr>
<td>( \frac{\partial X}{\partial \xi} )</td>
<td>( \frac{\partial X}{\partial i} ) ( \frac{\partial a_H}{\partial \xi} ) ( \frac{\partial a_H}{\partial i} )</td>
</tr>
</tbody>
</table>
\[
\frac{\partial z}{\partial x} = \frac{2}{\nu} z_N - \left( \frac{f_H}{q} \right) q A_{\mu}^{\sigma} \left\{ 1 + \frac{\sigma_{\mu}(1 + \delta)}{n \Lambda R} \right\} \frac{\partial \sigma_{\mu}}{\partial x} \\
+ \left( \delta_{\mu} - \sigma_{\mu \alpha} \right) \left( \frac{\partial \sigma_{\mu}}{\partial x} \right) + \frac{C_{\sigma}}{\partial x} \frac{\partial \sigma_{\mu}}{\partial x}
\]

\[
\frac{\partial z}{\partial \xi} = \frac{f_H}{q} q A_{\mu}^{\sigma} \left\{ 1 + \frac{\sigma_{\mu}(1 + \delta)}{n \Lambda R} \right\} \frac{\partial \sigma_{\mu}}{\partial \xi} \\
+ \left( \delta_{\mu} - \sigma_{\mu \alpha} \right) \left( \frac{\partial \sigma_{\mu}}{\partial \xi} \right) + \frac{C_{\sigma}}{\partial \xi} \frac{\partial \sigma_{\mu}}{\partial \xi}
\]

\[
\frac{\partial z}{\partial \eta} = \frac{\partial z}{\partial \xi} \frac{\partial \sigma_{\mu}}{\partial \xi} + \frac{\partial z}{\partial \xi} \frac{\partial \sigma_{\mu}}{\partial \xi}
\]

\[
\frac{\partial M}{\partial \xi} = - \left( \frac{\partial z}{\partial \xi} \right) \frac{\partial z}{\partial \xi} + \frac{\partial z}{\partial \xi} \frac{\partial z}{\partial \xi}
\]

\[
\frac{\partial M}{\partial \eta} = - \left( \frac{\partial z}{\partial \eta} \right) \frac{\partial z}{\partial \eta} + \frac{\partial z}{\partial \eta} \frac{\partial z}{\partial \eta}
\]

\[
\frac{\partial M}{\partial \eta} = - \left( \frac{\partial z}{\partial \eta} \right) \frac{\partial z}{\partial \eta} + \frac{\partial z}{\partial \eta} \frac{\partial z}{\partial \eta}
\]

\[
\frac{\partial M}{\partial \xi} = - \left( \frac{\partial z}{\partial \xi} \right) \frac{\partial z}{\partial \xi} + \frac{\partial z}{\partial \xi} \frac{\partial z}{\partial \xi}
\]

\[
\frac{\partial M}{\partial \eta} = - \left( \frac{\partial z}{\partial \eta} \right) \frac{\partial z}{\partial \eta} + \frac{\partial z}{\partial \eta} \frac{\partial z}{\partial \eta}
\]

64
### Nondimensional Vertical Stabilizer Derivatives

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{\partial \theta}{\partial \eta}$</td>
<td>$\frac{1}{\dot{\eta}}$</td>
</tr>
<tr>
<td>$\frac{\partial M_t}{\partial \xi}$</td>
<td>$\frac{-1}{4\eta A_t} \left[ \left( \frac{\partial \gamma}{\partial \dot{\eta}} \right) + 2T_t \right]$</td>
</tr>
<tr>
<td>$\frac{\partial a_v}{\partial \xi}$</td>
<td>$\frac{\partial \eta_{\xi}}{\partial \dot{\xi}}$</td>
</tr>
<tr>
<td>$\frac{\partial M_j}{\partial \eta}$</td>
<td>$\frac{\partial \eta_{\eta}}{\partial \dot{\eta}}$</td>
</tr>
<tr>
<td>$\frac{\partial e_l}{\partial \eta}$</td>
<td>$\left( \frac{\partial \gamma}{\partial \dot{\eta}} \right) + 2T_t$</td>
</tr>
<tr>
<td>$\frac{\partial a_v}{\partial \eta}$</td>
<td>$\frac{\partial \eta_{\eta}}{\partial \dot{\eta}}$</td>
</tr>
<tr>
<td>$\frac{\partial}{\partial \eta} \left( \frac{\partial \beta}{\partial \dot{\eta}} + \frac{\partial \eta_{\xi}}{\partial \dot{\eta}} + \frac{\partial \eta_{\eta}}{\partial \dot{\eta}} \right)$</td>
<td>$\left( \frac{\partial \gamma}{\partial \dot{\eta}} \right) + 2T_t$</td>
</tr>
</tbody>
</table>

### Vertical Stabilizer Derivatives in Forward Flight

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\left( \frac{\partial x}{\partial \dot{x}} \right)_t$</td>
<td>$\frac{2}{\dot{x}} \left[ \frac{v}{\dot{x}} + 2 \Delta \tilde{B}_{\infty} \right]$</td>
</tr>
<tr>
<td>$\left( \frac{\partial x}{\partial \dot{y}} \right)_t$</td>
<td>$\frac{q}{\eta} q A_\iota \sigma_i \left{ \left( \alpha_i - \alpha_{i0} \right) \left( 1 - \frac{2a_v(1 + \delta_i)}{\pi A_R} \right) + \alpha_i \right} \frac{\partial \eta_{\xi}}{\partial \dot{y}}$</td>
</tr>
<tr>
<td>$\left( \frac{\partial x}{\partial \dot{y}} \right)_{\infty}$</td>
<td>$\frac{1}{\dot{y}} \left( \frac{\partial \gamma}{\partial \dot{y}} \right) \left( \frac{2}{\dot{x}} \right) \left( \frac{\partial \gamma}{\partial \dot{y}} \right) \left( \frac{1}{\dot{y}} \right)$</td>
</tr>
<tr>
<td>$\left( \frac{\partial y}{\partial \dot{x}} \right)_t$</td>
<td>$\frac{2}{\dot{x}} \left( \frac{v}{\dot{x}} \right) q A_\iota \sigma_i \frac{\partial \eta_{\xi}}{\partial \dot{x}}$</td>
</tr>
<tr>
<td>$\left( \frac{\partial y}{\partial \dot{y}} \right)_t$</td>
<td>$\frac{1}{\dot{y}} \left( \frac{\partial \gamma}{\partial \dot{y}} \right) \left{ \left( \frac{v}{\dot{y}} q A_\iota \eta_{\eta} \frac{\partial \eta_{\xi}}{\partial \dot{y}} \Delta \tilde{B}_{\infty} \right) \left[ \left( \frac{v}{\dot{y}} \right) + \frac{1}{\dot{y}} \left( \frac{\partial \gamma}{\partial \dot{y}} \right) \right] \right}$</td>
</tr>
</tbody>
</table>

65
<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \frac{\partial \gamma}{\partial \rho} )</td>
<td>( \frac{\partial \gamma}{\partial \gamma} ) ( h_v )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \tau} )</td>
<td>( -\frac{\partial \gamma}{\partial \tau} ) ( l_v )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \zeta} )</td>
<td>( \frac{\partial \gamma}{\partial \zeta} ) ( h_v )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \psi} )</td>
<td>( \frac{\partial \gamma}{\partial \psi} ) ( h_v )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \xi} )</td>
<td>( \frac{\partial \gamma}{\partial \xi} ) ( h_v )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \eta} )</td>
<td>( -\frac{\partial \gamma}{\partial \rho} ) ( l_v )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \xi} )</td>
<td>( -\frac{\partial \gamma}{\partial \tau} ) ( l_v )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \zeta} )</td>
<td>( -\frac{\partial \gamma}{\partial \psi} ) ( l_v )</td>
</tr>
<tr>
<td>( \frac{\partial \gamma}{\partial \xi} )</td>
<td>( -\frac{\partial \gamma}{\partial \xi} ) ( l_v )</td>
</tr>
</tbody>
</table>

66
### Nondimensional Fuselage Derivatives

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{\partial y}{\partial i}$</td>
<td>$-\frac{1}{V}$</td>
</tr>
<tr>
<td>$\frac{\partial \beta}{\partial y}$</td>
<td>$\frac{1}{V}$</td>
</tr>
<tr>
<td>$\frac{\partial e_{M_F}}{\partial i}$</td>
<td>$r_i \frac{1}{4} \left[ \frac{\partial Z}{\partial i} \left</td>
</tr>
<tr>
<td>$\frac{\partial e_{M_F}}{\partial \beta}$</td>
<td>$r_i \frac{1}{4} \frac{\partial \gamma}{\partial i}$</td>
</tr>
<tr>
<td>$\frac{\partial \alpha_F}{\partial i}$</td>
<td>$\frac{\partial e_{M_F}}{\partial i}$</td>
</tr>
<tr>
<td>$\frac{\partial \alpha_F}{\partial \beta}$</td>
<td>$\left( \frac{\partial e_{M_F}}{\partial \beta} \cdot \frac{\partial \gamma}{\partial i} \right)$</td>
</tr>
<tr>
<td>$\frac{\partial \gamma}{\partial \alpha_F}$</td>
<td>(From curves)</td>
</tr>
<tr>
<td>$\frac{\partial M}{\partial \alpha_F}$</td>
<td>$\left( \frac{\partial M}{\partial \beta} \right)_i$</td>
</tr>
<tr>
<td>$\frac{\partial N}{\partial \alpha_F}$</td>
<td>$\left( \frac{\partial N}{\partial \beta} \right)_i$</td>
</tr>
<tr>
<td>$\frac{\partial \gamma}{\partial \beta}$</td>
<td>(From curves)</td>
</tr>
</tbody>
</table>

### Fuselage Derivatives in Forward Flight

<table>
<thead>
<tr>
<th>Derivative</th>
<th>Equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\frac{\partial X}{\partial i}$</td>
<td>$\frac{2}{V} X_i$</td>
</tr>
<tr>
<td>$\frac{\partial Y}{\partial i}$</td>
<td>$\frac{2}{V} Y_i$</td>
</tr>
<tr>
<td>$\frac{\partial Z}{\partial i}$</td>
<td>$\frac{2}{V} Z_i$</td>
</tr>
<tr>
<td>$\frac{\partial L}{\partial i}$</td>
<td>$-\frac{D_F - \frac{\partial L}{\partial \alpha_F}}{\frac{\partial \gamma}{\partial \beta}}$</td>
</tr>
<tr>
<td>$\frac{\partial M}{\partial i}$</td>
<td>$\frac{2}{M_F + \frac{\partial M}{\partial \alpha_F}} \frac{\partial M}{\partial \alpha_F}$</td>
</tr>
<tr>
<td>$\frac{\partial N}{\partial i}$</td>
<td>$\frac{2}{N_F + \frac{\partial N}{\partial \alpha_F}} \frac{\partial N}{\partial \alpha_F}$</td>
</tr>
<tr>
<td>$\frac{\partial R}{\partial i}$</td>
<td>$\frac{q}{\frac{\partial \beta}{\partial \beta}}$</td>
</tr>
<tr>
<td>$\frac{\partial \gamma}{\partial \beta}$</td>
<td>(From curves)</td>
</tr>
</tbody>
</table>
APPENDIX D. VARIABLE LIST.

Variable list for JANRAD Stability Routine

A - rotor disk area
Ab - area of blades
Abt - area of tail rotor blades
Ap - projected area of fuselage under rotor
A1 - lateral cyclic pitch
als - longitudinal flapping
a - main rotor blade lift curve slope
Ah - area of the horizontal stab
ah - lift curve slope of the horizontal stab
alph - angle of attack of horizontal stab
alphas - angle of attack WRT shaft axis
alphoh - alpha-zero-lift of the horizontal stab (at the trim condition)
alplov - alpha-zero-lift of the vertical stab
alplow - alpha-zero-lift of the wing
alpv - angle of attack of vertical stab
alpw - angle of attack of wing
altpp - mean alpha TPP
ao - main rotor coning angle
aot - tail rotor coning angle
At - disc area of the tail rotor
at - tail rotor blade lift curve slope
at - tail rotor blade lift curve slope
Av - area of the vertical stab
av - lift curve slope of the vertical stab
Aw - area of the wing
aw - lift curve slope of wing
B1 - longitudinal cyclic pitch
b1s - lateral flapping
beta - trim sideslip angle
bh - span of the horizontal stab
bv - semispan of the vertical stab
bw - span of the wing
c - main rotor chord
cdoh - Cdo of the horizontal stab
cdow - Cdo of the wing
ch - coefficient of ????
chsig - Ch/sigma
clvertmax - maximum cl of the vertical stab with full rudder defl
cmu - circulation coefficient for NOTAR™
cmun - design Cmu for NOTAR™ boom in hover (.3-.6)
cot - tail rotor chord
cpipitch - Control power/inch pitch

68
cpiroll - Control power/inch pitch
cpiyaw - Control power/inch pitch
cppitch - Control power pitch
cproll - Control power roll
cpyaw - Control power yaw
cT - coefficient of thrust
ctsig - Ct/sigma
ctsigt - Ct/sigma of the tail rotor
cTr - wing root chord
cTtw - wing tip chord
cq - coefficient of torque
cqsig - Cq/sigma
deldv - biplane effect (mutual interference of tail rotor and vertical stab)
delih - span efficiency factor of the horizontal stab
delifv - span efficiency factor of the vertical stab
deliw - span efficiency factor of the wing
delta3 - tail rotor delta-3 angle
delvmax - maximum rudder deflection on the vertical stab
desdmdq - designed pitch damping (dM/dq)
desndrd - designed yaw damping (dN/dr)
desdrdp - designed roll damping (dR/dp)
detafdalph - fuselage downwash ratio for the horizontal stab
detafdalphw - fuselage downwash ratio for the wing
detafdbeta - fuselage sideslip ratio (= despinkalpha)
diAN - NOTAR™ boom diameter
e - blade offset
GW - gross weight
g - acceleration due to gravity
hh - horizontal stab vertical offset
hhd - height from waterline to horizontal tail
hm - main rotor vertical offset
hmd - height from waterline to main rotor hub
hslot - total of slot heights for NOTAR™ boom
ht - tail rotor vertical offset
htd - height from waterline to tail rotor hub
htn - NOTAR™ thruster vertical offset
hv - vertical stab vertical offset
hvd - height from waterline to vertical fin
hw - wing vertical offset
hwd - height from waterline to wing
htnd - height from waterline to NOTAR™
Ib - blade flapping inertia
Ibt - tail rotor blade flapping moment of inertia
Ixx - mass moment of inertia about x axis
Iyy - mass moment of inertia about y axis
Ixz - mass moment of inertia about xz plane
Izz - mass moment of inertia about z axis
Ic - Ixx*Izz
i - shaft incidence main rotor
ih - angle of incidence of horizontal stab
iw - angle of incidence of wing
lamp - lambda' - inflow ratio WRT TPP
lh - horizontal stab longitudinal offset
lhd - fuselage station of horizontal tail
lm - main rotor longitudinal offset
lmd - fuselage station of main rotor hub
lt - tail rotor longitudinal offset
ltl - fuselage station of tail rotor hub
ltn - NOTAR™ thruster longitudinal offset
ltnl - fuselage station of NOTAR™ boom
ltnl - fuselage station of NOTAR™ thruster
lv - vertical stab longitudinal offset
lvd - fuselage station of vertical tail
iw - wing longitudinal offset
lwd - fuselage station of wing
lockno - main rotor lock number
locknot - tail rotor lock number
lt - tail rotor longitudinal offset
lv - vertical stab longitudinal offset
maxr - maximum rudder deflection
mu - advance ratio
mut - tail rotor advance ratio
n - a counter in CTPLOTS.M
nt - number of tail rotor blades
ohm - omega
ohmt - tail rotor omega
phin - NOTAR™ thruster sleeve rotation angle
pho - roll trim attitude (euler angle)
prpitch - Pilot rating (dM/dq)/lyy
prroll - Pilot rating (dR/dp)/lxx
pryaw - Pilot rating (dN/dr)/lzz
q - dynamic pressure, 1/2 rho V^2
qhq - horizontal tail dynamic press ratio (qh/q)
qvin - dynamic press due to downwash at the NOTAR™ slots
qvq - vertical tail dynamic press ratio (qv/q)
R - blade radius
Rt - tail rotor blade radius
rho - air density
sidearm - maximum pedal travel or twist grip deflection
sigma - solidity
sigmat - tail rotor solidity
swirl - mean swirl angle off main rotor measured at the NOTAR™ slots
T - thrust
Tt - thrust of the tail rotor
thetal - main rotor blade twist
thetal1 - tail rotor blade twist
thetao - blade pitch at root
thetat - induced angle at the tip
tho - trim pitch attitude (euler angle)
V - forward velocity
v1 - induced flow velocity at a hover
vlf - approximation for v1 at forward airspeed
vfvl - rotor downwash ratio for the fuselage
vhl - rotor downwash ratio for horizontal stab
vwl - rotor downwash ratio for wing
uo - x initial velocity
wo - y initial velocity
wo - z initial velocity
xcg - height from waterline to cg
xcouple - designed cross coupling
Ytmaxn - maximum Y force from NOTAR™ thruster
ycg - length from buttline to cg
Yv - side-force of vertical stab
yhd - length from buttline to horizontal tail
ym - main rotor lateral offset
ymd - length from buttline to main rotor hub
ytd - length from buttline to tail rotor hub
ytnd - length from buttline to NOTAR™
ywd - length from buttline to wing
yvd - length from buttline to vertical tail
zcg - fuselage station of cg
Zh - vertical force of horizontal stab
Zw - vertical force of wing

CRUISE

CRUISE -- BASIC MR Derivatives

dalda - dals/dAl
daldb - dals/dBl
daldlamp - dals/d(lambda')
daldmu - dals/dmu
daldp - dals/dp
daldq - dals/dq
daldtheto - dals/dtheta(o)
dbetdydot - d(Beta)/dYdot
dblda - dbls/dAl
dbldb - dbls/dBl
dbdlamp - dbls/d(lambda')
dbdmu - dbls/dmu
dbdp - dbls/dp
dbdq - dbls/dq
dbdtheto - dbls/dtheta(o)
dchsigda - d(Ch/sigma)/dals
dchsigdb - d(Ch/sigma)/d(Bl/s)
dchsigdlamp - d(Ch/sigma)/d(lambda')
dchsigdmu - d(Ch/sigma)/dmu
dcgisigdlamp - d(Cg/sigma)/d(lambda')
dctsigdmu - d(Ct/sigma)/dmu
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')
dcgisigdlamp - d(Cq/sigma)/d(lambda')

CRUISE -- MR derivatives

dmdalm - dM/dAl
dmdblm - dM/dBl
dmdpm - dM/dp
dmdqm - dM/dq
dmdthetom - dM/dtheta(0)
dmdxdotm - dM/dXdot
dmdydotm - dM/dYdot
dmdzdotm - dM/dZdot
dnalm - dN/dr
dndthetom - dN/dtheta(0)
dndxdotm - dN/dXdot
dndzdotm - dN/dZdot
drdalm - dR/dAl
drdblm - dR/dBl
drdpm - dR/dp
drdqm - dR/dq
drdthetom - dR/dtheta(0)
drdxdotm - dR/dXdot
drdydotm - dR/dYdot
drdzdotm - dR/dZdot
dxalm - dX/dAl
dxdblm - dX/dBl
dxdpm - dX/dp

72
dydalm - dY/dA1
dydblm - dY/db1
dydpm - dY/dp
dydqm - dY/dq
dydthetom - dY/dtheta(0)
dydxdotm - dY/dXdot
dydydotm - dY/dYdot
dydzdotm - dY/dZdot
dzdblm - dZ/db1

dzdrm - dZ/dr
dzdthetom - dZ/dtheta(0)
dzdxdotm - dZ/dXdot
dzdzdotm - dZ/dZdot

CRUISE -- TAIL ROTOR Derivatives

dctsigdalampt - d(Ct/sigma)/d(lambda')
dctsigdmut - d(Ct/sigma)/d(mu)
dctsigdthetot - d(Ct/sigma)/d(theta0)
dlampdydott - d(lambda'/sigma)/dYdot
dydxdott - dY/dXdot
dydydott - dY/dYdot
dydpt - dY/dp
dydr - dY/dr
dydydot - dY/d(theta0)

drdxdot - dR/dXdot
drdydot - dR/dYdot

drdpt - dR/dp

drdr - dR/dr

drdthetot - dR/dtheta(0)
dndxdott - dN/dXdot
dndydott - dN/dYdot

CRUISE -- V; AL STAB DERIVATIVES

dalpvdxdotv - d(alpha vert)/dXdot
dalpvdvydotv - d(alpha vert)/dYdot
dbetadydotv - d(beta)/dYdot
detavxdotv - d(eta tail rotor)/dXdot
detadvxdotv - d(eta tail rotor)/dYdot
detafdydotv - d(eta fuselage)/dYdot
drdxdotv - dR/dXdot
drdydotv - dR/dYdot
dxxdotv - dX/dXdot
dxdydotv - dX/dYdot
dyxdotv - dY/dXdot
dydydotv - dY/dYdot
dypv - dY/dp
\[ dydrv - dY/dr \]

**CRUISE -- HORIZONTAL STAB DERIVATIVES**

\[ \text{dalphdxdoth} - \frac{d(\alpha \text{ horiz})}{dXdot} \]
\[ \text{dalphdzzbldoth} - \frac{d(\alpha \text{ horiz})}{dZ \text{ double dot}} \]
\[ \text{dalphhdzdoth} - \frac{d(\alpha \text{ horiz})}{dZdot} \]
\[ \text{detamhdxdoth} - \frac{d(\text{eta main rotor})}{dXdot} \]
\[ \text{deltafxdzoth} - \frac{d(\text{eta fuse})}{dZdot} \]
\[ \text{detamhdxdoth} - \frac{d(\text{eta main rotor})}{dZdot} \]
\[ \text{dgamdzdoth} - \frac{d(\text{gamma})}{dZdot} \]
\[ \text{dxdxdoth} - \frac{dX}{dXdot} \]
\[ \text{dxdzbldoth} - \frac{dX}{dZ \text{ double dot}} \]
\[ \text{dxdzdoth} - \frac{dX}{dZdot} \]
\[ \text{dzdxdoth} - \frac{dZ}{dXdot} \]
\[ \text{dzzdoth} - \frac{dZ}{dZ \text{ double dot}} \]
\[ \text{dzzdoth} - \frac{dZ}{dZdot} \]

**CRUISE -- FUSELAGE**

The following are from the curves in Appendix A of Ref. 2:

\[ \text{dfdalpf} - \frac{d(D/q)}{d(\alpha \text{ fuse})} \]
\[ \text{dlqdalpf} - \frac{d(L/q)}{d(\alpha \text{ fuse})} \]
\[ \text{dmdlalpf} - \frac{d(M/q)}{d(\alpha \text{ fuse})} \]
\[ \text{dnqdalpf} - \frac{d(N/q)}{d(\beta)} \]
\[ \text{drqdalpf} - \frac{d(R/q)}{d(\beta)} \]
\[ \text{dsfqlralpf} - \frac{d(Y/q)}{d(\beta)} \]
\[ \text{dalpfdxdotf} - \frac{d(\alpha \text{ fuse})}{dXdot} \]
\[ \text{dalpfdzdotf} - \frac{d(\alpha \text{ fuse})}{dZdot} \]
\[ \text{dbetadydotf} - \frac{d(\beta)}{dYdot} \]
\[ \text{detamfdxdotf} - \frac{d(\text{eta})M}{dXdot} \]
\[ \text{detamfdzdotf} - \frac{d(\text{eta})M}{dZdot} \]
\[ \text{dgamdzdotf} - \frac{d(\text{gamma})}{dZdot} \]
\[ \text{dmdxdotf} - \frac{dM}{dXdot} \]
\[ \text{dmdzdotf} - \frac{dM}{dZdot} \]
\[ \text{dndydotf} - \frac{dN}{dYdot} \]
\[ \text{drdydotf} - \frac{dR}{dYdot} \]
\[ \text{dxdxdotf} - \frac{dX}{dXdot} \]
\[ \text{dxdzdotf} - \frac{dX}{dZdot} \]
\[ \text{dydydotf} - \frac{dY}{dYdot} \]
\[ \text{dzdxdotf} - \frac{dZ}{dXdot} \]
\[ \text{dzdxdotf} - \frac{dZ}{dZdot} \]

**HOVER -- BASIC MR Derivatives**

\[ \text{dalda} - \frac{d\alpha s}{dA} \]

74
daldb - dals/dB1
daldlamp - dals/d(lambda')
daldmu - dals/dmu
daldp - dals/dp
daldq - dals/dq
daldtheto - dals/dtheta(o)
dblda - dbls/dA1
dbldb - dbls/dB1
dbldlamp - dbls/d(lambda')
dbldmu - dbls/dmu
dbldp - dbls/dp
dbldq - dbls/dq
dblldtheto - dbls/dtheta(o)
dbeta dydot - d(Beta)/dYdot
dchsigda - d(Ch/sigma)/dals
dchsigdb - d(Ch/sigma)/dbls
dchsigdlamp - d(Ch/sigma)/d(lambda')
dchsigdmu - d(Ch/sigma)/dmu
dchsigdtheto - d(Ch/sigma)/dtheta(o)
dctsigdlamp - d(Ct/sigma)/d(lambda')
dctsigdmu - d(Ct/sigma)/dmu
dctsigdtheto - d(Ct/sigma)/dtheta(o)
dcsigdlamp - d(Cq/sigma)/d(lambda')
dcsigdmu - d(Cq/sigma)/dmu
dcsigdtheto - d(Cq/sigma)/dtheta(o)
dcysigdb - d(Cy/sigma)/dbls
dlampdxdot - d(lambda')/dXdot
dlampdydot - d(lambda')/dYdot
dlampdzdot - d(lambda')/dZdot
dmdalsm - dM/dals - main rotor stiffness
dmudxdot - dmudxdot
drdb1sm - dR/dbls - main rotor stiffness

RIGGING -- Cockpit stick rigging gains

dalmdela - dals/d(delta aileron)
dblmddele - dbls/d(delta elev)
ddelvddelp - rudder deflection per pedal travel
dphinddelp - phi(notar thruster sleeve angle)/d(pedal)
dthetodc - dtheta(0.7)/d(collective)
dthetomddelc - dthetaOm/d(collective)
dthetotddelp - dtheta(0.7,tail rotor)/d(pedal travel)
APPENDIX E. PLOT LISTING.

Plots are saved under the following filenames:

Longitudinal roots - rootlon.met
Lateral/Directional roots - rootlat.met
Coupled roots - rootcoup.met

Longitudinal Cyclic
Longitudinal Cyclic to U, Hover - cbe2uh.met
Longitudinal Cyclic to Theta, Hover - cbe2theh.met
Longitudinal Cyclic to Pitch Rate, Hover - cbe2qh.met
Longitudinal Cyclic to W, Hover - cbe2wh.met

Collective
Collective to U, Hover - cbc2uh.met
Collective to Pitch, Hover - cbc2theh.met
Collective to Pitch Rate, Hover - cbc2qh.met
Collective to W, Hover - cbc2wh.met

Lateral cyclic
Lateral Cyclic to Bank, Hover - cba2phih.met
Lateral Cyclic to Sideslip (v), Hover - cba2vh.met
Lateral Cyclic to Roll Rate, Hover - cba2ph.met
Lateral Cyclic to Yaw Rate, Hover - cba2rh.met
Lateral Cyclic to Yaw, Hover - cba2yh.met

Pedals
Pedals to Bank, Hover - cbp2phih.met
Pedals to Sideslip (v), Hover - cbp2vh.met
Pedals to Roll Rate, Hover - cbp2ph.met
Pedals to Yaw Rate, Hover - cbp2rh.met
Pedals to Yaw, Hover - cbp2yh.met
Sample output file for the Prouty helicopter at a hover.

*** RESULTS ***
prouth

*** INPUT DATA ***

Flight Conditions

- Forward velocity = 0 kts
- Temperature = 90 degs F
- Pressure altitude = 0 ft
- Auxiliary thrust = 0 lbs

Fuselage

- Gross weight = 20000 lbs
- Equivalent flat plate area = 19.3 ft^2
- Vertical projected area = 380.0 ft^2
- CG height above waterline = 0.0 ft
- CG fuselage station = 0.0 ft
- CG position rt of buttline = 0.0 ft
- \( I_{xx} = 35000.0 \text{ slug ft}^2 \)
- \( I_{yy} = 40000.0 \text{ slug ft}^2 \)
- \( I_{zz} = 35000.0 \text{ slug ft}^2 \)
- \( I_{xz} = 0.0 \text{ slug ft}^2 \)
- Downwash ratio = 1.50

Main Rotor

- Number of blades = 4
- Rotor radius = 30.0 ft
- Blade chord = 2.0 ft
- Blade twist = 10.00 degs
- Blade airfoil = HH-02
- Blade lift curve slope = 5.73
- Blade weight = 207.0 lbs
- Rotational velocity = 21.67 rad/s
- Blade grip length = 4.5 ft
- Hinge offset = 1.5 ft
- Flapping moment of inertia = 2870.0 slug ft^2
- Hub height above waterline = 7.5 ft
- Hub fuselage station = 0.0 ft
- Hub position rt of buttline = 0.0 ft
- Mast incidence = 0.00 deg

Tail rotor (zero if NOTAR)

- Number of blades = 3.0
- Blade chord = 1.0 ft
- Blade radius = 6.5 ft
- Lift curve slope = 5.73
- Rotational velocity = 100.00 rad/sec
- Flapping moment of inertia = 772 slug ft^2
Delta-3 angle = -30.00 deg
Blade twist = -5.00 deg
Hub height above waterline = 6.0 ft
Hub fuselage station = 37.0 ft
Hub position rt of buttline = 0.0 ft

Wing

Area = 0.0 ft^2
Span = 0.0 ft
CL = 0.00
CDo = 0.0000
Tip cord = 0.0 ft
Root cord = 0.0 ft
Wing efficiency factor = 0.00
Zero lift angle = 0.00 deg
Angle of incidence = 0.00 deg
Lift curve slope = 0.00
Height above waterline = 0.0 ft
Fuselage station = 0.0 ft
Position right of buttline = 0.0 ft
Rotor downwash ratio = 0.00
Fuselage downwash ratio = 0.00

Horizontal tail

Area = 18.0 ft^2
Span = 9.0 ft
CL = 0.20
CDo = 0.0045
Zero lift angle = 0.00 deg
Angle of incidence = 0.00 deg
Lift curve slope = 5.73
Height above waterline = -1.5 ft
Fuselage station = 33.0 ft
Position right of buttline = 0.0 ft
Dynamic pressure ratio = 0.60
Rotor downwash ratio = 1.50
Fuselage downwash ratio = 1.20

Vertical tail

Area = 33.0 ft^2
Span = 7.7 ft
CL = 0.20
CDo = 0.0045
Height above waterline = 3.0 ft
Fuselage station = 35.0 ft
Position right of buttline = 0.0 ft
Zero lift angle = 0.00 deg
Maximum Cl = 2.00
Dynamic pressure ratio = 0.60
Lift curve slope = 5.70

Rigging

Long cyclic pitch/inch defl = 3.33 deg/in
Lat cyclic pitch/inch defl = 2.05 deg/in
Collective pitch/inch defl = 1.33 deg/in
Tail rotor pitch change/defl = -8.70 deg/unit
Max deflection of control from neutral for NOTAR = 0.09 units
Displacement of anti-torque control until full rudder = 0.00 units

*** CALCULATED DATA ***

Main Rotor

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advance ratio</td>
<td>0.0</td>
</tr>
<tr>
<td>Inflow parameter wrt TPP</td>
<td>0.000</td>
</tr>
<tr>
<td>Tip path angle</td>
<td>0.0 degs</td>
</tr>
<tr>
<td>Rotor coning angle</td>
<td>7.4 degs</td>
</tr>
<tr>
<td>1st lat cyclic term-A1</td>
<td>0.00 degs</td>
</tr>
<tr>
<td>1st long cyclic term-B1</td>
<td>-0.00 degs</td>
</tr>
<tr>
<td>Lateral flapping</td>
<td>0.0 degs</td>
</tr>
<tr>
<td>Longitudinal flapping</td>
<td>0.0 degs</td>
</tr>
<tr>
<td>Lock number</td>
<td>7.2</td>
</tr>
</tbody>
</table>

Tail Rotor (zero if NOTAR)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tail rotor thrust</td>
<td>1349.5 lbs</td>
</tr>
<tr>
<td>Advance ratio</td>
<td>0.0</td>
</tr>
<tr>
<td>Inflow parameter</td>
<td>-0.073</td>
</tr>
<tr>
<td>Rotor coning angle</td>
<td>-0.1 degs</td>
</tr>
<tr>
<td>Lateral flapping</td>
<td>0.0 degs</td>
</tr>
<tr>
<td>Longitudinal flapping</td>
<td>0.00 degs</td>
</tr>
<tr>
<td>Lock number</td>
<td>2.7</td>
</tr>
</tbody>
</table>

State Matrices

Longitudinal uncoupled plant (A or F depending on notation)
States are [u w q theta]

\[
\begin{bmatrix}
-0.0106 & 0 & 2.5968 & -32.2000 \\
0 & -0.2698 & 0 & 0.0002 \\
0.0038 & 0 & -0.8878 & 0 \\
0 & 0 & 0.9977 & 0
\end{bmatrix}
\]

Longitudinal uncoupled input matrix (B or G depending on notation)
Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]

\[
\begin{bmatrix}
1.2248 & -0.0000 & -0.0738 & 0 \\
0 & -5.8773 & 0 & 0 \\
-0.4338 & 0.0000 & 0.0261 & -0.0238 \\
0 & 0 & 0 & 0
\end{bmatrix}
\]

Lateral/directional uncoupled plant (A or F depending on notation)
States are [v p phi r psi]

\[
\begin{bmatrix}
-0.0305 & -2.6260 & 32.1267 & 0.7353 & 0 \\
-0.0064 & -1.0274 & 0 & 0.0783 & 0 \\
0 & 1.0000 & 0 & -0.0000 & 0 \\
0.0130 & 0.0783 & 0 & -3.3360 & 0 \\
0 & 0 & 0 & 1.0000 & 0
\end{bmatrix}
\]
Lateral/directional uncoupled input matrix (B or G depending on notation)

Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]

<table>
<thead>
<tr>
<th></th>
<th>0.1200</th>
<th>0.0030</th>
<th>0.7533</th>
<th>-1.9736</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inputs</td>
<td>0.0486</td>
<td>0.0000</td>
<td>0.3049</td>
<td>-0.2101</td>
</tr>
<tr>
<td></td>
<td>0.3001</td>
<td>0.0000</td>
<td>1.2958</td>
<td>0.0000</td>
</tr>
</tbody>
</table>

Coupled plant (A or F depending on notation)

States are [u w q theta v p phi r psi]

Columns 1 through 7

<table>
<thead>
<tr>
<th></th>
<th>-0.0106</th>
<th>0.2506</th>
<th>-32.2000</th>
<th>-0.0057</th>
<th>-0.7469</th>
<th>0.0002</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>-0.2698</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0.0038</td>
<td>0</td>
<td>-0.8878</td>
<td>0</td>
<td>0.0013</td>
<td>0.2645</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0.9977</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0.0057</td>
<td>0</td>
<td>-0.7469</td>
<td>-0.0000</td>
<td>-0.0305</td>
<td>2.6260</td>
</tr>
<tr>
<td></td>
<td>0.0023</td>
<td>0</td>
<td>-0.3023</td>
<td>0</td>
<td>-0.0064</td>
<td>-1.0274</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>-0.0198</td>
<td>0</td>
<td>0</td>
<td>0.0130</td>
<td>0.0783</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Columns 8 through 9

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.0266</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0.0674</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0.7353</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0.0783</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>-0.0000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>-3.3360</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1.0000</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Coupled input matrix (B or G depending on notation)

Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]

<table>
<thead>
<tr>
<th></th>
<th>1.2248</th>
<th>-0.0000</th>
<th>-0.0738</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>-5.8773</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>-0.4338</td>
<td>0.0000</td>
<td>0.0261</td>
<td>-0.0238</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0.1200</td>
<td>0.0000</td>
<td>0.7533</td>
<td>-1.9736</td>
</tr>
<tr>
<td></td>
<td>0.0486</td>
<td>0.0000</td>
<td>0.3049</td>
<td>-0.2101</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0.3001</td>
<td>0</td>
<td>1.2958</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Eigenvalue

Uncoupled

Longitudinal plant

<table>
<thead>
<tr>
<th>Root</th>
<th>wn</th>
<th>damping</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1.0155</td>
<td>1.0155</td>
<td>1.0000</td>
</tr>
<tr>
<td>0.0585 + 0.3398i</td>
<td>0.3448</td>
<td>-0.1698</td>
</tr>
<tr>
<td>0.0585 - 0.3398i</td>
<td>0.3448</td>
<td>-0.1698</td>
</tr>
<tr>
<td>-0.2698</td>
<td>0.2698</td>
<td>1.0000</td>
</tr>
</tbody>
</table>
### Lateral/Directional plant

<table>
<thead>
<tr>
<th>Root</th>
<th>wn</th>
<th>damping</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-1.0000</td>
</tr>
<tr>
<td>-3.3433</td>
<td>3.3433</td>
<td>1.0000</td>
</tr>
<tr>
<td>-1.1784</td>
<td>1.1784</td>
<td>1.0000</td>
</tr>
<tr>
<td>0.0639 + 0.4025i</td>
<td>0.4075</td>
<td>-0.1568</td>
</tr>
<tr>
<td>0.0639 - 0.4025i</td>
<td>0.4075</td>
<td>-0.1568</td>
</tr>
</tbody>
</table>

### Coupled Plant

<table>
<thead>
<tr>
<th>Root</th>
<th>wn</th>
<th>damping</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-1.0000</td>
</tr>
<tr>
<td>-3.3432</td>
<td>3.3432</td>
<td>1.0000</td>
</tr>
<tr>
<td>-1.0970 + 0.2583i</td>
<td>1.1270</td>
<td>0.9734</td>
</tr>
<tr>
<td>-1.0970 - 0.2583i</td>
<td>1.1270</td>
<td>0.9734</td>
</tr>
<tr>
<td>0.0760 + 0.3983i</td>
<td>0.4054</td>
<td>-0.1873</td>
</tr>
<tr>
<td>0.0760 - 0.3983i</td>
<td>0.4054</td>
<td>-0.1873</td>
</tr>
<tr>
<td>0.0461 + 0.3582i</td>
<td>0.3611</td>
<td>-0.1276</td>
</tr>
<tr>
<td>0.0461 - 0.3582i</td>
<td>0.3611</td>
<td>-0.1276</td>
</tr>
<tr>
<td>-0.2689</td>
<td>0.2689</td>
<td>1.0000</td>
</tr>
</tbody>
</table>
APPENDIX G. SAMPLE OPEN LOOP PLOTS.

Sample open loop transfer plots

---

Open loop response Longitudinal Cyclic to $U$, Hover

Open loop response Longitudinal Cyclic to $\theta$, Hover
Open loop response Collective to Pitch Rate, Hover

Open loop response Collective to W, Hover
Open loop response Lateral Cyclic to Bank, Hover

Open loop response Lateral Cyclic to Sideslip ($\psi$), Hover

Angular Frequency (rad/sec)
Open loop response Latera Cyclic to Roll Rate, Hover

Open loop response Latera Cyclic to Yaw Rate, Hover
Open loop response Pedals to Sideslip (v), Hover

Open loop response Pedals to Roll Rate, Hover

Angular Frequency (rad/sec)
Open loop response Pedals to Yaw Rate, Hover

Angular Frequency (rad/sec)

Gain (dB)

Angular Frequency (rad/sec)
Open loop response Collective to U. Cruise

Open loop response Collective to Pitch, Cruise

Gain (dB)

Angular Frequency (rad/sec)
Open loop response Lateral Cyclic to Yaw, Cruise

Open loop response Pedals to Bank, Cruise
Open loop response Pedals to Yaw Rate, Cruise

Gain (dB)

Angular Frequency (rad/sec)

Open loop response Pedals to Yaw, Cruise

Gain (dB)

Angular Frequency (rad/sec)
APPENDIX H. COMPUTER CODE.

Computer code for JANRAD Stability

A. STAB.M

% STAB.M
% NPS Helo Preliminary Design Program
% Stability and Control Routines
% Written by MAJ Walter M. Wirth, Jr.
% September 1993

% This program was designed as an interactive preliminary
% design tool for stability and control analysis of a single
% main rotor conventional or compound helicopter. The
% program provides stability derivatives, roots of the aircraft
% plant, plots of various control parameters as well as open loop
% control bandwidths from control inputs to aircraft response.

clear
load temp
eval(['load ', filename]);
clc
disp('')
disp('')
disp('')
*** STABILITY AND CONTROL ROUTINE ***
disp('')
disp('')
disp('')
disp('')
pause(3)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% *** If editing an existing file: get file name, display edit
% menu, allow changes to selected variables, and save under
% desired file name. Loads to and saves from current
% directory as a .mat file. ***
% if exist('Ib');
% answer0=1;
% else
% answer0=2;
% end
% if answer0==1,
% check2=1;
% while check2 > 0
% clc
disp('')
disp('')
*** STABILITY AND CONTROL MENU ***
disp('')
*** ADDITIONAL PARAMETERS (1 of 3) ***
disp('')
disp(' Main Rotor')
disp(' 1. flapping mom of inertia 2. hub height above waterline')
disp(' 3. hub fuselage station 4. hub posn right of buttline')
disp(' 5. mast incidence')
% disp('')
disp(' Tail Rotor (enter zeros (0) if using NOTAR)')
disp(' 6. height above waterline 7. hub fuselage station')
disp(' 8. posn right of buttline 9. number of blades')

100
disp('10. blade chord 11. blade radius')
disp('12. lift curve slope 13. rotational velocity')
disp('14. flap mom of inertia 15. delta-3 angle')
disp('16. blade twist')

disp('Verticle Fin')
disp('17. height above waterline 18. fuselage station')
disp('19. posn right of buttline 20. alpha zero lift')
disp('21. CL max 22. dynamic pressure ratio')

disp('0. NO CHANGES')
choice=input('Input the parameter to change: ');
if choice==1,
    disp('')
    Ib=templ;
end
templ=Ib;
Ib=input('Blade flapping moment of inertia (slug ft^2): ');
if isempty(Ib),
    Ib=templ;
end
clear templ
elseif choice==2,
    disp('')
    hmd=templ;
hmd=input('Hub height above reference datum/waterline (ft): ');
if isempty(hmd),
    hmd=templ;
end
clear templ
elseif choice==3,
    disp('')
    lmd=templ;
lmd=input('Hub fuselage station (ft): ');
if isempty(lmd),
    lmd=templ;
end
clear templ
elseif choice==4,
    disp('')
    ymd=templ;
ymb=input('Hub position right of buttline (ft): ');
if isempty(ymd),
    ymd=templ;
end
clear templ
elseif choice==5,
    disp('')
    im=57.3
    templ=im=57.3;
im=input('Mast incidence (negative forward - deg):')/57.3;
if isempty(im),
    im=templ/57.3;
end
clear templ
elseif choice==6,
clc
disp(' ')
htd
templ=htd;
htd=input('Tail rotor height above reference datum/waterline (ft):');
if isempty(htd),
    htd=templ;
end
clear templ
elseif choice==7,
clc
disp(' ')
ltd
templ=ltd;
ltd=input('Tail rotor fuselage station (ft):');
if isempty(ltd),
    ltd=templ;
end
clear templ
elseif choice==8,
clc
disp(' ')
ytd
templ=ytd;
ytd=input('Tail rotor position right of buttlime (ft):');
if isempty(ytd),
    ytd=templ;
end
clear templ
elseif choice==9,
clc
disp(' ')
btt
templ=bt;
btt=input('Number of tail rotor blades:');
if isempty(btt),
    btt=templ;
end
clear templ
elseif choice==10,
clc
disp(' ')
cot
templ=cot;
cot=input('Blade chord (ft):');
if isempty(cot),
    cot=templ;
end
clear templ
elseif choice==11,
clc
disp(' ')
Rt

102
temp1= Rt;
Rt=input('Tail rotor blade radius (ft): ');
if isempty(Rt),
    Rt=temp1;
end
clear temp1
elseif choice==12,
    clc
disp(' ')
at
    temp1=at;
at=input('Average lift curve slope of tail rotor: ');
    if isempty(at),
       at=temp1;
end
clear temp1
elseif choice==13,
    clc
disp(' ')
    ohmt
    temp1=ohmt;
    ohmt=input('Rotational velocity of tail rotor (rad/sec): ');
    if isempty(ohmt),
       ohmt=temp1;
end
clear temp1
elseif choice==14,
    clc
disp(' ')
Ibt
    temp1=Ibt;
    Ibt=input('Blade flapping moment of inertia (slug ft^2): ');
    if isempty(Ibt),
       Ibt=temp1;
end
clear temp1
elseif choice==15,
    clc
disp(' ')
    delta3=57.3
    temp1=delta3*57.3;
    delta3=input('Delta-3 angle (deg): ')/57.3;
    if isempty(delta3),
       delta3=temp1/57.3;
end
clear temp1
elseif choice==16,
    clc
disp(' ')
    theta1=57.3
    temp1=theta1*57.3;
    theta1=input('Blade twist (deg): ')/57.3;
    if isempty(theta1),
       theta1=temp1/57.3;
end
clear temp1
elseif choice==17,
    clc
disp(' ')
hvd
hvd = input('Height above reference datum/waterline (ft): '); 
if isempty(hvd), 
    hvd = temp1; 
end 
clear temp1 
elseif choice == 18, 
    clc 
disp(' ') 
lvd 
temp1 = lvd; 
lvd = input('Fuselage station (ft): '); 
if isempty(lvd), 
    lvd = temp1; 
end 
clear temp1 
elseif choice == 19, 
    clc 
disp(' ') 
yvd 
temp1 = yvd; 
yvd = input('Position right of buttline (ft): '); 
if isempty(yvd), 
    yvd = temp1; 
end 
clear temp1 
elseif choice == 20, 
    clc 
disp(' ') 
alplov = 57.3 
temp1 = alplov * 57.3; 
alplov = input('Zero lift angle for vertical tail (deg): ') / 57.3; 
if isempty(alplov), 
    alplov = temp1 / 57.3; 
end 
clear temp1 
elseif choice == 21, 
    clc 
disp(' ') 
clvertmax 
temp1 = clvertmax; 
clvertmax = input('Maximum Cl for vertical tail: '); 
if isempty(clvertmax), 
    clvertmax = temp1; 
end 
clear temp1 
elseif choice == 22, 
    clc 
disp(' ') 
qvq 
temp1 = qvq; 
qvq = input('Dynamic pressure ratio (pg 489 Prouty): '); 
if isempty(qvq), 
    qvq = temp1; 
end 
clear temp1 
elseif choice == 23, 
    clc 
disp(' ') 
av
disp('Lift curve slope of vertical tail: ');
if isempty(av),
    av=templ1;
end
clear templ1
else if choice==0,
    check2=0
    clc
else
    disp('Enter a displayed number ... press any key to continue')
pause
end

check2=1;
while check2 > 0
    clc
disp('')
disp(' *** STABILITY AND CONTROL MENU ***')
disp(' *** ADDITIONAL PARAMETERS (2 of 3) ***')

disp(' Horizontal Tail')
disp(' 1. height above waterline 2. fuselage station')
disp(' 3. posn right of buttline 4. alpha zero lift')
disp(' 5. angle of incidence 6. lift curve slope')
disp(' 7. dynamic pressure ratio 8. rotor downwash ratio')
disp(' 9. fuselage downwash ratio')

disp(' Wing')
disp(' 10. height above waterline 11. fuselage station')
disp(' 12. posn right of buttline 13. alpha zero lift')
disp(' 14. angle of incidence 15. lift curve slope')
disp(' 16. tip cord 17. root cord')
disp(' 18. rotor downwash ratio 19. fuselage downwash ratio')

disp(' 0. NO CHANGES')
choice=input('Input the parameter to change: ');
if choice==1
    clc
    disp('')
    hhd
    templ1=hhd;
    hhd=input('Height above reference datum/waterline (ft): ');
    if isempty(hhd),
        hhd=templ1;
    end
clear templ1
else if choice==2,
    clc
    disp('')
    lhd
    templ1=lhd;
    lhd=input('Fuselage station (ft): ');
    if isempty(lhd),
        lhd=templ1;
clear temp1
elseif choice==3,
clc
disp(' ')
yhd
templ=yhd;
yhd=input('Position right of buttline: '); 
if isempty(yhd),
yhd=templ;
end
clear temp1
elseif choice==4,
clc
disp(' ')
alploh=57.3
templ=alploh*57.3;
alploh=input('Zero lift angle for horizontal tail (deg): ') /57.3;
if isempty(alploh),
alploh=templ/57.3;
end
clear temp1
elseif choice==5,
clc
disp(' ')
ih=57.3
templ=ih=57.3;
ih=input('Angle of incidence of horizontal tail (deg): ') /57.3;
if isempty(ih),
ih=templ/57.3;
end
clear temp1
elseif choice==6,
clc
disp(' ')
ah
templ=ah;
ah=input('Lift curve slope of horizontal tail: '); 
if isempty(ah),
ah=templ;
end
clear temp1
elseif choice==7,
clc
disp(' ')
qhq
templ=qhq;
qhq=input('Dynamic pressure ratio (pg 489 Prouty): '); 
if isempty(qhq),
qhq=templ;
end
clear temp1
elseif choice==8,
clc
disp(' ')
vhvl
templ=vhvl;
vhvl=input('Rotor downwash ratio (pg 489 Prouty): '); 
if isempty(vhvl),
vhvl=templ;
end
clear templ
elseif choice==9,
clc
disp('')
detafdalpfh
templ=detafdalpfh;
detafdalpfh=input('Fuselage downwash ratio (pg 489 Prouty): '); 
if isempty(detafdalpfh),
    detafdalpfh=templ1;
end
clear templ
elseif choice==10
clc
disp('')
hwd
templ=hwd;
hwd=input('Height above reference datum/waterline (ft): '); 
if isempty(hwd),
    hwd=templ;
end
clear templ
elseif choice==11
clc
disp('')
lwd
templ=lwd;
lwd=input('Fuselage station (ft): '); 
if isempty(lwd),
    lwd=templ;
end
clear templ
elseif choice==12
clc
disp('')
ywd
tywd
templ=ywd;
ywd=input('Position right of buttline (ft): '); 
if isempty(ywd),
    ywd=templ;
end
clear templ
elseif choice==13,
clc
disp('')
alplow=57.3
alplow=input('Lift angle for wing (deg): ') / 57.3;
if isempty(alplow),
    alplow=templ/57.3;
end
clear templ
elseif choice==14,
clc
disp('')
iw=57.3
iw=input('Angle of incidence of wing (deg): ') / 57.3;
if isempty(iw),
    iw=templ/57.3;
clear templ
elseif choice==15,
cic
  disp('  ') 
  aw 
  templ=aw; 
  aw=input('Lift curve slope of wing: '); 
  if isempty(aw), 
    aw=templ; 
  end 
  clear templ 
elseif choice==16,
cic 
  disp('  ') 
  ctw 
  templ=ctw; 
  ctw=input('Tip cord (ft): '); 
  if isempty(ctw), 
    ctw=templ; 
  end 
  clear templ 
elseif choice==17,
cic 
  disp('  ') 
  crw 
  templ=crw; 
  crw=input('Root cord (ft): '); 
  if isempty(crw), 
    crw=templ; 
  end 
  clear templ 
elseif choice==18,
cic 
  disp('  ') 
  vvvl 
  templ=vvvl; 
  vvvl=input('Rotor downwash ratio (pg 489 Prouty): '); 
  if isempty(vvvl), 
    vvvl=templ; 
  end 
  clear templ 
elseif choice==19,
cic 
  disp('  ') 
  detafdalpfw 
  templ=detafdalpfw; 
  detafdalpfw=input('Fuselage downwash ratio (pg 489 Prouty): '); 
  if isempty(detafdalpfw), 
    detafdalpfw=templ; 
  end 
  clear templ 
else 
  check2=0 
  cic 
  disp('  ') 
  disp('enter a displayed number ... press any key to continue') 
  pause 
end
check2=1;
while check2 > 0

clc

disp(' *** STABILITY AND CONTROL MENU ***')
disp(' *** ADDITIONAL PARAMETERS (3 of 3) ***')

disp(' CG location and Inertias/fuselage parameters')
disp(' 1. cg ht. above waterline  2. cg fuselage station')
disp(' 3. cg posn rt of buttline  4. Ixx')
disp(' 5. Iyy  6. Izz')
disp(' 7. Ixz  8. fuselage downwash ratio')

disp(' NOTAR if available (enter zeros if using tail rotor)')
disp(' 9. height above waterline 10. boom fuselage station')
disp('11. boom position left ref 12. NOTAR diameter')
disp('13. swirl angle at boom 14. NOTAR max force')
disp('15. thruster fuselage station')

disp(' Rigging')
disp('16. Bi main/in defl (del e) 17. A1 main/in defl (dela)')
disp('18. theta0m/in defl (del c) 19. theta0t/pedal defl (del r or p)')
disp('20. NOTAR sleeve twist/defl 21. max rudder defl')

disp('0. NO CHANGES')

choice=input('Input the parameter to change: ');
if choice==1,
  clc
  disp(' ')
  zcg
  templ=zcg;
  zcg=input('CG height above reference datum/waterline (ft): ');
  if isempty(zcg),
    zcg=templ;
  end
  clear templ
elseif choice==2,
  clc
  disp(' ')
  xcg
  templ=xcg;
  xcg=input('CG Fuselage station (ft): ');
  if isempty(xcg),
    xcg=templ;
  end
  clear templ
elseif choice==3,
  clc
  disp(' ')
  ycg
  templ=ycg;
  ycg=input('CG position right of buttline (ft): ');
  if isempty(ycg),
    ycg=templ;
  end
end
clear templ
elseif choice==4,
    clc
    disp(' ')  
    Ixx
    templ=Ixx;
    Ixx=input('Ixx (slug ft^2): ');  
    if isempty(Ixx),
        Ixx=templ;
    end
    clear templ
elseif choice==5,
    clc
    disp(' ')  
    Iyy
    templ=Iyy;
    Iyy=input('Iyy (slug ft^2): ');  
    if isempty(Iyy),
        Iyy=templ;
    end
    clear templ
elseif choice==6,
    clc
    disp(' ')  
    Izz
    templ=Izz;
    Izz=input('Izz (slug ft^2): ');  
    if isempty(Izz),
        Izz=templ;
    end
    clear templ
elseif choice==7,
    clc
    disp(' ')  
    Ixz
    templ=Ixz;
    Ixz=input('Ixz (slug ft^2): ');  
    if isempty(Ixz),
        Ixz=templ;
    end
    clear templ
elseif choice==8,
    clc
    disp(' ')  
    vfv1
    templ=vfv1;
    vfv1=input('Downwash ratio for fuselage (page 513 Prouty): ');  
    if isempty(vfv1),
        vfv1=templ;
    end
    clear templ
elseif choice==9,
    clc
    disp(' ')  
    htnd
    templ=htnd;
    htnd=input('Height above reference datum/waterline (ft): ');  
    if isempty(htnd),
        htnd=templ;
    end
110
clear templ
elseif choice==10,
clc
disp('')
ytnd
templ=ytnd;  
ytnd=input('Fuselage station (ft): ');  
if isempty(ytnd),  
ytnd=templ;  
end
clear templ
elseif choice==11,
clc
disp('')
dian
templ=dian;  
dian=input('NOTAR boom diameter (ft): ');  
if isempty(dian),  
dian=templ;  
end
clear templ
elseif choice==12,
clc
disp('')
swirl*57.3
templ=swirl*57.3;  
swirl=input('Swirl angle at boom (deg): ') / 57.3;  
if isempty(swirl),  
swirl=templ / 57.3;  
end
clear templ
elseif choice==13,
clc
disp('')
Ytmaxn
templ=Ytmaxn;  
Ytmaxn=input('Maximum NOTAR thruster force (lbs): ');  
if isempty(Ytmaxn),  
Ytmaxn=templ;  
end
clear templ
elseif choice==14,
clc
disp('')
lttnd
Ytmaxn
templ=Ytmaxn;  
Ytmaxn=input('Thruster fuselage station (ft): ');  
if isempty(lttnd),  
lttnd=templ;  
end
clear temp1
elseif choice==16,
clc
disp(' ')
dblmdele=*57.3
templ=dblmdele*57.3;
dblmdele=input('Long cyclic pitch per inch defl (deg/in):
')/57.3;
if isempty(dblmdele),
dblmdele=templ/57.3;
end
clear temp1
elseif choice==17,
clc
disp(' ')
dalmdele=*57.3
templ=dalmdele*57.3;
dalmdele=input('Lateral cyclic pitch per inch defl (deg/in):
')/57.3;
if isempty(dalmdele),
dalmdele=templ/57.3;
end
clear temp1
elseif choice==18,
clc
disp(' ')
dthetomdele=*57.3
templ=dthetomdele*57.3;
dthetomdele=input('Collective pitch per inch defl (deg/in):
')/57.3;
if isempty(dthetomdele),
dthetomdele=templ/57.3;
end
clear temp1
elseif choice==19,
clc
disp(' ')
dthetotddelp=*57.3
templ=dthetotddelp*57.3;
disp('Tail rotor pitch change per inch defl or percentage of
 twist')
disp('Enter 0 (zero) if using NOTAR')
dthetotddelp=input(' (deg/in or deg/deg of twist): ') /57.3;
if isempty(dthetotddelp),
dthetotddelp=templ/57.3;
end
clear temp1
elseif choice==20,
clc
disp(' ')
sidearm/2
templ=sidearm/2;
disp('Maximum deflection of anti-torque from neutral for NOTAR,
 enter')
sidearm=input('1000 if using tail rotor (deg or inch travel):
')*2;
if isempty(sidearm),
sidearm=templ*2;
if sidearm==0,sidarm=1e3,end
clear templ

dphi=pi/sidearm; % pi rad sleeve twist/sidearm deflect

elseif choice==21,
    clc
    disp('  
maxr
    templ=maxr;
disp('Displacement of anti-torque control until full rudder')
disp('deflection. Enter 0 (zero) if rudder is fixed')
maxr=input('  (deg or inch travel): '); 
if isempty(maxr),
    maxr=templ;
end
    clear templ
elseif choice==0,
    clc
    check2=0;
else
    disp('  
disp('enter a displayed number ... press any key to continue')
pause
end

disp('  
disp('  
*** SAVE INSTRUCTIONS ***')
disp('  
disp('A. Save the new data to a specified file name.')
disp('B. Do not use an extension or quotations.')
disp('C. Use letter/number combinations of 6 characters or less.')
disp('D. The file will be saved with a "mat" extension.')
disp('  
disp('E. If you made no changes or want the same name, press enter.')
disp('  
disp('ex: desig2')
filename=filenamel;
filenamel=input('save file as: ','s');
if isempty(filenamel),
    filenamel=filename;
end
    clear check
eval(['save ',filenamel]);
    check=0;
end

% end

% *** If creating a new file: get input for required variables
% and save under desired file name. Saves to current
directory as a .mat file. ***

else
    check4=1;
while check4>0;
    clc
    disp('Do you want to use a Tail Rotor or NOTAR?')
temp=input('Tail Rotor=0, NOTAR=1: '); 
    if temp==0

113
notar=0;
check4=0;
elseif temp==1
    notar=1;
    check4=0;
else
    disp('');
    disp('Enter a 0 or 1')
    disp('press any key to continue...')
    pause
end
end
check4=1;
while check4>0;
    clc
    disp('Do you want to use a controlable vertical tail?')
    temp=input('No=0, Yes=1: ');
    if temp==0
        ctail=0;
        check4=0;
    elseif temp==1
        ctail=1;
        check4=0;
    else
        disp('');
        disp('Enter a 0 or 1')
        disp('press any key to continue...')
        pause
    end
end
if Swing<.1
    wing=0;
else
    wing=1;
end
clc
    disp('Main rotor')
    disp('')
    Ib=input('Blade flapping moment of inertia (slug ft^2): ');
    hmd=input('Hub height above reference datum/waterline (ft): ');
    lmd=input('Hub fuselage station (ft): ');
    ymd=input('Hub position right of buttline (ft): ');
    im=input('Mast incidence (negative forward - deg): ') / 57.3;
    clc
if notar==0
    disp('Tail rotor')
    disp('')
    htd=input('Tail rotor height above reference datum/waterline (ft): ');
    ltd=input('Tail rotor fuselage station (ft): ');
    ytd=input('Tail rotor position right of buttline (ft): ');
    bt=input('Number of tail rotor blades: ');
    ctt=input('Tail rotor blade chord (ft): ');
    Rtt=input('Tail rotor blade radius (ft): ');
    at=input('Average lift curve slope of tail rotor: ');
    ohmt=input('Rotational velocity of tail rotor (rad/sec): ');
    Ibt=input('Tail rotor blade flapping moment of inertia (slug ft^2): ');
    delta3=input('Delta-3 angle (deg): ') / 57.3;
thetalt=input('Blade twist (deg): ') / 57.3;
htnd=O;ltnd=O;ytnd=O;dian=O;swirl=O;Ytmaxn=0;lttnd=O;
else if notar=1
  clc
disp('NOTAR')
disp('')
  htnO;ldO;ytndO;dianO;swirlO;YtmaxnO;lttndO;
ed
clc
disp('Vertical tail')
disp('')
  hvdo;ldO;ytndO;dianO;swirlO;YtmaxnO;lttndO;
ed
clc
disp('Horizontal tail')
disp('')
  hhdo;ldO;ytndO;dianO;swirlO;YtmaxnO;lttndO;
ed
if wing=1
  clc
disp('Wing')
disp('')
  hwdo;ldO;ytndO;dianO;swirlO;YtmaxnO;lttndO;
ed
else if wing=0
  hwdo;ldO;ytndO;dianO;swirlO;YtmaxnO;lttndO;
ed
clc
disp('CG location')
disp(' ') zcg=input('CG height above reference datum/waterline (ft): ');
xcg=input('CG Fuselage station (ft): ');
ycg=input('CG position right of buttline (ft): ');
clc
disp('Fuselage moments of inertia/downwash parameter')
disp(' ')
Ixx=input('Ixx (slug ft\(^2\)): ');
Iyy=input('Iyy (slug ft\(^2\)): ');
Izz=input('Izz (slug ft\(^2\)): ');
Ixz=input('Ixz (slug ft\(^2\)): ');
vfvl=input('Downwash ratio for fuselage (page 513 Prouty): ');
clc
disp(' Rigging')
disp(' ')
dblmddele=input('Long cyclic pitch per inch defl (deg/in): ')/57.3;
dalmddela=input('Lateral cyclic pitch per inch defl (deg/in): ')/57.3;
dthetomddelc=input('Collective pitch per inch defl (deg/in): ');
if notar==0
   disp(' ') disp('Tail rotor pitch change per inch defl or percentage of twist')
dthetotddelp=input(' (deg/in or deg/deg of twist): ')/57.3;
dphinddelp=0;sidearm=1000;
elseif notar==1
   disp(' ') disp('Max deflection of anti-torque from neutral for NOTAR, enter ') sidearm=input(' 1000 if using tail rotor (deg or inch travel): ')\*2;
   if sidearm==0,sidearm=1000,end
   dphinddelp=pi/sidearm;
end
if ctail==1
   disp(' ') disp('Displacement of anti-torque control until full rudder')
   maxr=input(' deflection (deg or inch travel): ');
elseif ctail==0
   maxr=0
end
clc
eval(['save ',filenamel]);
save stabtemp filenamel
check=0;
end
if dian==0;
   notar=0;
if bt==0;
   disp(' You must have a tail rotor or NOTAR/thruster! ') check=1
end
else
   notar=1;
end
clc
disp(' ')
disp(' ')

116
disp('') *** DATA ENTRY COMPLETE ***

******************************************************************************************

disp('') *** EVALUATING STABILITY DERIVATIVES ***
disp('')
pause(3)

if Vinf<20
    hover  % call hover routine
elseif Vinf>=20
    cruise  % call cruise routine
end
stabout

******************************************************************************************
% TRIM.M
%
% Rotor trim subroutine

disp('')
disp(' EXECUTING ROTOR TRIM ROUTINE **')
% *** calculation of required parameters ***

rho=.002377*(-.000031*PA+(-.002*temp+1.118));

% *** first guess at rotor profile drag (H force) ***

if Vinf < 16.9,
    Drotor=0;
else
    Drotor=Vinf*(rho/.002377);
end

q=0.5*rho*Vinf^2;
Adisk=pi*R^2;
Vtip=omega*R;
Dfuse=q*Af;
CDwing=CDwing+(CLwing^2/(ewing*pi*(bwing^2/Swing)));
CDhoriz=CDhoriz+(CLhoriz^2/(.8*pi*(bhoriz^2/Shoriz)));
CDvert=CDvert+(CLvert^2/(.8*pi*(bvert^2/Svert)));
Dwing=q*CDwing*Swing;
Dhoriz=q*CDhoriz*Shoriz;
Dvert=q*CDvert*Svert;
Dftotal=(Dfuse+Dwing+Dhoriz+Dvert)-Taux;
Lwing=q*CLwing*Swing;
Lhoriz=q*CLhoriz*Shoriz;
Lvert=q*CLvert*Svert;
Lftotal=Lwing+Lhoriz;
alphaT=atan2(C(Dftotal+Drotor),(GW-Lftotal));
mu=Vinf*cos(alphaT)/Vtip;

% *** thrust calculation ***

if Vinf < 16.9,
    T=(1+(0.3*Afv/Adisk))*GW;
    CT=T/(Adisk*rho*Vtip^2);
else
    T=(GW-Lftotal)/cos(alphaT);
    CT=T/(Adisk*rho*Vtip^2);
end

% *** setup blade radius elements, azimuth elements, 
% induced velocity distributions, and determination 
% of coning angle and tip loss parameter ***

B=1-(sqrt(2*CT)/b);
Reff=B*R;
Rbar=Reff-e;
dr=(Reff-grip)/nbe;
r=grip:dr:Reff-dr;,
x=x+dr/2;
xTl=0.7; % *** first guess at rT ***
RbarT=xTl+Rbar;
mlade=wblade/32.17;
betao = asin((T/b*RbarT - (.5*(R-e)) e*wblade)/((.5*(R-e)) e)^2*omega^2*mblade));
betat = twist*(0.7-(r/R));
psi = 0.360/naz:360-360/naz; psi = psi'/57.3;

% *** induced velocity determination ***

if Vinf < 16.9,
    A = *pi;
    Bv = (b/2)*omega*a*cblade;
    Tv = 0;
    delT = T - Tv;
    while abs(delT) > .01*T
        theta = twist*(0.7-(r/R)) + thettoa;
        C = (-b/2)*cblade*omega^2*r*a.*thetav;
        vi = (-Bv + sqrt((Bv^2 - (4*A*C))/(2*A)));
        dTv = (b/2)*rho*(omega*r)^2*a.*(thetav - (vi.*(omega*r))) * cblade * dr;
        Tv = sum(dTv);
        delT = T - Tv;
        if delT < 0,
            thettoa = thettoa - 0.5*thettoa*abs(delT/T);
        else
            thettoa = thettoa + 0.5*thettoa*abs(delT/T);
        end
    end
else
    lambdaT = [1 - (2*mu*sin(alphaT)) mu^2*(sin(alphaT)^2+1) - 2*mu^3*sin(alphaT)
               mu^4*sin(alphaT) 2 - CT^2/4];
    lambdaT = max(real(roots(lambdaT)));
    vi = lambdaT*Vtip - Vinf*sin(alphaT);
    vi = vi*ones(r);
end

% *** first guess at theta ***

thetaic = 0.035*((0.0006e-3*Vinf^2 + 0.244e-3*Vinf)/0.105);
thetais = -0.087*(0.0006e-3*Vinf^2 + 0.244e-3*Vinf)/0.105;
theta = thettoa + thetaic.*cos(psi) + thetais.*sin(psi);

% *** rotor trimming routine ***

disp('');
disp('');
disp('');

% *** TRIMMING COLLECTIVE ***

k = 1;
error0 = (T* .02) + 1;
while abs(error0) > T* .02
    Tpsi = zeros(psi);
    thralc:
    error0 = T*(mean(Tpsi)*b);
    if error0 < -T*.02,
        thettoa = thettoa - 0.35*thettoa*abs(1.5*error0/T)*(1-mu);
    elseif error0 > T*.02,
        thettoa = thettoa + 0.35*thettoa*abs(1.5*error0/T)*(1-mu);
    end
end
theta=thetao+thetalc.*cos(psi)+thetals.*sin(psi);
if k > 1,
    if abs(error0) > abs(error1),
        clc
disp(' ')
disp('')
disp('This configuration will not trim')
disp('Try a lower airspeed or a new design')
disp('')
disp('Program execution terminated')
disp(')
error('*** END OF PROGRAM ***')
end
end
error1=error0;
k=k+1;
end

disp(' ')
*** TRIMMING CYCLIC ***

t0=clock;
k=1;
error0=((T/b)*rT1*(R-grip))/0.04+1;

while error0 > ((T/b)*rT1*(R-grip))/0.04
    time=time(clock,t0);
    if time > 15,
        disp(' ')
        disp('still trimming ...')
        t0=clock;
    end

    Mpsi(:,k)=zeros(psi);
tmcalc
theta=[theta theta(:,k)];
Mpsi=[Mpsi Mpsi(:,k)];

% *** calculation of initial dthetadM ***
if k < 2,
    theta(:,k+1)=theta(:,k)+0.25/57.3;
    Mpsi(:,k+1)=zeros(psi);
    k=k+1;
tmcalc
    k=k-1;
    dthetadM=(theta(:,k+1)-theta(:,k))/(Mpsi(:,k+1)-Mpsi(:,k));
end

% *** calculation of M first harmonic parameters ***
Mlc=2*sum(Mpsi(:,k).*cos(psi))/nai;
Mls=2*sum(Mpsi(:,k).*sin(psi))/nai;

% *** removal of first harmonic terms from Mpsi ***
Mpsi(:,k+1)=Mpsi(:,k)-Mlc.*cos(psi)-Mls.*sin(psi);

120
```matlab
% calculation of new theta

Mpsi(:,k+1)=Mpsi(:,k);
error0=max(delM)-min(delM);
if k > 1,
  if error0 > error1,
    clc
disp('')
disp('')
disp('This configuration will not trim')
disp('Try a lower airspeed or a new design')
disp('')
disp('Program execution terminated')
disp('')
error('*** END OF PROGRAM ***')
end
enderror1=error0;

%% calculation of new theta

Mpsi(:,k+1)=Mpsi(:,k)+(dtetadM.*delM);
theta(:,k+1)=theta(:,k)+(dtetadM.*delM);

if error0 <= ((T/b)*rT1*(r-grip))*0.04,
  thetalc=2*sum(theta(:,k).*cos(psi))/naz;
  thetals=2*sum(theta(:,k).*sin(psi))/naz;
else
  thetalc=2*sum(theta(:,k+1).*cos(psi))/naz;
  thetals=2*sum(theta(:,k+1).*sin(psi))/naz;
end

theta(:,k+1)=thetao+thetalc.*cos(psi)+thetals.*sin(psi);

%% calculation of new dthetadM

theta=[theta theta(:,k+1)];
Mpsi=[Mpsi Mpsi(:,k+1)];
theta(:,k+2)=theta(:,k)+0.25/57.3;
Mpsi(:,k+2)=zeros(Mpsi(:,k+1));
k=k+2;
tmcalc
k=k-2;
dteta=dsi=theta(:,k+1)+theta(:,k)/((Mpsi(:,k+2)-Mpsi(:,k)));
k=k+1;
end

disp('')
disp('')
disp('')

theta=theta(:,k);
k=1;
error0=(T*.01)+1;

while abs(error0) > T*.01
  Tpsi=zeros(psi);
  thrcalc
  error0=T-(mean(Tpsi)*b);
  if error0 < -T*.01,
    thetao=thetao-0.25*thetao*abs(1.25*error0/T)*(1-mu);
  end
end
```

121
elseif error0 > T*.01,
    thetau=thetaco+0.25*thetaco*abs(1.25*error0/T)*(1-mu);
end
thetaco=thetaco+thetalc.*cos(psi)+thetalc.*sin(psi);
if k > 1,
    if abs(error0) > abs(error1),
        clc
        disp('  
        disp('This configuration will not trim'.
        disp('Try a lower airspeed or a new design'.
        disp('  
        disp('Program execution terminated').
        disp('  
        error('*** END OF PROGRAM ***')
    end
end
error1=error0;
end
k=k+1;
end

% *** calculating drag moments ***

DMpsi=zeros(psi);
dmc

% *** calculating rotor H force ***

if Vinf < 16.9,
    Hrotor=0;
    dT=[dT dD];
    dD=[dD dD];
else
    dT=[dT dD];
    dD=[dD dD];
    for i=1:length(r)+1,
        Hic(i)=2*sum(dT(i,:).*cos(psi))/naz;
        His(i)=2*sum(dD(i,:).*sin(psi))/naz;
    end
    Hrotor=(((b*cos(alphaT)/2)* (sum(His)-sin(betao)*sum(Hic)))+Drotor)/2;
end

% *** calculating new rT ***

rT2=(((mean(Mpsi(:,length(Mpsi(1,:))-1))/mean(Tpsi))/R)+rT1)/2;

% *** check rotor drag and rT, retrim rotor if required ***

while abs(Drotor-Hrotor) > 0.2*Hrotor | abs(rT1-rT2) > 0.015*rT1
    if abs(Drotor-Hrotor) > 0.2*Hrotor,
        disp('  
        disp(' *** ADJUSTING ROTOR DRAG ***')
    end
    Drotor=Hrotor;
end
if abs(rT1-rT2) > 0.015*rT1,
    disp('')
    disp(' *** ADJUSTING MEAN THRUST LOCATION ***')
end

disp('')
disp(' *** RETRIMMING ROTOR ***')
dT=dT(:,1:nbe);
dD=dD(:,1:nbe);

% *** recalculating parameters ***
alphaT=atan((Dftotal+Drotor)/(GW-Lftotal));
mu=Vinf*cos(alphaT)/Vtip;

if Vinf >= 16.9, 
    T=(GW-Lftotal)/cos(alphaT); 
    CTE=T/(Adisk*rho*Vtip^2); 
    lambdat=[1 1 2]*mu*sin(alphaT) 
    mu^2*(sin(alphaT)^2+1)-2*mu^3*2*mu^4*sin(alphaT) 
    lamaT=max(roots(lambdat)); 
    vis=R2*Vtip*sin(alphaT); 
    vil=ones(r);
end

B=1-(sqrt(2*CT)/b);
Reff=B*R;
Rbar=Reff-e;
R=grip/Rbar;
Rbar=Rbar-T*Rbar;

betao=asin((T/b*Rbar-1.*R-e)*wblade)/(1.5*(R-e)+e)^2*omega^2*mblade);

% *** trimming collective ***
t0=clock;
k=1;
error0=(T*.02)+1;

while abs(error0) > T*.02
    Tpsi=zeros(1); 
    thrcalc 
    error0=T-(mean(Tpsi)*b); 
    if error0 < T*.02, 
        thetaco=thetao-0.35*thetaco*abs(1.5*error0/T)*(1-mu); 
    elseif error0 > T*.02, 
        thetaco=thetao+0.35*thetaco*abs(1.5*error0/T)*(1-mu); 
    end 
    theta=thetaco+thetalc.*cos(phi)+thetals.*sin(phi); 
    if k > 1, 
        if abs(error0) > abs(error1), 
            clc 
            disp('')
            disp('')
        end 
    end
disp('This configuration will not trim')
disp('Try a lower airspeed or a new design')
disp('')
disp('Program execution terminated')
disp('')
error('*** END OF PROGRAM ***')
end
end
error1=error0;
k=k+1;
end

% *** trimming cyclic ***

k=1;
error0=((T/b)*rT2*(R-grip))*0.04+1;

while error0 > ((T/b)*rT2*(R-grip))*0.04
    time=time(clock,t0);
    if time > 15,
        disp('still trimming ...')
        disp('')
        t0=clock;
    end
    Mpsi(:,k)=zeros(psi);
tmcalc
theta=[theta theta(:,k)];
Mpsi=[Mpsi Mpsi(:,k)];

% *** calculation of initial dthetadM ***

if k < 2,
    theta(:,k+1)=theta(:,k)+0.25/57.3;
    Mpsi(:,k+1)=zeros(psi);
    k=k+1;
tmcalc
k=k-1;
dthetadM=(theta(:,k+1)-theta(:,k))/(Mpsi(:,k+1)-Mpsi(:,k));
end

% *** calculation of M first harmonic parameters ***

Mlc=2*sum(Mpsi(:,k).*cos(psi))/naz;
Mls=2*sum(Mpsi(:,k).*sin(psi))/naz;

% *** removal of first harmonic terms from Mpsi ***

Mpsi(:,k+1)=Mpsi(:,k)-Mlc.*cos(psi)-Mls.*sin(psi);
delM=Mpsi(:,k+1)-Mpsi(:,k);
error0=max(delM)-min(delM);
if k > 1,
    if error0 > error1,
        clc
        disp('')
        disp('')
        disp('This configuration will not trim')
        disp('Try a lower airspeed or a new design')
    end
end
disp('')
disp('Program execution terminated')
disp('')
error('*** END OF PROGRAM ***')
end
end
error1=error0;

*** calculation of new theta ***
delM=0.5*(1-mu)*delM;
theta(:,k+1)=theta(:,k)+(dthetadM.*delM);
if error0 <= ((T/b)*rT2*(R-grip))* .04,
    thetalC=2*sum(theta(:,k).*cos(psi))/naz;
    thetalS=2*sum(theta(:,k).*sin(psi))/naz;
else
    thetalC=2*sum(theta(:,k+1).*cos(psi))/naz;
    thetalS=2*sum(theta(:,k+1).*sin(psi))/naz;
end
theta(:,k+1)=thetao+thetalC.*cos (psi)+thetalS.*sin (psi);

*** calculation of new dthetadM ***
theta=[theta theta(:,k+1)];
Mpsi=[Mpsi Mpsi(:,k+1)];
theta(:,k+2)=theta(:,k)+0.25/57.3;
Mpsi(:,k+2)=zeros(Mpsi(:,k+1));
k=k+2;
tmcacalc
k=k-2;
dthetadM=(theta(:,k+2)-theta(:,k))./(Mpsi(:,k+2)-Mpsi(:,k));
k=k+1;
end

*** retrimming collective ***
theta=theta(:,k);
k=1;
error0=(T*.01)+1;
while abs(error0) > T*.01
    Tpsi=zeros(psi);
    tmcacalc
    error0=T-(mean(Tpsi)*b);
    if error0 < -T*.01,
        thetao=thetao-0.25*thetao*abs(1.25*error0/T)*(1-mu);
    elseif error0 > T*.01,
        thetao=thetao+0.25*thetao*abs(1.25*error0/T)*(1-mu);
    end
    theta=thetao+thetalC.*cos (psi)+thetalS.*sin (psi);
    if k > 1,
        if abs(error0) > abs(error1),
            clc
disp('')
disp('')
disp('This configuration will not trim')
disp('Try a lower airspeed or a new design')
disp('')
disp('Program execution terminated')
        end
    end
end
disp('')
error('*** END OF PROGRAM ***')
end
end
error1=error0;
k=k+1;
end

% *** recalculating rotor H force ***
if Vinf < 16.9,
    Hrotor=0;
    dT=[dT ddT];
    dD=[dD ddD];
else
    dT=[dT ddT];
    dD=[dD ddD];
    for i=1:length(r)+1,
        Hlc(i)=2*sum(dT(:,i).*cos(psi))/naz;
        His(i)=2*sum(dD(:,i).*sin(psi))/naz;
    end
    Hrotor=((b*cos(alphaT)/2)*(sum(His)-sin(betao)*sum(Hlc)))+Drotor)/2;
end

% *** recalculating rT ***
rT1=rT2;
rT2=((mean(Mpsi(:,length(Mpsi(1,:))-1))/mean(Tpsi))/R)+rT1)/2;
end

% *** recalculating drag moments ***

dT=dT(:,1:nbe);
dD=dD(:,1:nbe);
DMpsi=zeros(psi);
dmcalc
dT=[dT ddT];
dD=[dD ddD];
clc
disp('')
disp('')
disp('')
*** ROTOR TRIMMED ***'')
pause(3)

126
C. CRUISE.M

% CRUISE.M
% CALLED BY STAB.M
% Computes the stability derivatives in cruise flight.
% calls the following subroutines
%
% CMRGRP
% CTRGRP
% CFUSEGRP
% TRIM
% DCTPLOTS or DCTMATS
%
% evaluate basic stability derivatives
%
% MAIN ROTOR
% dctsigsmdmu dctsigsdtheto dctsigsdlamp
% dchsigsmdmu dchsigsdtheto dchsigsdlamp
% dcqsigsmdmu dcqsigsdtheto dcqsigsdlamp
% daldmu daldtheto dalldlamp
% dbldmu dbldtheto dblldlamp
%
% TAIL ROTOR
% dctsigsdlampt dctsigsdmut dctsigsdhetot
%
\begin{verbatim}
vctsig=ones(1,3); vcsigsig=ones(1,3); vchsig=ones(1,3);
valtpp=ones(1,3); vAl=ones(1,3); vBi=ones(1,3);
vals=ones(1,3); vbis=ones(1,3);
vmu=ones(1,3); vtheta7=ones(1,3);
vao=ones(1,3); vvl=ones(1,3); vlam=ones(1,3); vthetao=ones(1,3); vctsigt=ones(1,3);
vmut=ones(1,3); vlampt=ones(1,3); vthetaot=ones(1,3); thetal=-twist;
end eval(['load ',filename]);
\end{verbatim}

Vinf=Vinf*1.0005; % .05% higher
\begin{verbatim}
disp('') disp(' *** FIRST OF THREE TRIM ITERATIONS ***')
disp('') pause(3)
\end{verbatim}

trim
Q rotor=mean(DMpsi)*b;
solidity=b*cblade/(pi*R);
CQ=Q rotor/(Adisk*rho*Vtip*2*R);
CH=H rotor/(Adisk*rho*Vtip*2);
vctsig(1,3)=CT/solidity;
vcsigsig(1,3)=CQ/solidity;
vchsig(1,3)=CH/solidity;
valtpp(1,3)=-alpha7;

127
VAL(1,3) = -theta1c;
vBL(1,3) = -theta1s;
vmu(1,3) = mu;
theta1 = thetao;
vtheta(1,3) = theta1 - .7*theta1;
vtheta2(1,3) = vtheta(1,3) + .7*theta1;
va(1,3) = beta;
v1(1,3) = mean(vi);
valm(1,3) = vmu(1,3)*valtpp(1,3) - vvl(1,3)/omega/R;
lockn0 = rho* blade*R^4/Ib;
num1 = vtheta(1,3)*((8/3+32/45*vmu(1,3)^3)/pi);
num2 = theta1*(2+vmu(1,3)^4/12);
num3 = valm(1,3) - 2*vmu(1,3)*vmu(1,3)/2;
den1 = 1+3/2*vmu(1,3)*vmu(1,3) - 5*vmu(1,3)^4/24;
vals(1,3) = vmu(1,3)*num1 + num2 + num3)/den1 - vBL(1,3);
vbls(1,3) = vals(1,3)*12/lockn0 - v(1/e)/R/1+e/3;
cqsig = CQ/solidity;
omm = omega;
sigma = solidity;
lv = lvd - xcg;
g = 32.2;
A = pi*R*R;
% tail rotor
if notar == 0
    Abt = bt*Rt*cot;
    At = pi*Rt*Rt;
    mut = Vinfin/ohm/Rt;
    sigmat = Abt/At;
    lt = ltcg;
    ht = htcg;
    yt = ytcg;
    locknot = rho*at*cot*R^4/Ibt;
    Tt = (cqsig*rho*A*1*ohm*R)^2*R-Lvert*lv/lt;
    ctsigt = Tt/(rho*At*(ohm*Rt)^2);
    lampt = ctsigt*sigma/2/mu;
    aot = 2/3*locknot*ctsigat/at - 3/2*g*Rt^2/1+(ohm*Rt)^2;
    alst = -2*(4/3*mut*aot + ctsigt*sigmat/2/mut)/(2*mut*mut)) * (1 - (4*mut/(2+3*mut*mut)) * 2 + (4*mut/(2+3*mut*mut)^2);
    tan(delta3) = (1 - (2*mut*mut)/(2+3*mut*mut)) + (1 - (4*mut/(2+3*mut*mut)^2) * tan(delta3)^2);
    bist = 2*4/3*mut*aot + ctsigt*sigmat/2/mut)/(2*mut*mut) + alst*tan(delta3);
    thetao = (((4*ctsigat/at) - (.5*mut*mut/2)*theta1t + mut*bist*tan(delta3) - lampt )/... 
        (2*mut*mut) - aot*tan(delta3)) + 
        theta75 = thetao + .75*theta1t;
    vlt = sqrt(Tt/2/rho/At);
    vctsigat(1,3) = ctsigt;
    vmu(1,3) = mut;
    vtheta(1,3) = thetao;
    vlamp(1,3) = lampt;
end
% save stabtemp vctsig vchs xsig valtpp vAl vB1 vmu vthetao theta1 ...
vtheta2 vao v1 v1 v1 v1 lockno vals vbls rho filename vctsig vmut ...
vthetao vlmapn g
clear
load stabtemp
eval({'load ' ,filenam1})
load stabtemp
Vinf=Vinf*.9995; % .05% lower
V=Vinf;
disp('')  
disp(' *** SECOND OF THREE TRIM ITERATIONS ***')
disp('')
pause(3)
trim
Qrotor=mean(DMpsi)*b;
solidity=b*cblade/(pi*R);
CQ=Qrotor/(Adisk*rho*Vtip'^2*R);
CH=Hrotor/(Adisk*rho*Vtip'^2);
vctsig(1,1)=CT/solidity;
vcsig(1,1)=CQ/solidity;
vchsig(1,1)=CH/solidity;
valttpp(1,1)=alphaT;
VA1(1,1)=thetaC;
vB1(1,1)=thetaS;
vmu(1,1)=mu;
thesasave=thetao;
vthetao(1,1)=-thetao+.7*thetao;
vtheta7(1,1)=vthetao(1,1)+.7*thetao;
vao(1,1)=betao;
v1(1,1)=mean(vi);
vlamp(1,1)=vmu(1,1)-valttpp(1,1)-v1(1,1)/omega/R;
lockno=rho*a*cblade*R'*4/Ib;
num1=vthetao(1,1)*(8/3+32/45*vmu(1,1)'^3/pi);
num2=thetao*(2+vmu(1,1)'^4/12);
num3=vlamp(1,1)*(2-vmu(1,1)*vmu(1,1)/2);
den1=1+3/2*vmu(1,1)*vmu(1,1)-5*vmu(1,1)'^4/24;
vals(1,1)=vmu(1,1)*(num1+num2+num3)/den1-vB1(1,1);
vals1(1,1)=vals(1,1)*12/lockno*e/R/(1+e/R/3);
cqsig=CQ/solidity;
chm=omega;
sigma=solidity;
1v=1vld-xcg;
A=pi*R*R;

if notar==0
    Abt=bt*Rt*Cot;
    At=pi*Rt*Rt;
    mut=Vinf/omega/Rt;
    sigmat=Abt/At;
    l1=ltd-xcg;
    ht=htd-xcg;
    yt=ytd-ycg;
    lockno=rho*at*cot*Rt'^4/Ibt;
    Tt=(cqsig*sigma*rho*A*(chm*R)'^2*R-Lvert*1v)/l1;
    ctsigt=Tt/(rho*At*(chm*Rt)'^2);
    lampt=-ctsigt*sigma/2/mu;
aot=2/3*lockno*ctsigt/at-3/2*g*Rt'^2/(chm*Rt)'^2;
else
    aot=-2*((4/3*mut*aot+ctsigt*sigmat/2/mut)/(2+mut*mut))((1-(4*mut/(2+3*mut*mut))'^2)*...
\[ \tan(\delta_3) = \left( \frac{(2-\mu^*\mu)/(2+3*\mu^*\mu) + (1-(4*\mu^*/(2+3*\mu^*\mu))^2) \cdot \tan(\delta_3)^2}{\mu^*} \right)^2; \]

\[ \text{BLST.2} = \frac{(4/3*\mu^*\alpha^*/\sigma^*/\sigma^*/2/\mu)}{(2+\mu^*\mu)} \times \text{ALAT} \times \tan(\delta_3); \]

\[ \theta_{\text{OTS}} = \left( \frac{(4*\alpha^*/\sigma^*)}{-0.5*\mu^*\mu/2} \right)^2 \times \theta_{\text{ALT}} \times \tan(\delta_3) \times \tan(\delta_3); \]

\[ \theta_{75\text{T}} = \theta_{\text{OTS}} + 0.75 \times \theta_{\text{ALT}}; \]

\[ \text{VCTSIG} = (1, 1); \]

\[ \text{VCQSIG} = (1, 1); \]

\[ \text{VCHSIG} = (1, 1); \]

\[ \text{VALTPP} = (1, 1); \]

\[ \text{VAL} = (1, 1); \]

\[ \text{VMU} = (1, 1); \]

\[ \text{VTHETA0} = (1, 1); \]

\[ \text{VTHETA7} = (1, 1); \]

\[ \text{VAO} = (1, 1); \]

\[ \text{VVL} = (1, 1); \]

\[ \text{VLAMPT} = (1, 1); \]

\[ \text{END} \]

\[ \text{SAVE STABTEMP VCTSIG VCQSIG VCHSIG VALTPP VAL VBL VTHETA THETA0 \ldots} \]

\[ \text{THETA7 VAO VVL VLAMP LOCKNO VALS VBLS RHO FILENAME1 VCTSIG VMU \ldots} \]

\[ \text{VTHETA0 VAO VV1 VLAMP LOCKNO VALS VBLS RHO FILENAME1 VCTSIG VMU \ldots} \]

\[ \text{EXECUTE g} \]

\[ \text{CLEAR} \]

\[ \text{LOAD STABTEMP} \]

\[ \text{EVAL(’load’, filename1))} \]

\[ \text{LOAD STABTEMP} \]

\[ \text{DISP(’ ’)} \]

\[ \text{DISP(’ ’)} \]

\[ \text{PAUSE(3)} \]

\[ \text{TRIM} \]

\[ \text{QROT}=\text{mean(DMPSI)}*b; \]

\[ \text{SOLIDITY}=b*cblade/(\pi*R); \]

\[ \text{CQ} = \text{QROT} \times \text{Adisk} \times \rho \times \text{Vtip}^2 \times \text{R}; \]

\[ \text{CH} = \text{HROT} \times \text{Adisk} \times \rho \times \text{Vtip}^2 \times \text{R}; \]

\[ \text{DISP(’ ’)} \]

\[ \text{DISP(’ ’)} \]

\[ \text{DISP(’ ’)} \]

\[ \text{DISP(’ ’)} \]

\[ \text{DISP(’ ’)} \]

\[ \text{VCTSIG}(1, 2) = \text{CT/SOLIDITY}; \]

\[ \text{VCQSIG}(1, 2) = \text{CQ/SOLIDITY}; \]

\[ \text{VCHSIG}(1, 2) = \text{CH/SOLIDITY}; \]

\[ \text{VALTPP}(1, 2) = -\alpha_T; \]

\[ \text{VAL}(1, 2) = -\theta_T; \]

\[ \text{VAL}(1, 2) = -\theta_L; \]

\[ \text{VMU}(1, 2) = \mu; \]

\[ \text{VTHETA0} = \text{VTHETA0}; \]

\[ \text{VTHETA0}(1, 2) = \text{VTHETA0}(1, 2) - 0.75 \times \theta_T; \]

\[ \text{VTHETA7}(1, 2) = \text{VTHETA0}(1, 2) + 0.75 \times \theta_T; \]

\[ \text{VAC}(1, 2) = \beta_T; \]

\[ \text{VVL}(1, 2) = \text{MEAN(VI)}; \]

\[ \text{VAMP}(1, 2) = \text{VMU}(1, 2) \times \text{VALTPP}(1, 2) - \text{VVL}(1, 2) \times \text{OMEGA}/R; \]

\[ \text{V=VINF; \]

\[ \text{LOCKNO} = \rho \times a \times \text{CLADE} \times R^4 / Rb; \]

\[ \text{NUM1} = \text{VTHETA0}(1, 2) \times (8/3 + 32/45 \times \text{VMU}(1, 2)^3 / \pi); \]

\[ \text{NUM2} = \text{THETA0} \times (2 + \text{VMU}(1, 2)^4 / 12); \]

\[ \text{NUM3} = \text{VLAMP}(1, 2)^2 \times (2 - \text{VMU}(1, 2)^2)^2 / 2; \]

\[ \text{DEN1} = 1 + 3/2 \times \text{VMU}(1, 2)^2 \times \text{VMU}(1, 2)^2 - 5 \times \text{VMU}(1, 2)^4 / 4; \]

\[ \text{VALS}(1, 2) = \text{VMU}(1, 2)^2 \times (\text{NUM1} + \text{NUM2} + \text{NUM3}) / \text{DEN1} \times \text{VBL1}(1, 2); \]

\[ \text{VBLS}(1, 2) = \text{VALS}(1, 2)^2 / 12 \times \text{LOCKNO} \times e / R \times (1 + e/R); \]

\[ \text{CSIG} = \text{CQ/SOLIDITY}; \]

\[ \text{OMEGA} = \text{OMEGA}; \]

\[ \text{130} \]
sigma=solidity;
lv=lvd-xcg;
A=pi*R*R;

% tail rotor
%
if notar==0
    Abt=bt*R*cot;
    At=pi*R*R;
    mut=Vin/ohmt/Rt;
    sigmat=Abt/At;
    lt=1td-xcg;
    ht=htd-xcg;
    yt=ytd-ycg;
    locknot=rho*at*cot*Rt^4/Ibt;
    Tt=(cqs*sigma*rho*A*(ohmt*R)^2*lt)/lt;
    ctsigt=Tt/(rho*At*(ohmt*R)^2);
    lampt=ctsigt*sigma/2/mu;
    aot=2/3*locknot*ctsigt/at-3/2*g*Rt^2/(ohmt*Rt)^2;

    alst=(1.2*((4/3*mut*aot+ctsigt*sigmat/2/mu)/(2+mut*mut)))*(1-(4*mut/(2+3*mut*mut))^2)*... 
        tan(delta3))/(((2-mut*mut)/(2+3*mut*mut))+(1-(4*mut/(2+3*mut*mut))^2)*tan(delta3)^2); 
    biset=2*(((4/3*mut*aot*ctsigt*sigma/2/mu)/(2+mu*mu)+alst*tan(delta3));
    theta=((-4*ctsigt/at)-(1+mut/mu)*theta7); 

    vlt=sqrt(Tt/2/rho/At);
    vctsigt(1,2)=ctsigt;
    vmut(1,2)=mut;
    vtheta(1,2)=theta;
    vlampt(1,2)=lampt;
end
%
save stabtemp vctsig vcqsig vchsig valtpp vAl vB1 vmu vthetao thetal ... 
vtheta7 vao vvl vlampl lockno vals vb1s T rho filenamel
%
% Solving for unknowns
%
DCPLOTS
%
dcysigdb=inv(T*sin(.1));
%
if notar==0
    dctplott
end

% CONVERSION CALCULATION
%
% converting data from crank
theta7=vthetao(1,2);
thetao=theta7-.7*theta1;
ctsigt=vctsigt(1,2);
chsig=vchsig(1,2);
cqs=tvcqsigt(1,2);
altp=valtpp(1,2);
Al=vAl(1,2);
B1=vB1(1,2);
a0=vao(1,2);
v1=vvl(1,2);
V=Vinf;

% Main rotor
hm=hmd-xcg;
ym=ymd-ycg;
lm=lmd-xcg;
c=cbblade;
Ab=c*R^4;
lamp=vlamp(1,2);
mu=vmu(1,2);
ct=CT;
f=Af0;

thetat=4/a*ctsig+sqrt(sigma*ctsig/2);
lockno=rho*a*c*R^4/Ib;
theta75=thetao+.75*thetal;
alse=vals(1,2);
bls=vbls(1,2);
m=GW/g;
Ic=Ixx*Izz;
lh=lhd-xcg;
hh=hhd-zcg;
yh=ychd-ycg;
hv=hvd-zcg;
vy=yyd-ycg;
delvmax=pi/4;  % test max vertical fin defl before stall
if maxr==0
    delvddelp=0;
else
    delvddelp=delvmax/maxr;  % vertical fin deflection/maxr
end

if notar==1;
    NOTAR
    v1lmax=1.2*v1;  % downwash at the boom slots (NOTAR)
ltn=ltn-d-xcg;
ln=ltn-xcg;
yh=ytnd-xcg;
hn=htnd-zcg;
htn=hn;

Tt=1**2*sigma*rho*(ohm*R)^2*R-Lvert*lv)/ltn;
cmun=.55;  % a typical value
bn=.5*R;
hslot=0.28*dian/2;
arm=bn/dian;
At=pi*dian*dian/4;
Rt=dian/2;
end

if Swing<.1
    wing=0;
else
    Aw=Swing;
    lw=lwd-xcg;
    hw=hwd-zcg;
    yw=ywd-ycg;
    alpw=CLwing/aw;
    deliw=swing;
    Zw=Lwing;
    bw=bwing;

132
cdow=CDowing;
    wing=1;
end
Av=Svert;
    bv=bsert;
    Ah=Shoris;
    bh=hsoriz;
    cdov=CDovert;
    cdoh=CDohoriz;
    deliv=ewing;
    delih=ewing;
    alph=CLhoriz/ah;
    Zh=Lhoriz;
    Yv=Lvert;
if Yv<.1;Yv=.1; end;
    deldv =1.72*(1-b/(20+Rt))*(2*Yv/(2*Rt+bv));
% trim conditions
    th0=-T*(im+als)/GW;
    ph0=-(T*bis+Tt)/GW;
    wo=0;
    vo=0;
    u0=Vinf;
    gamc=0;
% STABILITY CALCULATION
% cmrgrp
cmrgrp
% cbodygrp
% CHECK THE WING CALCULATIONS
% A11 = (dxdxdotm+dxdxdotth+dxdxdotf+dxdxdott)/m;
A12 = (dxdzdotm+dxdzdotth+dxdzdotf+dxdzdotw)/m;
A13 = (dxdqm)/m-wo;
A14 = -g*cos(th0);
A15 = (dxdydotm+dxdydotv)/m;
A16 = dxmp/m;
A17 = 0;
A18 = 0;%(dxdw)/m+vo; recheck this one
% A21 = (dxdxdotm+dxdxdotth+dxdxdotf+dxdxdott)/m;
A22 = (dxdzdotm+dxdzdotth+dxdzdotf+dxdzdotw)/m;
A23 = (dzdqh-dzdqw)/m-uo;
A24 = -g*cos(ph0)*sin(th0);
A25 = 0;%(dzdydotw)/m;
A26 = 0;%(dzdpw)/m-vo;
A27 = -g*sin(ph0)*cos(th0);
A28 = dzdrm/m;% (dzdrm+dzdrw)/m; recheck this one
% A31 = (dmdxdotm+dmdxdotth+dmdxdotf+dmdxdott)/Iyy;
A32 = (dmdzdotm+dmdzdotth+dmdzdotf+dmdzdotw)/Iyy;
A33 = (dmdqm+dmdqh+dmdqw)/Iyy;
A34 = 0;
A35 = dmdydotm/Iyy;
% A36 = (dmdpm)/Iyy;
A37 = 0;
A38 = \(0\); \(\text{mdrf}/I_{yy}\);
A41 = 0;
A42 = 0;
A43 = \(\cos(\phi)\);
A44 = 0;
A45 = 0;
A46 = 0;
A47 = 0;
A48 = \(-\sin(\phi)\);  

A51 = \(\frac{(dydxdotm+dydxdotm+dydxdotn+dydxdotv)}{m}\);
A52 = \(\frac{(dydxdotm+dydxdotn)}{m}\);
A53 = \(dydgm/m\);
A54 = \(-g\sin(\phi)\sin(\theta)\);
A55 = \(\frac{(dydydmt+dydydtn+dydydov+dydydof)}{3}\);
A56 = \(\frac{(dydpn+dydpt+dydpv)}{m+wo}\);
A57 = \(g\cos(\phi)\cos(\theta)\);
A58 = \((dydr+dydn+dydvr)/m-uo\);

A61 = \((I_{zz}*(drdxdotm+drdxdotn+drdxdotn+drdxdotn)/Ic)\)
A62 = \((I_{zz}*(drdxdotm+drdxdotn)+I_{xz}*(dndxdotm+dndxdotn))/Ic\);
A63 = \((I_{zz}*(drdqm)+I_{xz}*(0))/Ic\); % recheck this one
A64 = 0;
A65 = \((I_{zz}*(drdydotm+enydodt+drdydotv+drdydotf))\)
A66 = \((I_{zz}*(drdpt+drdpm+drdpm+drdpm)+I_{xz}*(dndpt+dndpn+dnddp))/Ic\);
A67 = 0;
A68 = \((I_{zz}*(drdrt+drdrn+drdrv+drdrv)+I_{xz}*(dndrm+dndr+drdrn+dndrn))/Ic\);

% A71 = 0;
A72 = 0;
A73 = \(\sin(\phi)\tan(\theta)\);
A74 = 0;
A75 = 0;
A76 = 0;
A77 = 0;
A78 = \(\cos(\phi)\tan(\theta)\);

% A81 = \((I_{xz}*(drdxdotm+drdxdotm+drdxdotn+drdxdotn)/Ic)\)
A82 = \((I_{xz}*(drdxdotm+drdxdotn)+I_{xz}*(dndxdotm+dndxdotn))/Ic\);
A83 = \((I_{xz}*(drdqm)+I_{xz}*(0))/Ic\); % recheck
A84 = 0;
A85 = \((I_{xz}*(drdydotm+dydydtn+drdydotv+drdydotf))\)
A86 = \((I_{xz}*(drdpm+drdpm+drdpm+drdpm)+I_{xz}*(dndpt+dndpn+dnddp))/Ic\);
A87 = 0;
A88 = \((I_{xz}*(drdrt+drdrv+drdrv)+I_{xz}*(dndrm+dndr+dndrn+dndrn))/Ic\);

% longitudinal plant augmented is X=[u w q theta]'
flonaug=[A1 A2 A3 A4;A11 A12 A13 A14;A21 A22 A23 A24;A31 A32 A33 A34];
flonaug=poly(flonaug);
flonaug=roots(flonaug);

% lateral plant augmented is X=[v p phi r psi]'
plotaug=[A55 A56 A57 A58;A65 A66 A67 A68;A75 A76 A77 A78;A85 A86 A87 A88;0 0 0 1 0];
plotaug=poly(plotaug);
Rlataug=roots(Plataug);
coupled plant
```
Amat=[A11 A12 A13 A14 A15 A16 A17 A18 0;
A21 A22 A23 A24 A25 A26 A27 A28 0;
A31 A32 A33 A34 A35 A36 A37 A38 0;
A41 A42 A43 A44 A45 A46 A47 A48 0;
A51 A52 A53 A54 A55 A56 A57 A58 0;
A61 A62 A63 A64 A65 A66 A67 A68 0;
A71 A72 A73 A74 A75 A76 A77 A78 0;
A81 A82 A83 A84 A85 A86 A87 A88 0;
0 0 0 0 0 0 0 0 1 0];
Pcoup=poly(Amat);
Rcoup=roots(Pcoup);
```
B11= dxdblm*dblmddele/m;
B12= dxdthetom*dthetomddelc/m;
B13= dxdaln*dalnndela/m;
B14= 0;
B21= dxdblm*dblmddele/m;
B22= dxdthetom*dthetomddelc/m;
B23= 0;
B24= 0;
B31= dmdblm*dblmddele/Iyy;
B32= dmdthetom*dthetomddelc/Iyy;
B33= dmdaln*dalnndela/Iyy;
B34= 0;
B41= 0;
B42= 0;
B43= 0;
B44= 0;
B51= dydblm*dblmddele/m;
B52= dydthetom*dthetomddelc/m;
B53= dydaln*dalnndela/m;
B54= (dydphin*dphinddelp+dyddelv*ddelvddelp+dydthetot*dthetotddelp)/m;
B61= Ixz*drdblm*dblmddele/Ic;
B62= (Ixz*drdthetom*dthetomddelc+Ixz*dndthetom*dthetomddelc)/Ic;
B63= Ixz*drdaln*dalnndela/Ic;
B64= (Ixz*(drdphin*dphinddelp+drddelv*ddelvddelp+drdthetot*dthetotddelp)+...
Ixz*(dndphin*dphinddelp+dnddelv*ddelvddelp+dndthetot*dthetotddelp))/Ic;
B71= 0;
B72= 0;
B73= 0;
B74= 0;
B81= Ixz*drdblm*dblmddele/Ic;
B82= (Ixz*drdthetom*dthetomddelc+Ixz*dndthetom*dthetomddelc)/Ic;
B83= Ixz*drdaln*dalnndela/Ic;
B84= (Ixz*(drdphin*dphinddelp+drddelv*ddelvddelp+drdthetot*dthetotddelp)+...
Ixz*(dndphin*dphinddelp+dnddelv*ddelvddelp+dndthetot*dthetotddelp))/Ic;
Glonaug=[B11 B12 B13 B14;B21 B22 B23 B24;B31 B32 B33 B34;B41 B42 B43 B44];
Glataug=[B51 B52 B53 B54;B61 B62 B63 B64;B71 B72 B73 B74;B81 B82 B83 B84;
0 0 0 0];
% coupled input matrix
Bmat=[B11 B12 B13 B14;B21 B22 B23 B24;B31 B32 B33 B34;B41 B42 B43 B44;
B51 B52 B53 B54;B61 B62 B63 B64;B71 B72 B73 B74;B81 B82 B83 B84;0 0 0 0];
% xcouple=12/lockno*e/R/(1+e/3/R);
% designed damping
desdmdq=dmdqm+dmdqh+dmdqw;
desdrdp=drdpm+drdpt+drdpw+drdpv+drdpn;
desdnr=dnrnm+dnrnt+dnrvt+dnrrn;
% cooper harper pilot rating
prpitch=desdmdq/Iyy;
prroll=desdrdp/Ixx;
pryaw=desdnr/Izz;
% control power
cppitch=B31*Iyy;
cproll=B63*Ixx;
cpyaw=B84*Izz;
cpipitch=B31;
cpiroll=B63;
cpiyaw=B84;
% theta0=theta7;

136
D. DCTPLOTS.M

% DCTPLOTS.M
% plots ct/sigma, ch/sigma, cq/sigma, als and bls
% VERSUS
% mu, thetao and lamp
% fits a polynomial to the curve, takes a derivative of the curves
% then evaluates the appropriate parameters for CRUISE.M
% d(ct/sigma), d(ch/sigma), d(cq/sigma), dals and dbls
% VERSUS
% dm, dthetao and dlamp
% uses the perturbation points from CRANK.M about nominal
% polyfitting data to obtain plots for n=8
% n=2;
ectsigmu=polyfit(vmu,vctsig,n);
ectsigthetao=polyfit(vthetao,vctsig,n);
ectsiglamp=polyfit(vlamp,vctsig,n);
% echsigmu=polyfit(vmu,vchsig,n);
echsigtetao=polyfit(vthetao,vchsig,n);
echsigtlamp=polyfit(vlamp,vchsig,n);
% ecqsigtetao=polyfit(vmu,vcqsig,n);
echsigtetao=polyfit(vmu,vcqsig,n);
echsigtlamp=polyfit(vlamp,vcqsig,n);
% ealsmu=polyfit(vmu,vals,n);
ealsdthetao=polyfit(vthetao,vals,n);
ealsdlamp=polyfit(vlamp,vals,n);
% eblsmu=polyfit(vmu,vbls,n);
eblsthetao=polyfit(vthetao,vbls,n);
eblslamp=polyfit(vlamp,vbls,n);
% % taking derivatives
% edctsigdmu=deriv2(ectsigmu);
edctsigdthetao=deriv2(ectsigthetao);
edctsiglamp=deriv2(ectsiglamp);
% edchsigtetao=deriv2(echsigtetao);
edchsigtlamp=deriv2(echsigtlamp);
% edcqsigtetao=deriv2(ecqsigtetao);
edcqsigtlamp=deriv2(ecqsigtlamp);
% edalsmu=deriv2(ealsmu);
edalsdthetao=deriv2(ealsdthetao);
edalsdlamp=deriv2(ealsdlamp);
% edblsmu=deriv2(eblsmu);
edblsthetao=deriv2(eblsthetao);
edblslamp=deriv2(eblslamp);
% % evaluating derivatives
\`\`dctsigdmu=polyval(edctsigdmu,vmu(1,2));
dctsigdtheto=polyval(edctsigdtheto,vthetao(1,2));
dctsigdlamp=polyval(edctsigdlamp,vlamp(1,2));
\`
\`dchsigdmu=polyval(edchsigdmu,vmu(1,2));
dchsigdtheto=polyval(edchsigdtheto,vthetao(1,2));
dchsigdlamp=polyval(edchsigdlamp,vlamp(1,2));
\`
\`dcqsigdmu=polyval(edcqsigdmu,vmu(1,2));
dcqsigdtheto=polyval(edcqsigdtheto,vthetao(1,2));
dcqsigdlamp=polyval(edcqsigdlamp,vlamp(1,2));
\`
\`dalsdmu=polyval(edalsdmu,vmu(1,2));
dalsdtheto=polyval(edalsdtheto,vthetao(1,2));
dalsdlamp=polyval(edalsdlamp,vlamp(1,2));
\`
\`dblsdmu=polyval(edblsdmu,vmu(1,2));
dblsdtheto=polyval(edblsdtheto,vthetao(1,2));
dblsdlamp=polyval(edblsdlamp,vlamp(1,2));
\`
E. DCTPLOTT.M

% DCTPLOTT.M
% plots ct/sigmat
% VERSES
% mut, thetaot and lampt
% fits a polynomial to the curve, takes a derivative of the curves
% then evaluates the appropriate parameters for CRUISE.M
% d(ct/sigma)
% VERSES
% dmut, dthetaot and dlampt
% uses the perturbation points from JANRAD about nominal
% polyfitting data to obtain plots for n=2
n=2;
ectsigmut=polyfit(vmut,vctsigt,n);
ectsigthetot=polyfit(vthetaot,vctsigt,n);
ectsiglampt=polyfit(vlampt,vctsigt,n);
% taking derivatives

edctsigdmut=deriv2(ectsigmut);
edctsigdthetot=deriv2(ectsigthetot);
edctsigdlampt=deriv2(ectsiglampt);
% evaluating derivatives

dctsigdmut=polyval(edctsigdmut,vmu(1,2));
dctsigdthetot=polyval(edctsigdtheto,vthetao(1,2));
dctsigdlampt=polyval(edctsigdlamp,vlamp(1,2));
F. CRGRT.N

% CRGRPN
% CALLED BY CRUISE.N
% Computes the basic MR derivatives in cruise flight.
% Computes the stability derivatives of the main rotor in cruise flight.
% Uses data loaded in the workspace by JANRAD.M and STAB.M
% Compute the basic main rotor derivatives.

dmudxdot=1/ohm/R;
dlampdxdotadle=R*(altp-p-sigma/2/mu*(dthsigdlamp*dthsig)*mu));
dlampsdxdot=1/(ohm*R*{(1+dthsigdlamp*sigma/2/mu));
dbetdydot=1/V;
dchsigt=dchsigt+a/glamp;
dchsiga=dchsiga;
dal*=-16/(lockno*ohm*(1-e/R)^2*(1-mu*mu/2));
12*e/R/(lockno*ohm*(1-e/R)^3*(1-mu*mu/4/4));
dalp=1/(ohm*(1-mu*mu/2))*192*e/R/(lockno*ohm*(1-e/R)^5*(1-mu*mu/4/4));
dalda=-12*e/R*(1-mu*mu/2)/(lockno*(1-e/R)^3*(1-mu*mu/4/4));
daldb=-1*(1-3/2*mu*mu/2)/(1-mu*mu/2);
dbldq=-inv(ohm*(1-mu*mu/2))+(192*e/R/(lockno*ohm*(1-e/R)^5*(1-mu*mu/4/4));
dbldp=-16/(lockno*ohm*(1-e/R)^2*(1-mu*mu/2));
12*e/R/(lockno*ohm*(1-e/R)^3*(1-mu*mu/4/4));
dblda=1;
dblda=12*e/R*(1+3/2*mu*mu/2)/(lockno*(1-e/R)^3*(1-mu*mu/4/4));
dmdalamu=34*e/R*Ab*rho*R*(ohm*R)^2*a/lockno;
dmdalam=-dmdalam;

% Compute the main rotor stability derivatives.

dxudxdot=x=(ohm*R)^2((dchsigt*dchsigt*daldmu+(als+im)*dchsigtdu))
*dmudxdot+((dchsigt*dchsigt*daldmu+als+im)*dchsigtdu);,
dsudxdot=dxudxdot;
duydxdot=x=(ohm*R)^2*(dchsigt*daldmu)*daldmu+als+im)*dchsigtdu;,
dzudxdot=x=(ohm*R)^2*(dchsigt)*daldmu;
dwn=1;
dw=12*e/R*(1+3/2*mu*mu/2)/(lockno*(1-e/R)^3*(1-mu*mu/4/4));
dmdalam=34*e/R*Ab*rho*R*(ohm*R)^2*a/lockno;
\[
\begin{align*}
\text{drdxdotm} &= \text{drdblsm} \cdot \text{dbldlamp} \cdot \text{dlampdxdot} + \text{dydsdotm} \cdot \text{hm} + \text{dzdxdotm} \cdot \text{ym}; \\
\text{drdqm} &= \text{drdblsm} \cdot \text{dbldq} + \text{dydqm} \cdot \text{hm}; \\
\text{drdpm} &= \text{drdblsm} \cdot \text{dbldp} + \text{dydpm} \cdot \text{hm}; \\
\text{drdthetom} &= \text{drdblsm} \cdot \text{dbldtheto} + \text{dydthetom} \cdot \text{hm} + \text{dzdthetom} \cdot \text{ym}; \\
\text{drdalm} &= \text{drdblsm} \cdot \text{dblda} + \text{dydalm} \cdot \text{hm}; \\
\text{drdblm} &= \text{drdblsm} \cdot \text{dbldb} + \text{dydblm} \cdot \text{hm}; \\
\text{dmdxdotm} &= \text{dmdalm} \cdot (\text{daldmu} \cdot \text{dmudxdot} + \text{daldlamp} \cdot \text{dlampdxdot}) + \text{dxdxdotm} \cdot \text{hm} + \text{dzdxdotm} \cdot \text{lm}; \\
\text{dmdydotm} &= -\text{dxdydotm} \cdot \text{hm} + \text{dmdalm} \cdot (\text{Al} - \text{bi}) \cdot \text{dbetdydot}; \\
\text{dmdxdotm} &= \text{dmdalm} \cdot \text{daldlamp} \cdot \text{dlampdxdot} - \text{dxdxdotm} \cdot \text{hm} + \text{dzdxdotm} \cdot \text{lm}; \\
\text{dmdgm} &= \text{dmdalm} \cdot \text{dalq} - \text{dxdgm} \cdot \text{hm}; \\
\text{dmdpm} &= \text{dmdalm} \cdot \text{daldp} - \text{dxdpm} \cdot \text{hm}; \\
\text{dmdthetom} &= \text{dmdalm} \cdot \text{daldtheto} - \text{dxdtheto} \cdot \text{hm} + \text{dzdthetom} \cdot \text{lm}; \\
\text{dmdalm} &= \text{dmdalm} \cdot \text{dald} - \text{dxdalm} \cdot \text{hm}; \\
\text{dmdblm} &= \text{dmdalm} \cdot \text{dald} - \text{dxdblm} \cdot \text{hm}; \\
\text{dndxdotm} &= \text{rho} \cdot \text{Ab} \cdot (\text{ohm} \cdot \text{R})^2 \cdot \text{R}^2 \cdot \text{dcqsigdmu} \cdot \text{dmudxdot} + \text{dcqsigdlamp} \cdot \text{dlampdxdot}; \\
\text{dndzdotm} &= \text{rho} \cdot \text{Ab} \cdot (\text{ohm} \cdot \text{R})^2 \cdot \text{R}^2 \cdot \text{dcqsigdlamp} \cdot \text{dlampdxdot}; \\
\text{dndrm} &= \text{-}2 / \text{ohm} \cdot \text{rho} \cdot \text{Ab} \cdot (\text{ohm} \cdot \text{R})^2 \cdot \text{R} \cdot \text{dcqsig}; \\
\text{dndthetom} &= \text{rho} \cdot \text{Ab} \cdot (\text{ohm} \cdot \text{R})^2 \cdot \text{R} \cdot \text{dcqsigdtheto}; \\
\% \\
\% \text{ return to CRUISE.M}
\end{align*}
\]
G. CTRGRP.M

% CTRGRP.M
% CALLED BY CRUISE.M
% Computes the stability derivatives of the tail rotor in cruise flight.
% Computes the stability derivatives of the NOTAR in cruise flight.
% Uses data loaded in the workspace by JANRAD.M and STAB.M.

if notar==0
  % FIX THIS
  Compute the stability derivatives of the tail rotor in cruise flight.
  dctsigdlampt=1.04 % table lookup, Prouty table 9.5
  dctsigdmut=-0.07 % table lookup, Prouty chap 3
  dctsiddthetot=659;
  dlampdydott=-1/((ohmt*Rt)*(1+dctsigdlampt*sigmat/(2*mut)));
  dydxdott=rho*Abt*(ohmt*Rt)^2*dctsigdmut*dmuxdott;
  dydxdott=rho*Abt*(ohmt*Rt)^2*dctsigdlampt*dlampdydott;
  dydpt=dydyptht;
  dydpt=dydypdctt;
  dydthetott=rho*Abt*(ohmt*Rt)^2*dctsiddthetott;
  drxdott=dydxdott*ht;
  drdxdott=dydxdott*ht;
  drdxdott=dydxdott*ht;
  drdxdott=dydxdott*ht;
  drdxdott=-dydxdott*lt;
  drdxdott=-dydxdott*lt;
  drdxdott=-dydxdott*lt;
  drdxdott=-dydxdott*lt;
  drdxdott=-dydxdott*lt;
  drdxdott=-dydxdott*lt;
  drdxdott=-dydxdott*lt;
  % If tail rotor is to be used, zero out NOTAR derivatives:
  dydxdotn=0;dydydotn=0;dydzdotn=0;dydhn=0;drdxdotn=0;
  drdydotn=0;drdzdotn=0;drdhn=0;drdxdotn=0;drdxdotn=0;
  ndndxdotn=0;ndndxdotn=0;ndndxdotn=0;
  ndndxdotn=0;drdzdotn=0;drdxdotn=0;drdxdotn=0;
  drdxdotn=0;drdxdotn=0;drdxdotn=0;
  drdxdotn=0;drdxdotn=0;drdxdotn=0;
  drdxdotn=0;drdxdotn=0;drdxdotn=0;
  drdxdotn=0;drdxdotn=0;drdxdotn=0;
  % Compute the stability derivatives of the NOTAR in cruise flight.
  qvin=.5*rho*vmax^2;
  dydxdotn=0;
  dydcsaar=qvin*bn^2*max(((-V/125)+1),0);
  % downwash blown off circulation control section above 125 ft/sec (75 kts)
  dcsoardcloar=1;
  dcloardpslot=2/pi;
  dpslotdcmu=dian/(2*bn)*sqrt(cmuh*dian/(2*hslot))*1.5;
  dcmudvmax=-64*hslot/(dian*vmax);
  dvmaxdxdotn=1;
  dcsaardswirl=-1/pi;
  dswirldydotn=1/v1;
dydydotn = dydcsaar*dcsoardswithl*dsvrirldydot;

dydsdotn = dydcsaar*dcsoardcloar*dcloardpslot*dspslotdcmu*dcmsudvimax*dvimax
dsdot;

dydpn = dydcsaar*dcsoardcloar*dcloardpslot*dspslotdcmu*dcmsudvimax*dvimax

dydxdotn = dydxdotn*hn;
dydrn = dydxdotn*ln;

% If NOTAR is to be used, zero out tail rotor derivatives:

dydxdott = 0; dyddydot = 0; dydpt = 0; dydrt = 0; dydthetot = 0; drdxdott = 0;

drdxdott = 0; drdpt = 0; drdrt = 0; drdthetot = 0; dndxdott = 0; dndydotn = 0;

ddyphin = Ytmax/(pi/2);

drphin = dydphin*htn;

% return to CRUISE.M
H. CBODYGRP.M

% CBODYGRP.M
% CALLED BY CRUISE.M
% Computes the stability derivatives of the fuselage, wing, vertical fin and horizontal stabilizer in cruise flight.
% Uses data loaded in the workspace by JANRAD.R4 STAB.M CRUISE.M
% CMRGRP.M CTRGRP.M
%
% THIS SUBROUTINE MUST FOLLOW CMRGRP.M AND CTRGRP.M
%
% Compute the stability derivatives of the fuselage

  dpmdfdxdotf= -1/V;
  dbetaydotf=1/V;
  detamfdxdotf= v/fv1/(4*q*pi*R^2)*(-dzdxdotm-2*T/V);
  detamfdxdotf= v/fv1/(4*q*pi*R^2)*dxzdotm;
  dalpfdxdotf= detamfdxdotf;
  dalpfdxdotf= (detamfdxdotf+dgamzdxdotf);
%
% the following are from the curves in appendix A of Prouty:
% and can be modified if necessary
  df/alpf=0.5;
  dl/alpf=19.2;
  dsf/dbetaf=-55;
  dms/dbetaf=445;
  dnd/dbetaf=-205;
  dr/dbetaf=58;
%
  dxzdotf=-(f*q*q*df/alpf)*dalpfdxdotf;
  dydotf=1/V*(q*dsf/dbetaf.*f*q);
  dzdxdotf=0;
  dzdxdotf=(f*q*q*dl/alpf)*dalpfdxdotf;
  dydotf=q*dr/dbetaf*dbetadydotf;
  mjd=xmjd-(detamfdxdotf+dgamzdxdotf);
  mjd=xmjd-(detamfdxdotf+dgamzdxdotf);
  nq=dbetadydotf;
  r=q*dr/dbetaf*dbetadydotf;
%
% Compute the stability derivatives of the wing

  if wing=1
    dxzdotw= -1/V;
    detamhdxdotw= v/vw1/(4*q*pi*R^2)*(-dzdxdotm-2*T/V);
    detamhdxdotw= v/vw1/(4*q*pi*R^2)*dzdxdotm;
    detafdxdotw=detafdalpfw/(1/(4*q*pi*R^2)*dzdxdotm-dgamzdxdotf);
    dalphdxdotw=dalh1dxdotw;
    xalpdxdotw=(detamhdxdotw+detafdxdotw+dgamzdxdotf);
    dalphhdxdotw=dalh1dxdotw+1/w/V;
%
    dxzdotw=2*cdoq;
  
  dxzdotw=q*Aw*aw*(1-2*aw*(1+deliw)*Aw/pi/bw^2)+alp-w-iw)*
    dalphhdxdotw;
  dxzdbldotw=dalh1dxdotw+1/w/V;
  dxdzdotw2/V2=2*q*Aw*aw*(1-aw*1+deliw)*Aw/pi/bw^2*(2*(alp-w-2)+cdoq)*dalh1dxdotw;
  dxdzdotw2/V2=2*q*Aw*aw*(1-aw*1+deliw)*Aw/pi/bw^2*(2*(alp-w-2)+cdoq)*dalh1dxdotw;
  dxdzdotw2/V2=2*q*Aw*aw*(1-aw*1+deliw)*Aw/pi/bw^2*(2*(alp-w-2)+cdoq)*dalh1dxdotw;
  dxdzdbldotw=dalh1dxdotw+1/w/V;
\%
dsdqw=dsdxdotw*1w;
dsdqw=dsdxdotw*1w;
dmdxdotw=-dxdxdotw*hv+dsdxdotw*1w;
dmdsdotw=-dxdxdotw*hv+dsdxdotw*1w;
dmdsdbldotw=-dxdxdotw*hv+dsdxdotw*1w;
dmdsdqw=dsdqw*1h;
\%
drdw=q*Aw*bw*bw*aw*alpw/4/V/V/Ixx;
drdpw=-q*Aw*bw*bw/4/V/V/Ixx*pi*(1+3*ctw/crw)/(6*(1+ctw/crw));
\% else
\% Zero stability derivatives of the wing when no wing is installed
\%
dxdxdotw=0;dxdxdotw=0;dsdxdotw=0;
dsdw=0;dsdxbldotw=0;dsdxdotw=0;
\% Compute the stability derivatives of the vertical fin
\%
detafdbeta=.06; \% assumed to be .06 because of little study of effect
detatvdxdotv=1/V;
detatvdxdotv=-1/(4*qvq*q*Av)*v(dyxdott-2*Tt/V);
dalphvdxdotv=detatvxdotv;
detatvdxdotv=dyxdott/(4*qvq*q*At);
detafdyuqdetatfdydotv;
dalphvdxdotv=-detatvxdotv+detatfdydotv;
\%
dxdtotv=0;
dydydotv=0;
dxdotv=2/V*Yv+qvq*q*Av*av*dalphvdxdotv;
dydydotv=1/(1-deldv/Vv)*...
\%
(dydpv=dydydotv*1v;
dydpv=dydydotv*1v;
dydydotv=dyxdott*1v;
dydydotv=dyxdott*1v;
dydydotv=dyxdott*1v;
dydydotv=dyxdott*1v;
\%
\% Compute the stability derivatives of the horizontal stabalizer
\%
dgamdzdoth=-1/V;
detahhdxdotv=vhv/(4*q*pi*R^2)*(-dzdxdotm-2*T*cos(altpy)/V);
detahhdxdotv=vhv/(4*q*pi*R^2)*dzdxdotm;
detahhdsdth=detafdalph*
(1/(4*pi*R^2)*dzdxdotm-dgamdzdoth);
dalphhdxdotv=-detahhdxdotv;
dalphhdxdotv=-abs(detahhdxdotv+detafhdzdoth+dgamdzdoth);
\%
\begin{verbatim}
dalphdzdldoth=-detamhdzdoth*1h/V;
\% etamh=vhv1*v1/V;  fix this
etahf=detafdalpfh*tho;
Xh=Lhoriz*sin(tho-(etamh+etafh+gamc)) -Dhoriz*cos(tho-(etafh+etafh+gamc));
dxdxdoth=2/V*Xh+qh*q*Ah*ah*((alph-alploh)*(1-2*ah*(1+delih)*Ah/pi/bh^2)
+alph-ih)*...
dalphdzdloth;
dxdzdoth=qh*q*Ah*ah*((alph-alploh)*(1-2*ah*(1+delih)*Ah/pi/bh^2)+alph-ih)*...
dalphdzdloth;
dxdzdloth=dxdzdloth*dalphdzdldoth/dalphdzdloth;
dxdxdoth=2/V*Zh-qh*q*Ah*ah*(1+ah*(1+delih)*Ah/pi/bh^2*(2*(alph-alploh)*
...}
\end{verbatim}
I. HOVER.M

% HOVER.M
% CALLED BY STABDER.M
% Computes the stability derivatives at a hover.
% calls the FOLLOWING subroutines to compute stability derivatives
% HMRGRP
% HTRGRP
% TRIM
%
% computation of stability derivatives
% the only derivatives important at hover are main and tail rotor
% format compact
% evaluate dctsiggdtheto dcqsigdtheto dctsiggdthetot and dcqsigdthetot

vctsig=zeros(1,2);
vcqsig=zeros(1,2);
vtetao=zeros(1,2);
vctsigt=ones(1,2);
vctsign=ones(1,2);
vctetao=ones(1,2);
eval(['load ',filenamel])
thetao=zeros(1,2);
GW=GW*1.0005; % .5% higher
disp('')
disp('')
*** FIRST OF TWO TRIM ITERATIONS ***

pause(3)
trim
*** Calculation of output parameters from trim ***
Qrotor=mean(DMpsib);
solidity=b*cblade/(pi*R);
CQ=Qrotor/(Adisk*rho*Vtip^2*R);
CH=Hrotor/(Adisk*rho*Vtip^2);
CQ_sig=CH/solidity;
vctsig(1,2)=CQ/solidity;
vcqsig(1,2)=CQ/solidity;
theta(save)=thetao;

147
blist=0;
theta75t=thetaot+.75*thetalt;
vlt=sqrt(Tt/2/rho/At);
aot=theta75t+atan(vlt/ohmt/Rt/.7);

cdt=.006+(aot*at)^2/pi/Rt/cot;
phit=sqrt(ctsigt*sigmat/2);

ther=4/ctsigt+sqrt(phit);
sigmat=sigmat/2*(at/2*(thet-phit)*phit+cdt/4);
end

vctsigt(1,2)=ctsigt;
vcqsigt(1,2)=cqsigt;

vthetaot(1,2)=thetaot;
save stabtemp vctsig vcqsig vthetao thetal
vctsig vctsig
vthetao

gcler
load stabtemp
eval(['load ',filenamel])
load stabtemp
disp('')
disp('')
disp('')
disp('')
disp('')
disp('')

*** SECOND OF TWO TRIM ITERATIONS ***

*** PERTURBATIONS COMPLETE ***

*** EVALUATING STABILITY DERIVATIVES ***

Qrotor=mean(DMpsi)*b;
solidity=b*cblade/(pi*R);
CQ=Qrotor/(Adisk*rho*Vtip^2*R);
CN=Hrotor/(Adisk*rho*Vtip^2);

vctsig(1,1)=CT/solidity;
vcqsig(1,1)=CQ/solidity;

vthetao(1,1)=vthetao;

vt=thetao-7*thetalt;
dctsigdthetao=(vctsig(1,1)-vctsig(1,2))/(vthetao(1,1)-vthetao(1,2));
dcqsigdthetao=(vcqsig(1,1)-vcqsig(1,2))/(vthetao(1,1)-vthetao(1,2));

omega=omega;
sigma=solidity;

A^R*R;
if notar=0
Abt=bt*Rt*cot;
At=pi*Rt*Rt;
mmt=Vinf/ohmt/Rt;
sigmat=Abt/At;
lt=ltd-xcg;
ht=htd-zcg;
yt=ytd-ycg;
locknot=rho*at*cot*Rt^4/Ibt;

Tt=Qrotor/lt;
sigmat=gt/(rho*lat*(ohmt*Rt)^2);
vlt=sqrt(Tt/2/rho/At);
lampt=vlt/ohmt/Rt;
aot=3*a*locknot*ctsigt-at-3/g*Rt^2/(ohmt*Rt)^2;
theta2/4-at*ctsigt+sqrt(CQ/2)) 3/4*thetalt;
alst=0;
blist=0;
theta75t=thetaot+.75*thetalt;
vlt=sqrt(Tt/2/rho/At);
aoat=theta75+atan(vlt/ohmt/Rt/0.7);
cdt=.006+(aoat*at)^2/pi/Rt/cot;
phi=sqrt(cqsig*sigmat/2);
theta4/at=sigmat+sqrt(phi);
cqsig=sigmat/2*(at/2*(theta-phi)+phi+cdt/4);
end
vctsigt(1,1)=ctsigt;
cqcsigt(1,1)=cqsigt;
vthetaelot(1,1)=thetaelot;
dctsigdthetot=(vctsigt(1,2)-vctsigt(1,1))/(vthetaelot(1,2)-vthetaelot(1,1));
dcqsigdthetot=(vcqsigt(1,2)-vcqsigt(1,1))/(vthetaelot(1,2)-vthetaelot(1,1));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CONFIGURATION CALCULATIONS
% converting data from crank
theta7=thetao;
thetao=theta7-.7*theta1;
ctsig=CT/solidity;
chsig=CH/solidity;
cqsig=CQ/solidity;
alphaT-
Al=-theta1c;
B1=-theta1s;
alpha=betao;
vi=mean(vi);
V=vinf;
% Main rotor
G=32.2;
hm=hmd-zcG;
ym=ymd-ycG;
lmlmld-xcG;
c=cblade;
Ab=c*R4;
lamp=0;
mu=0;
c=CT;
f=AfV;
theta1=4/a*ctsig+sqrt(sigma*ctsig/2);
lockno=rho*a*c*R4/Ib;
theta75=thetao+.75*theta1;
um1=thetao*(8/3+32/45*mu^3/pi);
num2=thetao*(2+mu^4/12);
num3=lamp*(2-mu^4/2);
den1=1+3/2*mu^5*mu^4/24;
al1=mu*(num1+num2+num3)/den1-B1;
bal1=als*12/lockno*e/R/(1+e/R/3);
mgW/g;
Ic=Ixx*IzZ;
Ic=IxX*IzZ;
if notarm=1;
% NOTAR
vmax=1.2*vi;
% downwash at the boom slots (NOTAR)
delvmax=pi/4;
ddelvddelp=delvmax/maxr;
ltn1=lntnd-xcG;
ln=lntnd-xcG;
yh=ytnd-xcG;
hn=htnd-xcG;
htn=hn;
Tc=cqsig*sigma*rho*A*(ohm*R)^2*R/ltn;
cmgm=55; % typical value
bn = 5*R;
halot = 0.028*dian/2;
arm = bn/dian;
end

trim conditions
tho = -T*(im+als)/GW;
pho = -(T*bis+Tc)/GW;
wo = 0;
vo = 0;
uo = 0;

% STABILITY CALCULATION

hmgrp
htgrp

% computation of A,B,C,D matrices

A11 = (dxdxdotm)/m;
A12 = (dxdxdotm)/m;
A13 = (dxdgm)/m-wo;
A14 = -g*cos(tho);
A15 = (dxdydotm)/m;
A16 = (dxdpn)/m;
A17 = 0;
A18 = (dxdm)/m+vo;

A21 = (dxdxdotm)/m;
A22 = (dxdxdotm)/m;
A23 = (dxdgm)/m+uo;
A24 = -g*cos(pho)*sin(tho);
A25 = (dxdydotm+dsdydott)/m;
A26 = (dxdpm)/m-vo;
A27 = -g*sin(pho)*cos(tho);
A28 = (dxdm)/m;

A31 = (dmdxdotm)/Iyy;
A32 = (dmdxdotm)/Iyy;
A33 = (dmdgm)/Iyy;
A34 = 0;
A35 = (dmdydott+dmdydott)/Iyy;
A36 = (dxdpm)/Iyy;
A37 = 0;
A38 = (dmdm+dmdm+dmdm)/Iyy;

A41 = 0;
A42 = 0;
A43 = cos(pho);
A44 = 0;
A45 = 0;
A46 = 0;
A47 = 0;
A48 = -sin(pho);

A51 = (dydxdotm)/m;
A52 = (dydxdotm+dydxdotm)/m;
A53 = (dydgm)/m;
A54 = -g*sin(pho)*sin(tho);
A55 = (dydxdotm+dydxdotm+dydxdotm)/m;
A56 = (dydpm+dydpm+dydpm)/m+wo;
A57 = \cos(\phi) \cdot \cos(\theta) \\
A58 = (dydx + dydt + dydr) / (m \cdot u_0) \\
A61 = (Izz \cdot (drdxdotm + drdxdotn) + Ixx \cdot (ddnxdotm + dndxdtotn)) / 1c \\
A62 = (Izz \cdot (drdxdotm + drdxdotn) + Ixx \cdot (ddnxdotm + dndxdtotn)) / 1c \\
A63 = (Izz \cdot (drdq) + Ixx \cdot (dndq)) / 1c \\
A64 = 0; \\
A65 = 6 \\
A66 = (Izz \cdot (drdxdotm + drdxdotn) + Ixx \cdot (ddnxdotm + dndxdtotn)) / 1c \\
A67 = 0; \\
A68 = (Izz \cdot (drdx + drdt + drdn) + Ixx \cdot (dndx + dndt + dndn)) / 1c \\
A71 = 0; \\
A72 = 0; \\
A73 = \sin(\phi) \cdot \tan(\theta) \\
A74 = 0; \\
A75 = 0; \\
A76 = 1; \\
A77 = 0; \\
A78 = \cos(\phi) \cdot \tan(\theta) \\
A81 = (Ixx \cdot (drdxdotm + drdxdotn) + Ixx \cdot (ddnxdotm + dndxdtotn)) / 1c \\
A82 = (Ixx \cdot (drdxdotm + drdxdotn) + Ixx \cdot (ddnxdotm + dndxdtotn)) / 1c \\
A83 = (Ixx \cdot (drdq) + Ixx \cdot (dndq)) / 1c \\
A84 = 0; \\
A85 = 8 \\
A86 = (Ixx \cdot (drdxdotm + drdxdotn) + Ixx \cdot (ddnxdotm + dndxdtotn)) / 1c \\
A87 = 0; \\
A88 = (Ixx \cdot (drdx + drdt + drdn) + Ixx \cdot (dndx + dndt + dndn)) / 1c \\

% longitudinal plant augmented \textbf{X}=[u w q \theta] \\
Plonaug=[A11 A12 A13 A14; A21 A22 A23 A24; A31 A32 A33 A34; A41 A42 A43 A44]; \\
Plonaug=poly(Plonaug); \\
Rlonaug=roots(Plonaug); \\

% Lateral plant augmented with \textbf{X}=[v p \phi r \psi]’ \\
Plataug=[A55 A56 A57 A58 0; A65 A66 A67 A68 0; A75 A76 A77 A78 0; A85 A86 A87 A88 0; 0 0 0 1 0]; \\
Plataug=poly(Plataug); \\
Rlataug=roots(Plataug); \\

% coupled plant \\
Amat=[A11 A12 A13 A14 A15 A16 A17 A18 0; A21 A22 A23 A24 A25 A26 A27 A28 0; A31 A32 A33 A34 A35 A36 A37 A38 0; A41 A42 A43 A44 A45 A46 A47 A48 0; A51 A52 A53 A54 A55 A56 A57 A58 0; A61 A62 A63 A64 A65 A66 A67 A68 0; A71 A72 A73 A74 A75 A76 A77 A78 0; A81 A82 A83 A84 A85 A86 A87 A88 0; 0 0 0 0 0 0 0 1 0]; \\
Pcoup=poly(Amat); \\
Pcoup=roots(Pcoup); \\
B11 = dxdb1m * dblmdddele/m; \\
B12 = dxdtheom * dthetommddelc/m; \\
B13 = dxdalm * dalmdddele/m; \\
B14 = 0;
\texttt{\%
B21 = d\text{d}b1m*d\text{d}b1mddele/m;
B22 = d\text{d}thetom*d\text{d}thetommddelc/m;
B23 = 0;
B24 = 0;
\%
B31 = d\text{m}b1m*d\text{m}b1mddele/Iyy;
B32 = d\text{m}thetom*d\text{m}thetommddelc/Iyy;
B33 = d\text{m}dalm*d\text{m}dalmmdela/Iyy;
B34 = (d\text{m}dphin*d\text{p}hinddelp+d\text{m}dthetot*d\text{th}etotddelp)/Iyy;
\%
B41 = 0;
B42 = 0;
B43 = 0;
B44 = 0;
\%
B51 = d\text{y}db1m*d\text{y}b1mddele/m;
B52 = d\text{y}thetom*d\text{y}thetommddelc/m;
B53 = d\text{y}dalm*d\text{y}dalmmdela/m;
B54 = (d\text{y}dphin*d\text{p}hinddelp+d\text{y}dthetot*d\text{th}etotddelp)/m;
\%
B61 = Izz*d\text{r}db1m*d\text{r}b1mddele/Ic;
B62 = (Izz*d\text{r}thetom*d\text{r}thetommddelc+Ixx*d\text{d}n\text{d}thetom*d\text{d}thetommddelc)/Ic;
B63 = Izz*d\text{r}dalm*d\text{r}dalmmdela/Ic;
B64 = (Izz*(drdphin*dphinddelp+drdthetot*dthetotddelp)+... 
Ixx*(d\text{d}n\text{d}phin*dphinddelp+d\text{d}n\text{d}thetot*dthetotddelp))/Ic;
\%
B71 = 0;
B72 = 0;
B73 = 0;
B74 = 0;
\%
B81 = Ixz*d\text{d}rdb1m*d\text{d}r1mddele/Ic;
B82 = (Ixz*d\text{d}rthetom*d\text{d}rthetommddelc+Ixx*d\text{d}n\text{d}thetom*d\text{d}thetommddelc)/Ic;
B83 = Ixz*d\text{d}r1m*d\text{d}r1mmdela/Ic;
B84 = (Ixz*(drdphin*dphinddelp+drdthetot*dthetotddelp)+... 
Ixx*(d\text{d}n\text{d}phin*dphinddelp+d\text{d}n\text{d}thetot*dthetotddelp))/Ic;
\%
Glonaug=[B11 B12 B13 B14;
B21 B22 B23 B24;
B31 B32 B33 B34;
B41 B42 B43 B44];
\%
Glataug=[B51 B52 B53 B54;
B61 B62 B63 B64;
B71 B72 B73 B74;
B81 B82 B83 B84;
0 0 0 0];
\%
% coupled input matrix
Bmat=[B11 B12 B13 B14;B21 B22 B23 B24;
B31 B32 B33 B34;B41 B42 B43 B44;
B51 B52 B53 B54;B61 B62 B63 B64;
B71 B72 B73 B74;B81 B82 B83 B84;0 0 0 0];
\%
% cross coupling
xcouple=12/lockno*e/R/(1+e/3/R);
% designed damping
desdmdg=dmdgm;
desdsrc=drdp+sdrdu+drdpn;
desdsrc=drdr+drd+drdrn;()}
% now cooper harper pilot rating
prpitch=desmdq/Iyy;
prroll=(drdpm+drdpt+drdpn)/Ixx;
pryaw=desndr/Izz;
% control power
cppitch=B31*Tyy;
cproll=B63*Txx;
cpyaw=B84*Tzz;
cpipitch=B31;
cpiroll=B63;
cpiyaw=B84;
% thetasc=theta7;
J. HMRGRP.M

% HMRGRP.M
% CALLED BY HOVER.M
% Computes the basic main rotor derivatives at a hover
% Computes the stability derivatives of the main rotor at a hover
% Uses data loaded in the workspace by JANRAD.M and STAB.M
%
% Compute the basic main rotor derivatives at a hover
% dmudxdot=1/ohm/R;
dlampdzdot=dmudxdot;
dalimu=8/3*thetao+2*theta1-2*v1/ohm/R;
dbdmu=4/3*a0;
dctsgldlamp=inv(8/a+(sqrt(sigma/2))/(sqrt(ctsig)));
dcqsqglamp=-a/4*(theta75-2*v1/ohm/R);
daldq=-16/(lockno*ohm*(1-e/R)^2)*12*e/R/(lockno*ohm*(1-e/R)^3);
dbdq=bdalq;
dbgp=1/ohm*(1-192*e/R/(lockno*2*(1-e/R)^5));
dbdq=bdalp;
dalpa=12*e/R/(lockno*(1-e/R)^3);
dbdp=bdalp;
dalpa=1/((144*(e/R)^2)/(lockno^2*(1-e/R)^6));

dchsigdalpha=3/2*ctsig*(1-a/18*theta75/ctsig);
dcysigdb=dchsigda;
dmals=3/4*e/R*Ab*rho*R*(ohm*R)^2*a/lockno;
drdbls=dmals;
%
% Compute the main rotor stability derivatives at a hover
% dxddxdotm=-rho*Ab*(ohm*R)^2*dchsigda*dalimu*dmudxdot;
dxddydotm=-rho*Ab*(ohm*R)^2*dcysigdb*bdalmu*dmudxdot;
dxddotm=0;
dxddpm=-rho*Ab*(ohm*R)^2*dchsigda*dalq;
dxddypm=-rho*Ab*(ohm*R)^2*dcysigdb*bdalp;
dxdrm=0;
dxddthetom=-rho*Ab*(ohm*R)^2*(als+im)*dctsgdtheto;
dxddalm=-rho*Ab*(ohm*R)^2*dchsigda*dalda;
dxddblm=-rho*Ab*(ohm*R)^2*dchsigda*daldb;
dyddotm=rho*Ab*(ohm*R)^2*dcysigdb*bdalmu*dmudxdot;
dyddydotm=dxddxdotm;
dydd0tm=dyddpm;
dydrm=0;
dyddpm=-dyddpm;
dyddthetom=rho*Ab*(ohm*R)^2*(2*bis)*dctsgdtheto;
dyddalm=dydblm;
dyddblm=-dyddalm;
dzddotm=0;
dzddydotm=0;
dzddotm=rho*Ab*(ohm*R)^2*dctsgdlamp*dlampdzdot;
dzdpm=0;
dzdqm=0;
dzdrm=0;
dzddthetom=rho*Ab*(ohm*R)^2*dctsgdtheto;
dzdblm=0;
drdxdotm=drdbls*bdalmu*dmudxdot+dyddotm*hm+dzddotm*hm;
drdydotm=mdals*bdalmu*dmudxdot+dxddotm*hm;
drdzdotm=dzddotm*ym;
drdqm=drdcls*dbldq+dydq*m*hm;
    drdrm=0;
    drdpm=drdcls*dbldp+dydpm*hm;
    drdthetom=dydthetom*hm+drdthetom*ym;
    drdalm=drdcls*dblda+dydalm*hm;
    drdalm=drdcls*dblda+dydalm*hm;
    dmdxidotm=mdmals*daidsmu*dmdxidotm-dmdxidotm*hm;
    dmdxidotm=drxidotm;
    dmdxidotm=dsdxdotm*lm;
    dmdqm=mdmals*daldq-dxdqm*hm;
    dmdqm=0;
    dmddpm=mdmals*daldp-dxdpm*hm;
    dmdthetom=dsdthetom*hm+dsdthetom*lm;
    dmdalm=mdmals*dalda-dxdaalm*hm;
    dmdalm=mdmals*dalda-dxdaalm*hm;
    dmdxidotm=0;
    dmdxidotm=rho*Ab* (ohm R)^2*R*dcqsigdlamp*dlampdsdot;
    dmdxidotm=0;
    dmdqm=0;
    dmdpm=0;
    dndrm=-2*rho*Ab* (ohm R)^2*R*dcqsig;
    dndthetom=rho*Ab* (ohm R)^2*R*dcqsigdtheto;
    &
    & return to HOVER.M
K. HTRGRP.M

% HTRGRP.M % CALLED BY HOVER.M % Computes the basic tail rotor or NOTAR derivatives at a hover % Computes the stability derivatives of the tail rotor or NOTAR at a hover % Computes the stability derivatives of the NOTAR at a hover % Uses data loaded in the workspace by JANRAD.M and STAB.M

if notar=0

% Compute the basic tail rotor derivatives at a hover.
% dmudxdott=1/ohmt/Rt;
dlampdydott=-dmudxdott;
da1dmut=8/3*thetao+2*thetal-2*v1/ohmt/Rt;
db1dmut=4/3*aot;
dctsigdlampt=inv(8/3*(sqrt(sigmat/2)/(sqrt(ctsig))));
dcqsigdlampt=-at/4*(theta75t-2*vlt/ohmt/Rt);
daldqt=0;
db1dpt=0;
da1dpt=0;
db1dqt=0;
da1dat=0;
db1dat=0;
dcdsigdat=3/2*ctsig*(l-at/18*theta75t/ctsig);
dcysigdbt=dchsigdat;
dmdal1st=0;
dmdb1st=0;

% Compute the tail rotor stability derivatives at a hover
% dydydott=rho*Abt*(ohmt*Rt)^2*dctsigdlampt*dlampdydott;
dydydott=dydydott*ht;
dydydott=-dydydott*lt;
dydydott=rho*Abt*(ohmt*Rt)^2*dctsigdthetat;
drdrdyydott=dydydott*ht;
drdrdyydott=dydydott*ht;
drdrdyydott=rho*Abt*(ohmt*Rt)^2*dcqsigdlampt*dlampdydott;
drdrdyydott=-drdydott*lt;

% If tail rotor is used, zero out NOTAR derivatives:
% dydxdotn=0;dydydotn=0;dydzdotn=0;dydpn=0;dydrn=0;
drzdotn=0;
drdxdotn=0;drdydotn=0;drdzdotn=0;drdpn=0;drdrn=0;
dmdydotn=0;mdrdrn=0;
dmdynphn=0;mdrdrphin=0;mdrddphin=0;mdyddphin=0;
else if notar==1

    % Compute the NOTAR derivatives at a hover.
    %
    qvin=.5*rho*vlmax^2;
    dyxdotn=0;
    dydcsoar=qvin*bn^2;
    dcsoardcloar=1;
    dcloardpslot=2/pi;
    dpslotdcmu=dian/(2*bn)*sqrt(cmun*dian/(2*hslot))*1.5;
    dcmudvimax=-64*hslot/(dian*vlmax);
    dvimaxdzdot=1;
    dcsoardswirl=-1/pi;
    dwirldydot=1/v1;
    dydydotn=dydcsoar*dcsoardswirl*dwirldydot;

dydzdotn=dydcsoar*dcsoardcloar*dcloardpslot*dpslotdcmu*dcmudvimax*dvimax
dsdot;
    dydydotn=dydydotn*hn;
    dydrn=-dydydotn*ln;
    drxdotn=dyxdotn*hn;
    drdydotn=dydydotn*hn;
    drdzdotn=dydzdotn*hn;
    drdpn=dydpn*hn;
    drdrn=dydrn*hn;
    dmdydotn=0;
    dmdrn=-dmdydotn*ln;
    ndxdotn=-dyxdotn*ln;
    nddydotn=-dydydotn*ln;
    nddzdotn=-dydzdotn*ln;
    nddpn=-dydpn*ln;
    nddrn=-dydrn*ln;
    ddpin=Ytmaxn/(pi/2);
    drdpin=dydpin*htn;
    dmdpin=0;
    nddpin=-dydpin*ln;

    % If NOTAR is to be used, zero out tail rotor derivatives:
    %
    dydydott=0;dydpt=0;dydrt=0;dydthetot=0;drdydott=0;
    dsdydott=0;
    drdrt=0;drdpt=0;drdthetot=0;
    dmdydott=0;dmdrt=0;dmdthetot=0;
    nddydott=0;ndrdt=0;ndrthetot=0;
    lampt=0;mut=0;not=0;blst=0;alst=0;locknot=0;
    ndrthetot=0;nddthetot=0;drdthetot=0;dmdthetot=0;
else
    disp('')
    disp('')
    disp('ERROR IN HTRGRP.M')
    disp('')
    disp('NO TAIL ROTOR OR NOTAR INSTALLED')
    disp('')
    disp('CHECK INPUT DATA')
    disp('')
    disp('Press any key to continue...')
    pause
end

% return to HOVER.M
L. STABOUT.R

% STABOUT.R
% calls
% CMDBWPLH
% CMDBWPLC
% *** Stability and Control Output Subroutine ***
pack
format compact
clc
disp(' ')
disp(' Do you want the results displayed on screen?')
disp(' ')
disp(' NOTE: if you want a hard copy of the plots, you must')
disp(' select (1) and view them on the screen first.')
flag=1;
answer=input('1. yes 2. no >> ');
while flag>0
    if answer==2,
        flag=0;
    elseif answer==1,
        % output to screen
        clc
disp(' ')
disp(' *** STABILITY AND CONTROL PROGRAM ***')
disp(' *** SCREEN VIEW MENU ***')
disp(' ')
disp(' What do you want to see?')
disp(' ')
disp(' 1. Input data.')
disp(' 2. Calculated data.')
disp(' 3. State Matrices.')
disp(' 4. Eigenvalues of the plants and plots of the roots.')
disp(' 5. Key control parameters.')
disp(' 6. Open loop transfer plots.')
disp(' ')
disp(' 0. Exit screen view.')
disp(' ')
    choice=input('Enter a number: ');
    if choice==1,
        clc
disp(' ')
disp(' *** INPUT DATA (screen 1 of 8) ***')
eval(['disp(' ',filename1,''),'])
disp(' ')
disp(' ')
end
fprintf(' Forward velocity = %6.0f kts\n',Vinf/1.69)
fprintf(' Temperature = %6.0f degs F\n',temp)
fprintf(' Pressure altitude = %6.0f ft\n',PA)
fprintf(' Auxiliary thrust = %6.0f lbs\n',Taux)
disp(' ')
disp(' Fuselage')
disp(' ')
fprintf(' Gross weight = %6.0f lbs\n',GW)
158
fprintf(' Equivalent flat plate area = %6.1f ft^2
',Afl)
fprintf(' Vertical projected area = %6.1f ft^2
',Afv)
fprintf(' CG height above waterline = %6.1f ft
',zcg)
fprintf(' CG fuselage station = %6.1f ft
',xcg)
fprintf(' CG position right of buttline = %6.1f ft
',ycg)
fprintf(' lxx M V6.lf slug ft^2
',Ixx)
fprintf(' Iyy = W6.1f slug ft^2
',Iyy)
fprintf(' Izz = W6.lf slug ft^2
',Izz)
fprintf(' Ixz = W6.1f slug ft^2
',Ixz)
fprintf(' Downwash ratio = %6.2f
',vfvl)
disp(' press any key to continue...')
 pause
clc
disp(' *** INPUT DATA CONTINUED (screen 2 of 8) ***')
eval(['disp('',filename1,'''P)'

disp(' Main Rotor')
disp(' Number of blades = %6.0f

fprintf(' Rotor radius = %6.1f ft

fprintf(' Blade chord = %6.1f ft

fprintf(' Blade twist = %6.2f degs

if airfoil==1,
 disp(' Blade airfoil = HH-02')
else
 disp(' Blade airfoil = VR-12')
end
fprintf(' Blade lift curve slope = %6.2f

fprintf(' Blade weight = %6.1f lbs

fprintf(' Rotational velocity = %6.2f rads/sec

fprintf(' Blade grip length = %6.1f ft

fprintf(' Hinge offset = %6.1f ft

fprintf(' Flapping moment of inertia = %6.1f slug ft^2

fprintf(' Hub height above waterline = %6.1f ft

fprintf(' Hub fuselage station = %6.1f ft

fprintf(' Hub position rt of buttline = %6.1f ft

fprintf(' Mast incidence = %6.2f deg

disp(' press any key to continue...')
 pause
clc
disp(' *** INPUT DATA CONTINUED (screen 3 of 8) ***')
eval(['disp('',filename1,'''P)'

disp(' Tail rotor (zeros if using NOTAR')
disp(' Number of blades = %6.1f

fprintf(' Blade chord = %6.1f ft

fprintf(' Blade radius = %6.1f ft

fprintf(' Lift curve slope = %6.2f

fprintf(' Rotational velocity = %6.2f rads/sec

fprintf(' Flapping moment of inertia = %6.1f slug ft^2

fprintf(' Delta-3 angle = %6.2f deg

fprintf(' Blade twist = %6.2f deg

fprintf(' Hub height above waterline = %6.1f ft

fprintf(' Hub fuselage station = %6.1f ft

fprintf('...')
 pause
clc
fprintf(' Hub position rt of buttock line = %6.1f ft\n',ytd)
disp(' ')
disp(' press any key to continue...')
pause
clc
disp(' ')
disp(' *** INPUT DATA CONTINUED (screen 4 of 8) ***')
eval(['disp(''
',filename1,'''')'
])
disp(' ')
disp(' NOTAR (zeros if using tail rotor)')
disp(' ')
fprintf(' Height above waterline = %6.1f ft\n',htnd)
fprintf(' Fuselage station = %6.1f ft\n',lftnd)
fprintf(' Position right of buttock line = %6.1f ft\n',ytdn)
fprintf(' NOTAR boom diameter = %6.1f ft\n',dian)
fprintf(' Swirl angle at boom = %6.2f deg\n',swirl*57.3)
fprintf(' Maximum thruster force = %6.1f lbs\n',Ytmaxn)
fprintf(' Thrust fuselage station = %6.1f ft\n',ltfndn)
disp(' ')
disp(' press any key to continue...')
pause
clc
disp(' ')
disp(' *** INPUT DATA CONTINUED (screen 5 of 8) ***')
eval(['disp(''
',filename1,'''')'
])
disp(' ')
disp(' Wing')
disp(' ')
fprintf(' Area = %6.1f ft^2\n',Swing)
fprintf(' Span = %6.1f ft\n',bwing)
fprintf(' CL = %6.2f \n',CLwing)
fprintf(' CDo = %6.4f \n',CDowing)
fprintf(' Tip cord = %6.1f ft\n',ctw)
fprintf(' Root cord = %6.1f ft\n',crw)
fprintf(' Wing efficiency factor = %6.2f \n',ewing)
fprintf(' Zero lift angle = %6.2f deg\n',alplow*57.3)
fprintf(' Angle of incidence = %6.2f deg\n',iw*57.3)
fprintf(' Lift curve slope = %6.2f \n',aw)
fprintf(' Height above waterline = %6.1f ft\n',hwd)
fprintf(' Fuselage station = %6.1f ft\n',lwd)
fprintf(' Position right of buttock line = %6.1f ft\n',ywd)
fprintf(' Rotor downwash ratio = %6.2f \n',vvvl)
fprintf(' Fuselage downwash ratio = 6.2f \n',detafdalpfw)
disp(' ')
disp(' press any key to continue...')
pause
clc
disp(' ')
disp(' *** INPUT DATA CONTINUED (screen 6 of 8) ***')
eval(['disp(''
',filename1,'''')'
])
disp(' ')
disp(' Horizontal tail')
disp(' ')
fprintf(' Area = %6.1f ft^2\n',Shoriz)
fprintf(' Span = %6.1f ft\n',bhoriz)
fprintf(' CL = %6.2f \n',CLhoriz)
fprintf(' CDo = %6.4f \n',CDohoriz)
fprintf(' Zero lift angle = %6.2f deg\n',alploh*57.3)
fprintf(' Angle of incidence = %6.2f deg\n',iw*57.3)
fprintf(' Lift curve slope = %6.1f \n',ah)
fprintf(' Height above waterline = \%6.1f ft\n',hhd)
fprintf(' Fuselage station = \%6.1f ft\n',lhd)
fprintf(' Position right of buttline = \%6.1f ft\n',yhd)
fprintf(' Dynamic pressure ratio = \%6.2f \n',qhq)
fprintf(' Rotor downwash ratio = \%6.2f \n',vhvl)
fprintf(' Fuselage downwash ratio = \%6.2f \n',detafdalpfh)
disp('\n')
disp('press any key to continue...')
pause
clc
disp(' *** INPUT DATA CONTINUED (screen 7 of 8) ***')
eval(['disp('' ',filenamel,'''')'])
disp('Vertical tail')
disp('')
fprintf(' Area = \%6.1f ft\^2\n',svert)
fprintf(' Span = \%6.1f ft\n',bvert)
fprintf(' CL = \%6.2f \n',clvert)
fprintf(' CDo = \%6.4f \n',cd0vert)
fprintf(' Height above waterline = \%6.1f ft\n',hvd)
fprintf(' Fuselage station = \%6.1f ft\n',lvd)
fprintf(' Position right of buttline = \%6.1f ft\n',yvd)
fprintf(' Zero lift angle = \%6.2f deg\n',alplov*57.3)
fprintf(' Maximum Cl = \%6.1f \n',clvertmax)
fprintf(' Lift curve slope = \%6.2f \n',av)
disp('press any key to continue...')
pause
clc
disp(' *** INPUT DATA CONTINUED (screen 8 of 8) ***')
eval(['disp('' ',filenamel,'''')'])
disp('Rigging')
disp('')
fprintf(' Long cyclic pitch/inch defl = \%6.2f deg/in\n',dblmddele*57.3)
fprintf(' Lat cyclic pitch/inch defl = \%6.2f deg/in\n',dalmddela*57.3)
fprintf(' Collective pitch/inch defl = \%6.2f deg/in\n',dthetomddelc*57.3)
fprintf(' Tail rotor pitch change/defl = \%6.2f deg/unit\n',dthetotddelp*57.3)
fprintf(' Max deflection of control')
disp('Max deflection of control')
fprintf(' from neutral for NOTAR = \%6.2f units\n',dphinddelp*57.3)
fprintf(' Displacement of anti-torque')
disp('Displacement of anti-torque')
fprintf(' control until full rudder = \%6.2f units\n',maxr)
disp('press any key to continue...')
pause
elseif choice==2,
clc
disp(' *** CALCULATED DATA (screen 1 of 2) ***')
eval(['disp('' ',filenamel,'''')'])
disp('Main Rotor')
disp('')
fprintf(' thrust = \%6.1f lbs\n',T)
fprintf(' torque = \%6.1f ft-lbs\n',Qrotor)
fprintf(' advance ratio = %6.1f \n',mu)
fprintf(' inflow parameter wrt TPP = %6.3f \n',lamp)
fprintf(' Tip path angle = %6.1f degs\n',altpp*57.3)
fprintf(' Rotor coning angle = %6.1f degs\n',ao*57.3)
fprintf(' 1st lat cyclic term=Al = %6.1f degs\n',Al*57.3)
fprintf(' 1st long cyclic term=Bl = %6.1f degs\n',Bl*57.3)
fprintf(' lateral flapping = %6.2f degs\n',bls*57.3)
fprintf(' longitudinal flapping = %6.2f degs\n',als*57.3)
fprintf(' Lock number = %6.1f \n',lockno)
disp('')
disp('press any key to continue...')
pause
clc
disp('')
 *** CALCULATED DATA (screen 2 of 2)***
eval(['disp('' ',filenamel,''')

disp('')
disp(' Tail Rotor (zero if NOTAR)'

disp('')
fprintf(' tail rotor thrust = %6.1f lbs\n',Tt)
fprintf(' advance ratio = %6.1f \n',mut)
fprintf(' inflow parameter = %6.3f \n',lampt)
fprintf(' Rotor coning angle = %6.1f degs\n',aot*57.3)
fprintf(' lateral flapping = %6.2f degs\n',blst*57.3)
fprintf(' longitudinal flapping = %6.2f degs\n',alst*57.3)
fprintf(' Lock number = %6.1f \n',locknot)
disp('')
disp('')
disp('press any key to continue...')
pause
 elseif choice==3, % state matrices
clc
disp('')
disp('Longitudinal uncoupled plant (A or F depending on notation)')
disp('States are [u w q theta]')
disp('')
disp(Flonaug)
disp('')
disp('Longitudinal uncoupled input matrix (B or G depending on notation)')
disp('Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]')
disp('')
disp(Glonaug)
disp('')
disp('press any key to continue...')
pause
clc
disp('')
disp('Lateral/directional uncoupled plant (A or F depending on notation)')
disp('States are [v p phi r psi]')
disp('')
disp(Flataug)
disp('')
disp('Lateral/directional uncoupled input matrix (B or G depending on notation)')
disp('Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]')
disp(Glataug)

162
disp(' ')
disp('press any key to continue...')
pause
clc
disp(' ')
disp('Coupled plant (A or F depending on notation)')
disp('States are [u w q theta v p phi r psi]')
disp(' ')
disp(Amat)
disp('press any key to continue...')
pause
clc
disp(' ')
disp('Coupled input matrix (B or G depending on notation)')
disp('Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]')
disp(' ')
disp(Bmat)
disp(' ')
disp('press any key to continue...')
pause
clc
elseif choice==4,
Veigenvalues and root loci
clc
disp(' ')
disp('After you view the root loci plot, a meta file is made.')
disp('When you are done a screen will tell you the file names')
disp('of the meta files. To get a hard copy of the plots, you')
disp('must graphics post process (GPP) the files for your')
disp('particular printer set-up then, print.')
disp(' ')
disp('NOTE: If ALL roots are real, MATLAB will NOT plot them')
disp('in the A-o- and plane, but will plot the root against')
disp('its position in the vector (e.g. the first root would')
disp('be plotted as (1,root)')
disp(' ')
disp('press any key to continue...')
pause
clc
disp(' ')
*** EIGENVALUES ***

eval(['disp('' ',filenamel,'''')'])
disp(' ')
disp('Uncoupled')
disp(' ')
disp('Longitudinal plant')
disp(' ')
disp('Root wn damping')
[wn,z]=damp(Rlonaug);
disp([Rlonaug,wn,z])
disp(' ')
disp('Lateral/Directional plant')
[wn,z]=damp(Rlataug);
disp(' ')
disp('Root wn damping')
[wn,z]=damp(Rlataug);
disp(' ')
disp('press any key to continue...')
pause
plot(Rlonaug,'*'),grid,title('Roots of Longitudinal Plant')
pause
!del rootlon.*
meta rootlon
plot(Rlatug,'*'),grid,title('Roots of Lateral/Directional Plant')
pause
!del rootlat.*
meta rootlat
clc
disp(' *** EIGENVALUES ***')
eval(['disp('',filenamel,'')'])
disp(' Coupled Plant')
disp(' ')
[wn,z]=damp(Rcoup);
disp(' Root wn damping')
disp([Rcoup wn z])
disp(' press any key to continue...')
pause
plot(Rcoup,'*'),grid,title('Roots of Coupled Plant')
pause
!del rootcoup.*
meta rootcoup
clc
disp(' Plots are saved under the following filenames:')
disp(' Longitudinal roots - rootlon.met')
disp(' Lateral/Directional roots - rootlat.met')
disp(' Coupled roots - rootcoup.met')
disp(' press any key to continue...')
pause
elseif choice==5, % key control parameters
clc
disp(' *** KEY CONTROL PARAMETERS (screen 1 of 2) ***')
eval(['disp('',filenamel,'')'])
disp(' ')
fprintf(' cross coupling = %.2f \n',xcouple)
disp(' ')
disp(' Designed damping')
fprintf(' pitch = %.1f ft-lbs/(rad/sec)\n',deadmag)
fprintf(' roll = %.1f ft-lbs/(rad/sec)\n',deadrp)
fprintf(' yaw = %.1f ft-lbs/(rad/sec)\n',deadndr)
disp(' ')
disp(' Control Power')
fprintf(' pitch = %.1f ft-lbs/in\n',cppitch)
fprintf(' roll = %.1f ft-lbs/in\n',cpproll)
fprintf(' yaw = %.1f ft-lbs/in\n',cppyaw)
disp(' ')
disp(' press any key to continue...')
pause
clc
disp(' *** KEY CONTROL PARAMETERS (screen 2 of 2) ***')
eval(['disp('',filenamel,'')'])
disp(' ')
disp(' Cooper Harper Pilot Ratings')
disp(' damping/moment of inertia')
fprintf('pitch (dM/dq)/Iyy = %.2f [ft-lbs/(rad/sec)]/(slug ft^2/n',prpitch)
fprintf('roll (dR/dp)/Ixx = %.2f [ft-lbs/(rad/sec)]/(slug ft^2/n',prroll)
fprintf('yaw (dN/dr)/Izz = %.2f [ft-lbs/(rad/sec)]/(slug ft^2/n',pryaw)
disp('')
disp(' control power/moment of inertia')
fprintf('pitch (d1/dn)/Iyy = %.2f (ft-lbs/in)/(slug ft^2/n',cpipitch)
fprintf('roll (dR/in)/Ixx = %.2f (ft-lbs/in)/(slug ft^2/n',cpiroll)
fprintf('yaw (dI/in)/Izz = %.2f (ft-lbs/in)/(slug ft^2/n',cpiyaw)
disp('')
disp('')
disp('press any key to continue...')
pause

elseif choices==6, % command bandwidth plots
clc
disp('')
disp('After you view a bode plot of the transfer function from')
disp('input to state output, a meta file is made. When you exit,')
disp('a screen will tell you the file names of the meta files.')
disp('To get a hard copy of the plots, you must graphics post')
disp('process (GPP) the files for your particular printer set-up')
disp('then print.')
disp('')
if Vinf<20
    cmdbwplh
else,
    cmdbwplc
end
elseif choice==0,
    flag=0;
else
    disp('')
disp('Enter a number on the menu')
pause(3)
end
end
end

% *** output to disk (text file) ***
% diary on
diary off
delete diary
diary on
disp('')
eval(['disp(''
     ** RESULTS **')
     ',filename,'''')]')
disp('')
disp('')
disp('')
disp('')
disp('')

*** INPUT DATA ***

Flight Conditions'

forward velocity = %.0f kts\n',Vinf/1.69)

Temperature = %.0f degs F\n',temp)

Pressure altitude = %.0f ft\n',PA)

Auxiliary thrust = %.0f lbs\n',Taux)

Fuselage')
disp('')
fprintf(' Gross weight = %6.0f lbs\n',GW)
fprintf(' Equivalent flat plate area = %6.1f ft^2\n',Af)
fprintf(' Vertical projected area = %6.1f ft^2\n',Afv)
fprintf(' CG height above waterline = %6.1f ft\n',zcg)
fprintf(' CG fuselage station = %6.1f ft\n',xcm)
fprintf(' CG position rt of buttline = %6.1f ft\n',ycg)
fprintf(' Ixx = %6.1f slug ft^2\n',Ixx)
fprintf(' Iyy = %6.1f slug ft^2\n',Iyy)
fprintf(' Izz = %6.1f slug ft^2\n',Izz)
fprintf(' Ixz = %6.1f slug ft^2\n',Ixz)
fprintf(' Downwash ratio = %6.2f \n',vfvl)

disp('')
disp('Main Rotor')
fprintf(' Number of blades = %6.0f \n',b)
fprintf(' Rotor radius = %6.1f ft\n',R)
fprintf(' Blade chord = %6.1f ft\n',cblade)
fprintf(' Blade twist = %6.2f degs\n',twist*57.3)
if airfoil==l,
    disp(' Blade airfoil = HH-02')
else
    disp(' Blade airfoil = VR-12')
end
fprintf(' Blade lift curve slope = %6.2f \n',a)
fprintf(' Blade weight = %6.1f lbs\n',wblade)
fprintf(' Rotational velocity = %6.2f rads/sec\n',omega)
fprintf(' Flapping moment of inertia = %6.1f slug ft\n',Ib)
fprintf(' Hub height above waterline = %6.1f ft\n',hmd)
fprintf(' Hub fuselage station = %6.1f ft\n',lmd)
fprintf(' Hub position rt of buttline = %6.1f ft\n',ymd)

disp('')
if notar==0
    disp(' Tail rotor (zero if NOTAR)')
    fprintf(' Number of blades = %6.1f \n',bt);
    fprintf(' Blade chord = %6.1f ft\n',cbladet);
    fprintf(' Rotor radius = %6.1f ft\n',Rt);
    fprintf(' Lift curve slope = %6.2f \n',at);
    fprintf(' Rotational velocity = %6.2f rads/sec\n',omega2);
    fprintf(' Flapping moment of inertia = %6.1f slug ft\n',Ib2);
    fprintf(' Hub height above waterline = %6.1f ft\n',hmdt);
    fprintf(' Hub fuselage station = %6.1f ft\n',lmdt);
    fprintf(' Hub position rt of buttline = %6.1f ft\n',ymdt);
    disp('')
elseif notar==l
    disp(' NOTAR')
    fprintf(' Height above waterline = %6.1f ft\n',htnd)
    fprintf(' Fuselage station = %6.1f ft\n',ltd)
    fprintf(' Position right of buttline = %6.1f ft\n',ytnd)
    fprintf(' NOTAR boom diameter = %6.1f ft\n',dian)
    fprintf(' Swirl angle at boom = %6.2f deg\n',swirl*57.3)
    fprintf(' Maximum thruster force = %6.1f lbs\n',Ytmax)
    fprintf(' Thrust fuselage station = %6.1f ft\n',lttd)
end

166
disp(' ')
end

disp(' Wing')

disp(' ')
fprintf(' Area = %6.1f ft^2
',Swing)
fprintf(' Span = %6.1f ft
',bwing)
fprintf(' CL = %6.2f \n',CLwing)
fprintf(' CDo = %6.4f \n',C Downing)
fprintf(' Tip cord = %6.1f ft\n',ctw)
fprintf(' Root cord = %6.1f ft\n',crw)
fprintf(' Wing efficiency factor = %6.2f \n',ewing)
fprintf(' Zero lift angle = %6.2f deg\n',lalplov*57.3)
fprintf(' Angle of incidence = %6.2f deg\n',iw*57.3)
fprintf(' Lift curve slope = %6.2f \n',av)
fprintf(' Height above waterline = %6.1f ft\n',hwd)
fprintf(' Fuselage station = %6.1f ft\n',lwd)
fprintf(' Position right of buttline = %6.1f ft\n',ywd)
fprintf(' Rotor downwash ratio = %6.2f \n',vwvl)
fprintf(' Fuselage downwash ratio = %6.2f \n',detafdalpfw)

disp(' ')

disp(' Horizontal tail')

disp(' ')

disp(' Vertical tail')

disp(' ')

disp(' Rigging')

disp(' ')
fprintf(' Long cyclic pitch/inch defl = %6.2f deg/in\n',dblede*57.3)
fprintf(' Lat cyclic pitch/inch defl = %6.2f deg/in\n',dalmde*57.3)
fprintf(' Collective pitch/inch defl = %6.2f deg/in\n',dtheta*57.3)
fprintf(' Tail rotor pitch change/defl = %6.2f deg/unit\n',d theta*57.3)
fprintf(' Max deflection of control = %6.2f units\n',dphind*57.3)
disp('Displacement of anti-torque')
fprintf('control until full rudder = %6.2f units\n',maxr)
disp('*** CALCULATED DATA ***')
disp('Main Rotor')
fprintf('thrust = %6.1f lbs\n',T)
fprintf('torque = %6.1f ft-lbs\n',Qrotor)
fprintf('advance ratio = %6.1f \n',mu)
fprintf('inflow parameter wrt TPP = %6.3f \n',lamp)
fprintf('Tip path angle = %6.1f degs\n',altpp*57.3)
fprintf('Rotor coning angle = %6.1f degs\n',ao*57.3)
fprintf('1st lat cyclic term-A1 = %6.2f degs\n',Al*57.3)
fprintf('1st long cyclic term-B1 = %6.2f degs\n',Bl*57.3)
fprintf('lateral flapping = %6.1f degs\n',bls*57.3)
fprintf('longitudinal flapping = %6.1f degs\n',als*57.3)
fprintf('Lock number = %6.1f \n',lockno)
disp('Tail Rotor (zero if NOTAR)')
fprintf('tail rotor thrust = %6.1f lbs\n',Tt)
fprintf('advance ratio = %6.1f \n',mut)
fprintf('inflow parameter = %6.3f \n',lampt)
fprintf('Rotor coning angle = %6.1f degs\n',aot*57.3)
fprintf('lateral flapping = %6.2f degs\n',blst*57.3)
fprintf('longitudinal flapping = %6.2f degs\n',alst*57.3)
fprintf('Lock number = %6.1f \n',locknot)
disp('State Matrices')
disp('Longitudinal uncoupled plant (A or F depending on notation)')
disp('States are [u w q theta]')
disp('Flonaug')
disp('Longitudinal uncoupled input matrix (B or G depending on notation)')
disp('Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]')
disp('Glonaug')
disp('Lateral/directional uncoupled plant (A or F depending on notation)')
disp('States are [v p phi r psi]')
disp('Flataug')
disp('Lateral/directional uncoupled input matrix (B or G depending on notation)')
disp('Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]')
disp('Glataug')
disp('Coupled plant (A or F depending on notation)')
disp('States are [u w q theta v p phi r psi]')
disp('Coupled input matrix (B or G depending on notation)
Inputs are [longitudinal cyclic, collective, lateral cyclic, pedals]

Eigenvalue')

disp('Uncoupled')

disp('Longitudinal plant')

disp('Root wn damping')

[Rlonaug, wn, z1] = damp(Rlonaug);
disp([Rlonaug, wn, z1])

disp('Lateral/Directional plant')

disp('Root wn damping')

[Rlataug, wn, z1] = damp(Rlataug);
disp([Rlataug, wn, z1])

disp('Coupled Plant')

disp('Root wn damping')

[Rcoup wn z1] = damp(Rcoup);
disp([Rcoup wn z1])

*** KEY CONTROL PARAMETERS ***

disp('Designed damping')

pitch = 6.1f ft-lbs/(rad/sec)/n', desdm dq)
disp('Control Power')

roll = 6.1f ft-lbs/(rad/sec)/n', desd rdp)
disp('Control Power')

yaw = 6.1f ft-lbs/(rad/sec)/n', desd ndr)

Cooper Harper Pilot Ratings

disp('damping/momemt of inertia')

pitch = 6.1f ft-lbs/(rad/sec)/(slug ft^2)/n', prpitch)

roll = 6.1f ft-lbs/(rad/sec)/(slug ft^2)/n', prroll)

yaw = 6.1f ft-lbs/(rad/sec)/(slug ft^2)/n', pryaw)

Control power/momemt of inertia

pitch = 6.2f ft-lbs/(rad/sec)/(slug ft^2)/n', cpi pitch)
fprintf(' roll (dR/in)/Ixx = %6.2f (ft-lbs/in)/(slug ft^2)\n',cpiroll)
fprintf(' yaw (dN/in)/Izz = %6.2f (ft-lbs/in)/(slug ft^2)\n',cpiyaw)
disp('')
disp('')
diary off

clc
disp('')
disp(' *** OUTPUT DATA INSTRUCTIONS (screen 1 of 3) ***')
disp('')
disp('Because this subroutine generates a large number of single')
disp('value data not shown on the output screen, a text file')
disp('VARLIST.TXT is on this disk which lists the variable names')
disp('for all the stability derivatives. Stability derivative')
disp('contributions for all major aircraft components can be found')
disp('by reading the text file VARLIST.TXT, then asking MATLAB the')
disp('variable name corresponding to the derivative.')
disp('')
disp('Press any key to continue')
pause
clc
disp('')
disp(' *** OUTPUT DATA INSTRUCTIONS (screen 2 of 3) ***')
disp('')
disp('A. Data from the output screen saved to a file named:')
eval(['disp('' ',filenamel,'.stb'')''])
disp('')
disp('B. This is a text file, use the TYPE command to view the file')
disp(' or use a text editor to view/print the file.')
eval(['flag=exist(''','',filenamel,'.stb''');''])
if flag < 1,
    eval(['''!rename diary ''',filenamel,'.stb''''']);''
else
    eval(['''!del 1',filename1,1.stb11''']);''
end
disp('')
disp('C. Matrix and vector data saved to a default file named:')
disp(' mstabdat.mat')
disp('')
disp('D. This is a "mat" binary file, use the LOAD command to')
disp(' retrieve the data for plotting.')
disp('')
disp('E. Rename "mstabdat.mat" to another "mat" file.')
disp(' The file "mstabdat.mat" will be overwritten when')
disp(' the program is executed.')
disp('')
eval(['''!del ''',filenamel,'.mat'''']);''
eval(['''!rename diary ''',filenamel,'.mat''''']);''
disp('')
disp('F. Do not rename the file as ''',filenamel,'.mat''''')
eval(['''!del ''',filenamel,'.mat''''']);''
eval(['''!rename diary ''',filenamel,'.mat''''']);''
disp(' The file ''.mat'' is already on disk''')
disp(' and used for future editing.')
disp('')
disp('Press any key to continue')
pause

%%% Output to disk (.mat file containing matrix variables
%%% Amat Bmat Rcoupl Flataug Glataug Rlataug Plataug Flataug
%%% Glataug Rlonaug Plonaug
%%% *** Configuring variables for output ***
save mstabdat Amat Bmat Rcoup Flataug Glataug Rlataug Plataug Flataug ... Glataug RLonaug Plonaug

% clear
clc
disp('')
disp('*** OUTPUT DATA INSTRUCTIONS (screen 3 of 3) ***')
disp('')
disp('A. Single value data saved to a default file named:')
disp('vstabdat.mat')
disp('')
disp('B. This is a "mat" binary file, use the LOAD command to')
disp('retrieve the data for plotting.')
disp('')
disp('C. Rename "vstabdat.mat" to another "mat" file.')
disp('The file "vstabdat.mat" will be overwritten when')
disp('the program is executed.')
disp('')
eval(['disp(''D. Do not rename the file as "',filenamel,'"')''])
eval(['disp('' The file "',filenamel,'" is already on disk")'])
disp('and used for future editing.')
%
% *** Configuring variables for output ***
%
clear Amat Bmat Rcoup Flataug Glataug Rlataug Plataug Flataug ... Glataug Plonaug Plonaug num den vAl vB1 vals vbls vmu vtheta7 ... vao vvl vlam p vthetao vctsig vcqsig vchsig valtp
% save vstabdat
clear
clc
disp('')
disp('*** END STABILITY AND CONTROL ROUTINE ***')
disp('')
disp('press any key to continue...')
pause
format loose
%
% return to JANRAD.M

171
% CMDWBPLH.M

% open loop response plots for longitudinal and lateral plants

disp('While viewing a plot, press any key to go to the next plot')
disp('')
disp('Do you want to see longitudinal or lateral/directional plots?')
disp('')
disp('1. Longitudinal (eight plots total).')
disp('2. Lateral Directional (ten plots total).')
disp('')
pview=input('Enter a number : ');
clc
w=logspace(-2,2);
if pview==1
    % open loop response plots for longitudinal and lateral plants
    w=logspace(-2,2);
    Du=[0 0 0 0];
    Cu=[1 0 0 0];
    Cthet=[0 0 0 1];
    Cqrat=[0 1 0 1];
    Cw=[0 1 0 0];
    disp('longitudinal cylic')
    % command bw e to u
    [NUM,DEN]=ss2tf(Flonaug,Glonaug,Cu,Du,1);
    semilogx(w,20*log10(bode(NUM,DEN,w))),grid
    title('Open loop response Longitudinal Cyclic to U, Hover')
    xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
    pause
    !del cbe2uh.*
    meta cbe2uh
% command bw e to theta
    [NUM,DEN]=ss2tf(Flonaug,Glonaug,Cthet,Du,1);
    semilogx(w,20*log10(bode(NUM,DEN,w))),grid
    title('Open loop response Longitudinal Cyclic to Theta, Hover')
    xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
    pause
    !del cbe2theh.*
    meta cbe2theh
% command bw e to q
    [NUM,DEN]=ss2tf(Flonaug,Glonaug,Cqrat,Du,1);
    semilogx(w,20*log10(bode(NUM,DEN,w))),grid
    title('Open loop response Longitudinal Cyclic to Pitch Rate, Hover')
    xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
    pause
    !del cbe2qh.*
    meta cbe2qh
% command bw e to w
    [NUM,DEN]=ss2tf(Flonaug,Glonaug,Cw,Du,1);
    semilogx(w,20*log10(bode(NUM,DEN,w))),grid
    title('Open loop response Longitudinal Cyclic to W, Hover')
    xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
    pause
    !del cbe2wh.*
    meta cbe2wh
% now collective
    disp('collective')
% command bw c to u
    [NUM,DEN]=ss2tf(Flonaug,Glonaug,Cu,Du,2);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Collective to U, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbc2uh.*
meta cbc2uh
% command bw c to theta
[NUM,DEN]=ss2tf(Flonaug,Glonaug,Cthet,Du,2);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Collective to Pitch, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbc2theh.*
meta cbc2theh
% command bw c to q
[NUM,DEN]=ss2tf(Flonaug,Glonaug,Cqrat,Du,2);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Collective to Pitch Rate, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbc2qh.*
meta cbc2qh
% command bw c to w
[NUM,DEN]=ss2tf(Flonaug,Glonaug,Cw,Du,2);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Collective to W, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbc2wh.*
meta cbc2wh
clc
disp('Plots are saved under the following filenames:')
disp('')
disp('Longitudinal Cyclic to U, Hover - cbe2uh.met')
disp('Longitudinal Cyclic to Theta, Hover - cbe2theh.met')
disp('Longitudinal Cyclic to Pitch Rate, Hover - cbe2qh.met')
disp('Longitudinal Cyclic to W, Hover - cbe2wh.met')
disp('')
disp('Collective to U, Hover - cbc2uh.met')
disp('Collective to Pitch, Hover - cbc2theh.met')
disp('Collective to Pitch Rate, Hover - cbc2qh.met')
disp('Collective to W, Hover - cbc2wh.met')
disp('')
disp('press any key to continue ...')
pause
%
% now for lateral directional plant
else pview=2
%
% now for lateral directional plant
Cphi=[0 0 1 0 0];
Cv=[1 0 0 0 0];
Cp=[0 1 0 0 0];
Cr=[0 0 0 1 0];% yaw rate
Cy=[0 0 0 0 1];% yaw angle
% lateral cyclic
disp('lateral cyclic')
\* command by lateral cyclic to bank
\[ \text{NUM, DEN}\] =ss2tf(Flataug,Glataug,Cphi,Du,3);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Lateral Cyclic to Bank, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cba2phih.*
meta cba2phih
\* command by lateral cyclic to sideslip (lateral velocity)
\[ \text{NUM, DEN}\] =ss2tf(Flataug,Glataug,Cv,Du,3);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Lateral Cyclic to Sideslip (v), Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cba2vh.*
meta cba2vh
\* command by lateral cyclic to roll rate
\[ \text{NUM, DEN}\] =ss2tf(Flataug,Glataug,Cp,Du,3);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Lateral Cyclic to Roll Rate, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cba2ph.*
meta cba2ph
\* command by lateral cyclic to yaw rate
\[ \text{NUM, DEN}\] =ss2tf(Flataug,Glataug,Cr,Du,3);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Lateral Cyclic to Yaw Rate, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cba2rh.*
meta cba2rh
\* command by lateral cyclic to yaw angle
\[ \text{NUM, DEN}\] =ss2tf(Flataug,Glataug,Cy,Du,3);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Lateral Cyclic to Yaw, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cba2yh.*
meta cba2yh
\* pedes
\* command by pedals to bank
disp('pedals')
\[ \text{NUM, DEN}\] =ss2tf(Flataug,Glataug(:,4)*30,Cphi,[0],1);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Pedals to Bank, Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbp2phih.*
meta cbp2phih
\* command by pedals to sideslip
\[ \text{NUM, DEN}\] =ss2tf(Flataug,Glataug(:,4)*30,Cv,[0],1);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Pedals to Sideslip (v), Hover')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbp2vh.*
meta cbp2vh
\* command by pedals to roll rate
\[ \text{[NUM,DEN]} = \text{ss2tf(Flataug,Glataug(:,4)*30,Cp,[0],1);} \]
\[ \text{semilogx(w,20*log10(bode(NUM,DEN,w)))}, \text{grid} \]
\[ \text{title('Open loop response Pedals to Roll Rate, Hover')} \]
\[ \text{xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')} \]
\[ \text{pause} \]
\[ \text{!del cbp2ph.*} \]
\[ \text{meta cbp2ph} \]
\[ \text{t command by pedals to yaw rate} \]
\[ \text{[NUM,DEN]} = \text{ss2tf(Flataug,Glataug(:,4)*30,Cr,[0],1);} \]
\[ \text{semilogx(w,20*log10(bode(NUM,DEN,w)))}, \text{grid} \]
\[ \text{title('Open loop response Pedals to Yaw Rate, Hover')} \]
\[ \text{xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')} \]
\[ \text{pause} \]
\[ \text{!del cbp2rh.*} \]
\[ \text{meta cbp2rh} \]
\[ \% command bw pedals to yaw \]
\[ \text{[NUM,DEN]} = \text{ss2tf(Flataug,Glataug(:,4)*30,Cy,[0],1);} \]
\[ \text{semilogx(w,20*log10(bode(NUM,DEN,w)))}, \text{grid} \]
\[ \text{title('Open loop response Pedals to Yaw, Hover')} \]
\[ \text{xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')} \]
\[ \text{pause} \]
\[ \text{!del cbp2yh.*} \]
\[ \text{meta cbp2yh} \]
\[ \% command bw pedals to yaw \]
\[ \text{disp('')} \]
\[ \text{disp('Press any key to continue ...')} \]
\[ \text{pause} \]
\[ \% \]
\[ \text{else} \]
\[ \text{end} \]
open loop response plots for longitudinal and lateral plants

while viewing a plot, press any key to go to the next plot
Do you want to see longitudinal or lateral/directional plots?

1. Longitudinal (eight plots total).
2. Lateral Directional (ten plots total).

pview=input('Enter a number : ');

c1c
w=logspace(-2,2);
if pview==1
Du=[0 0 0 0];
Cu=[1 0 0 0];
Cthet=[0 0 0 1];
Cqrat=[0 0 1 0];
Cw=[0 1 0 0];
disp('longitudinal clyc')
\% command bw e to u
[NUM,DEN]=ss2tf(Flonaug,Glonaug,Cu,Du,1);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Longitudinal Cyclic to U, Cruise')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbe2uc.*
meta cbe2uc
\% command bw e to theta
[NUM,DEN]=ss2tf(Flonaug,Glonaug,Cthet,Du,1);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Longitudinal Cyclic to Theta, Cruise')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbe2thec.*
meta cbe2thec
\% command bw e to q
[NUM,DEN]=ss2tf(Flonaug,Glonaug,Cqrat,Du,1);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Longitudinal Cyclic to Pitch Rate, Cruise')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbe2qc.*
meta cbe2qc
\% command bw e to w
[NUM,DEN]=ss2tf(Flonaug,Glonaug,Cw,Du,1);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Longitudinal Cyclic to W, Cruise')
xlabel('Angular Frequency (rad/sec)'),ylabel('Gain (dB)')
pause
!del cbe2wc.*
meta cbe2wc
\% now collective
disp('collective')
\% command bw c to u
[NUM,DEN]=ss2tf(Flonaug,Glonaug,Cu,Du,2);
semilogx(w,20*log10(bode(NUM,DEN,w))),grid
title('Open loop response Collective to U, Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause
!del cbc2uc.*
meta cbc2uc
% command bw c to theta
[NUM, DEN] = ss2tf(Flonaug, Glonaug, Cthet, Du, 2);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
title('Open loop response Collective to Pitch, Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause
!del cbc2thec.*
meta cbc2thec
% command bw c to q
[NUM, DEN] = ss2tf(Flonaug, Glonaug, Cqrat, Du, 2);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
title('Open loop response Collective to Pitch Rate, Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause
!del cbc2qc.*
meta cbc2qc
% command bw c to w
[NUM, DEN] = ss2tf(Flonaug, Glonaug, Cw, Du, 2);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
title('Open loop response Collective to W, Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause
!del cbc2wc.*
meta cbc2wc
cic
disp('')
disp('Plots are saved under the following filenames:')
disp('')
disp('Longitudinal Cyclic')
disp('Longitudinal Cyclic to U, Cruise - cbe2uc.met')
disp('Longitudinal Cyclic to Theta, Cruise - cbe2thec.met')
disp('Longitudinal Cyclic to Pitch Rate, Cruise - cbe2qc.met')
disp('Longitudinal Cyclic to W, Cruise - cbe2wc.met')
disp('')
disp('Collective')
disp('Collective to U, Cruise - cbc2uc.met')
disp('Collective to Pitch, Cruise - cbc2thec.met')
disp('Collective to Pitch Rate, Cruise - cbc2qc.met')
disp('Collective to W, Cruise - cbc2wc.met')
disp('')
disp('press any key to continue ...')
pause
% now for lateral directional plant
elseif pview==2
Cphi=[0 0 1 0 0];
CV=[1 0 0 0 0];
CP=[0 1 0 0 0];
CR=[0 0 0 1 0];% yaw rate
CY=[0 0 0 0 1];% yaw angle
% lateral cyclic
disp('lateral cyclic')
% command bw lateral cyclic to bank
[NUM, DEN] = ss2tf(Flataug, Glataug, Cphi, Du, 3);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
% command by lateral cyclic to sideslip (lateral velocity)
[NUM, DEN] = ss2tf(Flataug, Glataug, Cv, Du, 3);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
title('Open loop response Lateral Cyclic to Sideslip (v), Cruise')
pause

% command by lateral cyclic to roll rate
[NUM, DEN] = ss2tf(Flataug, Glataug, Cp, Du, 3);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
title('Open loop response Lateral Cyclic to Roll Rate, Cruise')
pause

% command by lateral cyclic to yaw rate
[NUM, DEN] = ss2tf(Flataug, Glataug, Cr, Du, 3);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
title('Open loop response Lateral Cyclic to Yaw Rate, Cruise')
pause

% command by lateral cyclic to yaw angle
[NUM, DEN] = ss2tf(Flataug, Glataug, Cy, Du, 3);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
title('Open loop response Lateral Cyclic to Yaw, Cruise')
pause

% command by pedals to bank
disp('pedals')
[NUM, DEN] = ss2tf(Flataug, Glataug(:,4)*30, Cphi, [0, 1]);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
title('Open loop response Pedals to Bank, Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause

% command by pedals to sideslip
[NUM, DEN] = ss2tf(Flataug, Glataug(:,4)*30, Cv, [0, 1]);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
title('Open loop response Pedals to Sideslip (v), Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause

% command by pedals to roll rate
[NUM, DEN] = ss2tf(Flataug, Glataug(:,4)*30, Cp, [0, 1]);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
title('Open loop response Pedals to Roll Rate, Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause
!del cbp2pc.*
meta cbp2pc
% command by pedals to yaw rate
[NUM, DEN] = ss2tf(Flataug, Glataug(:, 4)*30, Cr, [0], 1);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
title('Open loop response Pedals to Yaw Rate, Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause
!del cbp2rc.*
meta cbp2rc
% command by pedals to yaw
[NUM, DEN] = ss2tf(Flataug, Glataug(:, 4)*30, Cy, [0], 1);
semilogx(w, 20*log10(bode(NUM, DEN, w))), grid
title('Open loop response Pedals to Yaw, Cruise')
xlabel('Angular Frequency (rad/sec)'), ylabel('Gain (dB)')
pause
!del cbp2yc.*
meta cbp2yc

clc
disp(' ')
disp('Plots are saved under the following filenames:')
disp('')
disp('Lateral cyclic')
disp('Lateral Cyclic to Bank, Cruise - cba2phic.met')
disp('Lateral Cyclic to Sideslip (v), Cruise - cba2vc.met')
disp('Lateral Cyclic to Roll Rate, Cruise - cba2pc.met')
disp('Lateral Cyclic to Yaw Rate, Cruise - cba2rc.met')
disp('Lateral Cyclic to Yaw, Cruise - cba2yc.met')
disp('Pedals')
disp('Pedals to Bank, Cruise - cbp2phic.met')
disp('Pedals to Sideslip (v), Cruise - cbp2vc.met')
disp('Pedals to Roll Rate, Cruise - cbp2pc.met')
disp('Pedals to Yaw Rate, Cruise - cbp2rc.met')
disp('Pedals to Yaw, Cruise - cbp2yc.met')
disp('')
disp('press any key to continue ...')
pause
%
else
end
0. DERIV2.M
% DERIV2.M
% computes the derivative of a polynomial
function [deriv] = deriv(polyno);
der = [2 1 0];
d = der .* polyno;
deriv = [d(1,1) d(1,2)];
LIST OF REFERENCES


<table>
<thead>
<tr>
<th>No.</th>
<th>Distribution List</th>
<th>No. of Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Defense Technical Information Center</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Cameron Station</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Alexandria, Virginia 22304-6145</td>
<td></td>
</tr>
<tr>
<td>2.</td>
<td>Library, Code 52</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5000</td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>Professor Daniel J. Collins, Code AA/Co</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Department of Aeronautics and Astronautics</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5000</td>
<td></td>
</tr>
<tr>
<td>4.</td>
<td>Professor E. Roberts Wood, code AA/Wd</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Department of Aeronautics and Astronautics</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5000</td>
<td></td>
</tr>
<tr>
<td>5.</td>
<td>Professor Richard M. Howard AA/Ho</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Department of Aeronautics and Astronautics</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Naval Postgraduate School</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monterey, California 93943-5000</td>
<td></td>
</tr>
<tr>
<td>6.</td>
<td>MAJ Robert K. Nicholson, USA</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1116 Walker Road</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Great Falls, Virginia 22066</td>
<td></td>
</tr>
<tr>
<td>7.</td>
<td>MAJ Walter M. Wirth, USA</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>P.O. Box 201</td>
<td></td>
</tr>
<tr>
<td></td>
<td>456 North Monroe Street</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Monticello, Wisconsin 53570</td>
<td></td>
</tr>
<tr>
<td>8.</td>
<td>Dr. Dev Banerjee</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Bldg. 530/B325</td>
<td></td>
</tr>
<tr>
<td></td>
<td>McDonnell Douglas Helicopter Co.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5000 E. McDowell road</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Mesa, AZ 85215</td>
<td></td>
</tr>
<tr>
<td>9.</td>
<td>Mr. Dean Carico</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>RW04B NAWCAD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Patuxent River, MD 20670-5304</td>
<td></td>
</tr>
<tr>
<td>10.</td>
<td>Mr. Dean E. Cooper</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Sikorsky Aircraft Div., UTC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6900 Main Street</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stratford, CT 06601</td>
<td></td>
</tr>
</tbody>
</table>
11. Mr. Andrew W. Kerr  
Aeroflightdynamics Directorate  
NASA Ames Research Center  
Moffett Field, CA  94035

12. Mr. John C. McKeown  
Naval Air Systems Command  
Code AIR-530A  
1421 Jefferson Highway  
Arlington, VA  22243-5300