PIFCGT - A PIF Autopilot Design Program for General Aviation Aircraft

John R. Broussard

Contract NASI-16303
December 1983

FOREIGN DOMESTIC DISSEMINATION

Because of its significant early commercial potential, this information, which has been developed under a U.S. Government program, is being disseminated within the United States in advance of general publication. This information may be duplicated and used by the recipient with the express limitation that it not be published. Release of this information to other domestic parties by the recipient shall be made subject to these limitations.

Foreign release may be made only with prior NASA approval and appropriate export licenses. This legend shall be marked on any reproduction of this information in whole or in part.

Review for general release December 31, 1985

NASA Contractor Report 166123

NASA-CR-166123
19860005850
FOREWORD

The computer program described in this report was developed by Information & Control Systems, Incorporated (ICS) during the period from November 1980 to August 1981, under Contract NAS1-16303 for the National Aeronautics and Space Administration, Langley Research Center, Hampton, Virginia. This work was sponsored by the Avionics Technology Research Branch of the Flight Electronics Division. Dr. David R. Downing and Mr. Wayne H. Bryant served as Technical Representatives monitoring this contract.

The author wishes to express appreciation to Dr. E. Armstrong of the Langley Research Center, for providing ICS with information on the ORACLS program. The author was aided in computer software development by D. Taylor and M. Aherron of the ICS staff.
This Page Intentionally Left Blank
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I.</td>
<td>PIFCGT OVERVIEW</td>
<td>1</td>
</tr>
<tr>
<td>A.</td>
<td>PROGRAM IMPLEMENTATION</td>
<td>2</td>
</tr>
<tr>
<td>B.</td>
<td>DOCUMENTATION</td>
<td>3</td>
</tr>
<tr>
<td>II.</td>
<td>PIFCGT DESCRIPTION</td>
<td>4</td>
</tr>
<tr>
<td>A.</td>
<td>PIFCGT CHAPTERS</td>
<td>5</td>
</tr>
<tr>
<td>B.</td>
<td>DEFINITION OF VARIABLES</td>
<td>11</td>
</tr>
<tr>
<td>III.</td>
<td>SUBROUTINE DOCUMENTATION</td>
<td>18</td>
</tr>
<tr>
<td>A.</td>
<td>BLOCK DATA (AEROD)</td>
<td>19</td>
</tr>
<tr>
<td>B.</td>
<td>SAMPLED-DATA REGULATOR (ALPHA AND BETA)</td>
<td>20</td>
</tr>
<tr>
<td>C.</td>
<td>SOLUTION OF A<em>Y</em>B-X=C (AXBMC)</td>
<td>22</td>
</tr>
<tr>
<td>D.</td>
<td>CREATE WEIGHTING MATRICES (CFWM)</td>
<td>25</td>
</tr>
<tr>
<td>E.</td>
<td>FEEDFORWARD MATRICES (CGTPIF)</td>
<td>28</td>
</tr>
<tr>
<td>F.</td>
<td>CLOSED-LOOP d MODEL (CLDMOD)</td>
<td>35</td>
</tr>
<tr>
<td>G.</td>
<td>CLOSED-LOOP Y MODEL (CLYMOD)</td>
<td>37</td>
</tr>
<tr>
<td>H.</td>
<td>CLOSED-LOOP ψ AND Z MODEL (CLXMOD)</td>
<td>39</td>
</tr>
<tr>
<td>I.</td>
<td>COMBINE MATRICES (CORNER)</td>
<td>41</td>
</tr>
<tr>
<td>J.</td>
<td>DIAGONAL PARTITION (DIAPAR)</td>
<td>43</td>
</tr>
<tr>
<td>K.</td>
<td>FORM A DIAGONAL MATRIX (DIAPUT)</td>
<td>45</td>
</tr>
<tr>
<td>L.</td>
<td>DIMENSIONS (DIMSS)</td>
<td>47</td>
</tr>
<tr>
<td>M.</td>
<td>DISCRETE MODEL CONSTRUCTION (DISCMOD)</td>
<td>48</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (CONTINUED)

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>N</td>
<td>DISCRETE PIF CONTROL LAW SIMULATION (DPIFS)</td>
<td>51</td>
</tr>
<tr>
<td>O</td>
<td>DISCRETE RICCATI EQUATION SOLUTION (DREG)</td>
<td>56</td>
</tr>
<tr>
<td>P</td>
<td>SUBSET OF A MATRIX (EXTR)</td>
<td>60</td>
</tr>
<tr>
<td>Q</td>
<td>F AND G CONSTRUCTION FROM AERODYNAMIC COEFFICIENTS (FGAERO)</td>
<td>62</td>
</tr>
<tr>
<td>R</td>
<td>EIGENVALUES/EIGENVECTORS OF CLOSED-LOOP PLANTS USING CSQZ (FREEF)</td>
<td>70</td>
</tr>
<tr>
<td>S</td>
<td>EIGENVALUES/EIGENVECTORS OF CLOSED-LOOP PLANTS USING EIGEN (FREEO)</td>
<td>74</td>
</tr>
<tr>
<td>T</td>
<td>ACCELERATION AND VELOCITY VECTOR OBSERVATION MATRIX CONSTRUCTION (HADAC)</td>
<td>77</td>
</tr>
<tr>
<td>U</td>
<td>FM, GM, HM, DM, H AND D MATRIX CONSTRUCTION (HDCON)</td>
<td>83</td>
</tr>
<tr>
<td>V</td>
<td>INTERPOLATE AERODYNAMIC DATA (INTERP)</td>
<td>89</td>
</tr>
<tr>
<td>W</td>
<td>MULTIPLY A TRANSPOSE TIMES B TIMES A (MATBA)</td>
<td>90</td>
</tr>
<tr>
<td>X</td>
<td>REARRANGE MATRIX (MOVEALL)</td>
<td>92</td>
</tr>
<tr>
<td>Y</td>
<td>PIF EIGENVALUE PLACEMENT (PIFEIG)</td>
<td>94</td>
</tr>
<tr>
<td>Z</td>
<td>PIF CLOSED-LOOP EIGENVALUE EIGENVECTOR CALCULATIONS (PIFFRE)</td>
<td>99</td>
</tr>
<tr>
<td>AA</td>
<td>PIF GAIN COMPUTATIONS (PIFG)</td>
<td>101</td>
</tr>
<tr>
<td>BB</td>
<td>CONSTRUCT THE PIF MODELS (PIFMODL)</td>
<td>106</td>
</tr>
<tr>
<td>CC</td>
<td>PIF EIGENVALUE PLACEMENT (PIFPLC)</td>
<td>110</td>
</tr>
<tr>
<td>DD</td>
<td>OUTPUT NUMERICAL DATA (PRNT)</td>
<td>114</td>
</tr>
<tr>
<td>EE</td>
<td>PRINT TITLE (PRNTITL)</td>
<td>116</td>
</tr>
<tr>
<td>FF</td>
<td>OUTPUT LABELED NUMERICAL DATA (PRNT2)</td>
<td>118</td>
</tr>
</tbody>
</table>
TABLE OF CONTENTS (CONTINUED)

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GG. Sampled-data regulator using alpha and beta (qrmhat)</td>
<td>121</td>
</tr>
<tr>
<td>HH. Sampled-data regulator using ORACLS (QRMP1)</td>
<td>126</td>
</tr>
<tr>
<td>II. Reduce matrices (REMAT)</td>
<td>131</td>
</tr>
<tr>
<td>JJ. Step response eigenvalue information (RESPON)</td>
<td>134</td>
</tr>
<tr>
<td>KK. Print out information (RUINFO)</td>
<td>136</td>
</tr>
<tr>
<td>LL. Create a (3XN) matrix from 3 different 1XN matrices (R3MAT)</td>
<td>137</td>
</tr>
<tr>
<td>MM. Schur form solution of A * X * B - X = C (SHRSOL)</td>
<td>139</td>
</tr>
<tr>
<td>NN. Round off small numbers (SMALL)</td>
<td>141</td>
</tr>
<tr>
<td>OO. Iterative refinement solution of A * X * B - X = C (SOLVER)</td>
<td>142</td>
</tr>
<tr>
<td>IV. Summary - Design procedure</td>
<td>145</td>
</tr>
<tr>
<td>REFERENCES</td>
<td>147</td>
</tr>
<tr>
<td>Appendix A</td>
<td>148</td>
</tr>
<tr>
<td>Appendix B</td>
<td>167</td>
</tr>
<tr>
<td>Appendix C</td>
<td>169</td>
</tr>
</tbody>
</table>
I. PIFCGT OVERVIEW

The PIFCGT program is a collection of FORTRAN-coded subroutines which can be used to formulate, design, and evaluate a proportional-integral-filter (PIF) digital flight control law. The PIF control law is designed using the Linear Quadratic Regulator (LQR) formulation to obtain feedback gains and command generator tracker (CGT) theory to obtain feedforward gains. Structurally, the PIFCGT program is driven by a main executive which calls individual modular subroutines. The PIFCGT main executive is written using top-down structure and depends on user input to determine which of the program subroutines are to be used for a given control law design.

Flexibility in the PIFCGT program is accomplished by using a general multivariable control design approach and employing dynamic (vector) data storage. In almost all cases, data arrays in PIFCGT subroutines are treated as packed one-dimensional arrays which can easily be passed between subroutines without a maximum array size parameter appearing as an argument of the calling sequence. The dynamic storage capability allows subroutine program size to be specified and controlled through the PIFCGT main executive. The use of common blocks has been reduced to a minimum (PIFCGT, AEROD, CRWM, DPIFS, FGAERO, RUNINFO) in order to increase subroutine modularity and portability.

The PIFCGT program is explicitly written for a specific application: the design of any one of nine PIF autopilots for the NAVION general aviation aircraft. The PIFCGT program has been developed as part of
the effort in NASA Langley Research Center's General Aviation Terminal Area Operation Research (GATOR) program. The GATOR program is directed at developing and evaluating advanced flight control, guidance and display concepts that make use of the recent advances in digital flight control hardware, digital control theory, navigation and landing systems, and electronic displays.

This report constitutes the second part of a two part report. The first part in Ref. 1 is a discussion of the derivation, design, implementation, and flight testing of the PIF control law autopilots. This report documents the software that is used by a PIF control law designer to produce PIF control gains, evaluate PIF closed-loop performance, and determine PIF linear simulations to command inputs.

A. PROGRAM IMPLEMENTATION

The PIIFCGT program was developed using the facilities of the NASA Langley Research Center Computer Complex. The program was written for execution on a CDC 6600 using the FORTRAN FTN4 compiler.

PIIFCGT requires 140,000 octal locations for execution using a segmentation procedure. PIIFCGT requires 224,300 octal locations for execution without segmentation. The maximum number of model states allowed is 20 (12 aircraft states + 4 control states + 4 integrator states in a fully coupled PIF aircraft control design). The maximum number of control and command states allowed is 4.

A command file consisting of the JOB control information is required to execute each of the autopilot designs under the NASA Langley NOS 1.3 operating system. A command file example for the NOS 1.3 operating system is presented in Appendix B along with the segmentation directives in Appendix C.
All of the data needed by PIFCGT from a user designing one of the nine autopilots is read from a NAMELIST file. The NAMELIST files used in the NAVION PIF designs for each autopilot are presented in Appendix A.

The PIFCGT program can be easily extended to design one of the nine autopilots for a variety of aircrafts other than the NAVION by simply changing the aerodynamic data base in the BLOCK DATA (AEROD) subroutine. Changes to the aerodynamic data base can also be performed using the NAMELIST variables. A new autopilot command system (autothrottle for example) not currently available in PIFCGT can be incorporated into the program by adding (i.e., not modifying) software in three subroutines (HDCON, DPIFS, and PIFG) in the form of alternative "GO TO XX" code sections. This built-in "expandability" feature is a special feature of PIFCGT that makes new autopilot designs straightforward.

### B. Documentation

The next two chapters in the report are directed to input and output requirements of the programs and subroutines. Chapter 2 is devoted to the main program PIFCGT. A step by step discussion is given for the flow of the program and all variables used in the program are defined in a Table. Chapter 3 describes each of the primary level subroutines used by PIFCGT. The secondary level subroutines are composed of subroutines from the ORACLS library system (ORACLIB), Ref. 2, and specialized subroutines written by Information & Control Systems. Documentation for ORACLIB can be found in Ref. 2 and is not presented in this report.
II. PIFCGT DESCRIPTION

PIFCGT uses a top-down structure that depends on user input to determine which program chapters are to be used for a given design. There are five main chapters: data initialization and set up, PIF matrix construction, feedforward matrix construction, PIF design and performance analysis, and PIF time history computation. Two flags control the main logic, ICH(I,J) and ICHOSE. ICH(I,J) indicates which chapters are to be executed; I represents the ith chapter and J represents the jth section within the ith chapter. ICHOSE is an integer which indicates which autopilot mode is to be designed by the program. The nine autopilot ICHOSE options are shown in Table 1. Two of the APF LOC autopilots may eventually be discontinued depending on flight test results.

During the course of developing PIFCGT, print statements have been used to check intermediate calculations. All the print statements remain in PIFCGT, but are inactivated by setting logical variables beginning with the 4 letters "DEBU" to FALSE. Table 1 lists the areas of software for which each "DEBUG" variable activates print statements.

Two features may be added to the PIFCGT program by a user to enhance its capabilities. The ORACLS subroutines, EIGEN (used by FREEO, Section S) sometimes has numerical difficulties computing eigenvalues with sufficient accuracy. An alternative eigenvalue/eigenvector procedure not available in ORACLS (but available in FTNMLIB - a NASA Langley computer center math library) is discussed in FREEF, Section R. The second feature is a plotting capability. A plotting package was developed for PIFCGT but is highly machine dependent and is not documented in this report.
A. PIFCGT CHAPTERS

This section presents a list of the chapters that can be activated by the logical variable ICH. The format for the presentation of the PIFCGT chapters consists of the logical array variable element, the name of the program or subroutine that the logical element activates and a brief description of the function performed. The user should follow the logic in the PIFCGT main executive program while reading this section.

ICH(1,1) (PIFCGT) INITIALIZE CONSTANTS

PIFCGT begins by calling the ORACLS subroutine RDTITL. RDTITL defines data statements in internal ORACLS common blocks and reads a user defined title card which is shown at the end of the submit file in Appendix B.

ICH(1,2) (PIFCGT) READ IN FLAGS AND DESIGN PARAMETERS

The NAMELISTS NAM1, NAM2, NAM3, NAM5, NAM6, AERO1, AERO2, and NAM7 are read in from logical unit 20. If "DEBUXX" variables are TRUE, the NAMELISTS are printed using the namelist output format.

ICH(1,3) (RUNINFO) PRINT OUT FLAGS AND DESIGN PARAMETERS

The subroutine RUNINFO is called for the first time. RUNINFO is the information subroutine that echos the flag conditions and chapter calls the user has specified in the namelists.

ICH(1,4) (INTERP) INTERPOLATES AERODYNAMIC DATA AT VTP

The aerodynamic coefficients in the common blocks AERO and AER1 are assumed to be for two different straight and level trim conditions (a lower and a higher forward velocity, respectively). INTERP uses the variable VTP as an intermediate velocity to interpolate the coefficients for a trim condition at VTP. The interpolated values overwrite the values in AERO.
ICH(1,4) (FGAERO) CONSTRUCT F AND G (BODY AXIS) MODEL MATRICES

The nonlinear equations of motion of the aircraft have been explicitly linearized about a straight and level flight condition. The linearization equations and the aerodynamic data in AERO are used in FGAERO to construct the \((12 \times 12)\) plant system matrix, \(F\), and the \((12 \times 4)\) control effect matrix, \(G\).

ICH(1,5) (FGAERO) TRANSFORM F AND G SO ANGLES AND CONTROLS ARE IN DEG

FGAERO constructs \(F\) and \(G\) with units of feet for distance and radians for angles. If ICH(1,5) is TRUE (as is the case in all NAME-LISTS in Appendix A), \(F\) and \(G\) are converted to use angular units of degrees. Units of feet and radians produce numerically ill-conditioned matrices in later computations.

ICS(1,10) (DIMSS) SET UP DIMENSION PARAMETERS

ORACLS requires that each matrix vector have a two-dimensional integer vector that specifies the first dimension of the matrix in the first location of the integer vector and the second dimension of the matrix in the second location. DIMSS initializes the two-dimensional integer vectors.

ICH(1,11) (PIFCGT) PRINT OUT THE F AND G MATRICES

ICH(1,15) (FREEO) FIND EIGENVALUES AND EIGENVECTORS OF OPEN-LOOP F MATRIX

ICH(1,16) (HADAC) CONSTRUCT HA AND DA ACCELERATION OBSERVATION MATRICES

Later in the construction of the PIF control law (PIFG) some of the state feedback elements \((\Delta u, \Delta v, \Delta w)\) can be replaced with accelerometer output feedback \((\Delta a_x, \Delta a_y, \Delta a_z)\) by transforming the feedback gains. The matrices \(HA\) and \(DA\) mathematically relate the linearized accelerometer output with the body-axis aircraft states and controls using
\[ \Delta a = HA \Delta x + DA \Delta u \]  

(1)

and are required to perform the feedback gain transformation.

ICH(2,2) (HDCON) DEPENDING ON ICHOOSE CONSTRUCT H, D, FM, GM, HM, AND DM MATRICES

The command output matrices (H and D) and command model matrices (FM, GM, HM, and DM) are constructed in HDCON depending on the value of ICHOOSE. Logical vectors, XR, UR, YR, XMR, and UMR are initialized for later use in REMAT.

ICH(1,3) (RUNINFO) PRINT OUT FLAGS AND DESIGN PARAMETERS

RUNINFO is called for the second time and echoes values of parameters constructed in subroutines and specified in NAMELISTS.

ICH(2,4) (CFWM) CONSTRUCT COST FUNCTION WEIGHTING MATRICES; A, R, AM, RD, QZ

Linear Quadratic Cost function weighting elements in the common block QWHT are used to form the cost function weighting matrices Q, R, AM, RD, and QZ.

ICH(2,5) (PIFMODL) CONSTRUCT PIF MODEL MATRICES PHI, GAMA, QHAT, RHAT

The continuout-time mathematical model matrices for the PIF Linear Quadratic Regulator problem are formed.

ICH(2,6) (REMAT) REDUCE MATRICES BY ELIMINATION

Using the logical vectors XR, UR, YR, XMR, and UMR, undesired states, controls and commands are eliminated from the PIF matrices. The result is that the fully coupled PIF design matrices are reduced in dimension. The reduction performed in REMAT usually leaves only longitudinal related states or lateral-directional related states.

ICH(2,7) (PIFEIG) FORM DESIRED CLOSED-LOOP MODEL FROM PLANT MODEL

The Riccati matrix equation solution procedure in DREG requires a
starting stabilizing PIF gain to find a solution. The ORACLS subroutine, DSTAB, often is not able to numerically find a stabilizing gain matrix. PIFEIG, in combination with PIFPLC, finds a stabilizing gain matrix using an eigenvalue/eigenvector placement procedure. PIFEIG constructs a PIF model that has stable eigenvalues. The procedure works even if the PIF model is stabilizable but not controllable.

ICH(2,8) (PIFMODL) REPLACE U FEEDBACK WITH AX FEEDBACK IN HX DX
ICH(2,9) (PIFMODL) REPLACE V FEEDBACK WITH AY FEEDBACK IN HX DX
ICH(2,10) (PIFMODL) REPLACE W FEEDBACK WITH AZ FEEDBACK IN HX DX
ICH(2,12) (PIFMODL) REPLACE AX FEEDBACK WITH VT FEEDBACK IN HX DX
ICH(2,13) (PIFMODL) REPLACE AY FEEDBACK WITH BETA FEEDBACK IN HX DX
ICH(2,14) (PIFMODL) REPLACE AZ FEEDBACK WITH ALPHA FEEDBACK IN HX DX

The feedback gain in PIFG is transformed to allow for different sensors. AX, AY, and AZ are the body mounted accelerometer sensor observations. VT, BETA, and ALPHA are the pitot tube, sideslip vane and angle of attack vane sensor observations. If ICH(2,12) is TRUE, then ICH(2,8) must be TRUE, etc. HX and DX are the sensor observation matrices. Observation matrices for VT, BETA, and ALPHA are formed in HADAC.

ICH(2,11) (PIFEIG) USE RANDOM NUMBERS TO HELP FIND PHICL

The stable model formed in PIFEIG and stored in PHICL uses the feedback gain GK during construction. The elements in GK are generated internally in PIFEIG if ICH(2,11) is TRUE or must be passed to PIFEIG by the designed from NAMS if ICH(2,11) is FALSE.

ICH(3,1) (DISCMOD) FORM DISCRETE EQUIVALENT OF CONTINUOUS MODEL AND PLANT

If ICH(3,1) is FALSE the feedforward matrix construction scalar, SSS, is set to 0.0 and DF, DG, DFM, and DGM are set to equal F, G, FM, and GM, respectively. If ICH(3,1) is TRUE the feedforward matrix scalar, S
SSS is set to 1.0 and the discrete plant and model representations are obtained for a zero-order hold in the control.

One of the objectives of the lateral-directional command model is to cause the sideslip to be zero when the aircraft is banked in a turn. When designing the ICHOOSE= 9 (APR LOCI) autopilot, the second element in DGM must be adjusted so that the feedforward matrix element S12(1) is zero. \( S12(1) \phi_c \) is the steady state side velocity. The second element in DGM is adjusted by the program automatically, a maximum of IKDGM times using the feedback equation

\[
DGM(2) = DGM(2) + DKDGM * DELT * S12(1)
\]  

IKDGM and DKDGM are specified by the designer so that the iterative adjustment procedure shown in Eq. 2 is stable.

If ICHOOSE=15 or 16 (APR LOCR or APR LOCP autopilots) the digital model matrix element DGM(3) is set to zero. \textbf{Note:} If ICHOOSE=16 is the desired autopilot design, the subroutine HDCON makes the appropriate adjustments, then resets ICHOOSE to ICHOOSE=15. The ICHOOSE=16 (APR LOCP) design is performed this way because it is very similar to the ICHOOSE=15 command model (APR LOCR).

ICH(3,2) (CGTPIF) SOLVE FOR FEEDFORWARD MATRICES

A description of this subroutine, which is an involved procedure for finding the feedforward matrices S11, S21, S22, and S23, is given in Chapter 3, Section E.

ICH(4,2) (QRMP1) FORM DISCRETE EQUIVALENT OF CONTINUOUS SYSTEM USING ORACLS SUBROUTINES

ICH(4,3) (QRMHAT) FORM DISCRETE EQUIVALENT OF CONTINUOUS SYSTEM USING ALPHA AND BETA SUBROUTINES
These two subroutines transform a continuous-time Linear Quadratic Regulator PIF control problem into a sampled-data regulator PIF control problem. ORACLS fails to perform the transformation if there is only one control; QRMHAT must be used under this condition. When both subroutines execute properly, their answers are identical.

ICH(4,4) (PIFPLC) PLACE THE EIGENVALUES AND EIGENVECTORS USING PIF CONTROL

PIFPLC uses the discrete PIF plant representation and the discrete plant representation of PHICL (constructed in QRMHAT (or ORMP1)) to determine a stabilizing feedback gain using eigenvalue/eigenvector placement.

ICH(4,5) (PIFPLC) FIND LOG OF F + GK IN PIFPLC BEFORE FINDING EIGENVALUES

After PIFPLC determines the stabilizing feedback gain, the resulting closed-loop eigenvalues are printed out if DEBU44 is TRUE. If ICH(4,5) is TRUE, the discrete-time closed-loop plant is transformed to the equivalent continuous-time plant using the log of a matrix before determining the eigenvalues.

ICH(4,6) (DREG) SOLVE DISCRETE RICCATI EQUATION FOR FEEDBACK GAINS

The discrete-time algebraic Riccati equation is solved using Newton's method. If ICH(2,7) is TRUE, the initial feedback gain GK is assumed to be stabilizing. If ICH(2,7) is FALSE, the subroutine DREG attempts to find an initial stabilizing gain, using the ORACLS subroutine DSTAB. DSTAB is not always successful.

ICH(4,7) (LOGX1) FIND EQUIVALENT CONTINUOUS SYSTEM FOR A CLOSED-LOOP PIF CONTROLLER

The log of a matrix is used to transform PHICL=PHI-GAMA*GK to an equivalent continuous plant. The closed-loop eigenvalues are stable in the left-half complex plane.
ICH(4,8) (PIFFRE) FIND CLOSED-LOOP EIGENVALUES AND EIGENVECTORS

The ORACLIS subroutine EIGEN is used to find the eigenvalues and eigenvectors of PHICL. EIGEN is sometimes inaccurate.

ICH(4,9) (PIFG) COMPUTE THE PIF GAINS

The optimal regulator gains and feedforward gains are combined in PIFG to produce the final implementable PIF controller. If METERS is TRUE, PIFG transforms the implementable gains to use units of meters for distance and radians for angular units.

ICH(5,3) (DPIFS) SIMULATE PIF CONTROLLER FOR COMMAND INPUTS

ICH(5,4) (DPIFS) PRINT OUT PLANT STATES AND CONTROLS OF SIMULATION

ICH(5,5) (DPIFS) PRINT OUT STAR TRAJECTORY OF THE SIMULATION

Using information specified by the designer in UMAG, USTRT and USTP (discussed in HDCON), DPIFS simulates the PIF controller in the form used for implementation. A nonlinear representation of the command model is used to generate the command model trajectories. A linear model of the aircraft is used to propagate the plant dynamics. The plant states and controls are printed out if ICH(5,4) is TRUE. The star trajectory computed using the feedforward matrices is printed out if ICH(5,5) is TRUE.

B. DEFINITION OF VARIABLES

All variables used in the PIFCGT main program are defined in Table 1. The common block for each variable is also identified in Table 1.
TABLE 1. PIFCGT Variable Definitions

<table>
<thead>
<tr>
<th>MATRICES</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AICL</td>
<td>WORK MATRIX</td>
</tr>
<tr>
<td>AM</td>
<td>STATE, CONTROL CROSS WEIGHTING CAUSED BY QOVE IN CONTINUOUS PLANT</td>
</tr>
<tr>
<td>AMHAT</td>
<td>((N+M+L) \times M) PIF DISCRETE CROSS WEIGHTING MATRIX</td>
</tr>
<tr>
<td>DAC</td>
<td>((3 \times N)) ACCELEROMETER OUTPUT CONTROL OBSERVATION MATRIX</td>
</tr>
<tr>
<td>DX</td>
<td>((N \times M)) SENSOR OUTPUT CONTROL OBSERVATION MATRIX</td>
</tr>
<tr>
<td>ER, EI, IE</td>
<td>IF ICH(2,16) IS FALSE THEN PIFEIG ASSIGNS CLOSED LOOP EIGENVALUES TO THOSE DICTATED BY ER AND EI</td>
</tr>
<tr>
<td>ERV, IEV</td>
<td>IF ICH(2,16) IS FALSE THEN PIFEIG ASSIGNS CLOSED LOOP EIGENVALUES TO THOSE DICTATED BY ERV.</td>
</tr>
<tr>
<td>GAMA</td>
<td>((N+M+L) \times M) PIF PLANT CONTROL MATRIX</td>
</tr>
<tr>
<td>GK</td>
<td>((M \times (N+M+L))) PIF GAIN FROM RICCATI EQUATION SOLUTION</td>
</tr>
<tr>
<td>HAC</td>
<td>((3 \times N)) STATE OBSERVATION PERTURBATION MATRICES</td>
</tr>
<tr>
<td>HX</td>
<td>((N \times N)) ACCELEROMETER OUTPUT STATE OBSERVATION MATRIX</td>
</tr>
<tr>
<td>PHI</td>
<td>((N+M+L) \times (N+M+L)) PIF PLANT SYSTEM MATRIX</td>
</tr>
<tr>
<td>PHICL</td>
<td>((N+M+L) \times (N+M+L)) CLOSED-LOOP PIF PLANT IN DREG AND IDEAL MODEL MATRIX IN PIF-IEG, PIFPLC</td>
</tr>
<tr>
<td>DHCP</td>
<td>((N+M+L) \times (N+M+L)) PIF DISCRETE STATE QUADRIC WEIGHT MATRIX</td>
</tr>
<tr>
<td>RMAT</td>
<td>((M \times M)) PIF DISCRETE CONTROL QUADRIC</td>
</tr>
<tr>
<td>W9</td>
<td>WORK MATRIX</td>
</tr>
<tr>
<td>X</td>
<td>((N+M+L) \times (N+M+L)) PIF DISCRETE RICCATI EQUATION SOLUTION</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MATRIX SYSTEMS</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FM, GM, HM, DM</td>
<td>QUAD SYSTEM MATRICES FOR COMMAND MODEL (CONTINUOUS MODEL)</td>
</tr>
<tr>
<td>DFM, DGM, HM, DM</td>
<td>QUAD SYSTEM MATRICES FOR COMMAND MODEL (DISCRETE MODEL)</td>
</tr>
<tr>
<td>F, G, H, D</td>
<td>QUAD SYSTEM MATRICES FOR AIRCRAFT MODEL</td>
</tr>
<tr>
<td>DF, DG, H, D</td>
<td>(CONTINUOUS MODEL)</td>
</tr>
<tr>
<td>S11, S12, S21, S22</td>
<td>CGT FEEDFORWARD MATRIX SOLUTION (SEE CGTPIF)</td>
</tr>
<tr>
<td>S31, S32, S41, S42</td>
<td>PIF REFORMATED GAINS (SEE PIFG AND DPIFS)</td>
</tr>
</tbody>
</table>
TABLE 1. (CONTINUED) PIFCGT Variable Definitions

**AFRODYNAMIC DATA**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLBOOT</td>
<td></td>
</tr>
<tr>
<td>CDROT</td>
<td></td>
</tr>
<tr>
<td>CMADOT</td>
<td></td>
</tr>
<tr>
<td>CNADOT</td>
<td></td>
</tr>
<tr>
<td>CXD</td>
<td></td>
</tr>
<tr>
<td>CYD</td>
<td></td>
</tr>
<tr>
<td>CZD</td>
<td></td>
</tr>
<tr>
<td>CLO</td>
<td></td>
</tr>
<tr>
<td>CLO</td>
<td></td>
</tr>
<tr>
<td>CMO</td>
<td></td>
</tr>
<tr>
<td>CMH</td>
<td></td>
</tr>
<tr>
<td>CMN</td>
<td></td>
</tr>
<tr>
<td>CMN</td>
<td></td>
</tr>
<tr>
<td>CLO</td>
<td></td>
</tr>
<tr>
<td>CLO</td>
<td></td>
</tr>
<tr>
<td>CMO</td>
<td></td>
</tr>
<tr>
<td>CMH</td>
<td></td>
</tr>
<tr>
<td>CMN</td>
<td></td>
</tr>
<tr>
<td>CLO</td>
<td></td>
</tr>
<tr>
<td>CLO</td>
<td></td>
</tr>
<tr>
<td>CMO</td>
<td></td>
</tr>
<tr>
<td>CMH</td>
<td></td>
</tr>
<tr>
<td>CMN</td>
<td></td>
</tr>
<tr>
<td>CLO</td>
<td></td>
</tr>
<tr>
<td>CLO</td>
<td></td>
</tr>
<tr>
<td>CMO</td>
<td></td>
</tr>
<tr>
<td>CMH</td>
<td></td>
</tr>
<tr>
<td>CMN</td>
<td></td>
</tr>
</tbody>
</table>

**OTHER VARIABLES**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>JA</td>
<td>R weight on aileron position</td>
</tr>
<tr>
<td>NWHT</td>
<td>RD weight on aileron rate</td>
</tr>
<tr>
<td>JAO</td>
<td>Aileron trim value</td>
</tr>
<tr>
<td>OEBUMA</td>
<td>T - prints out namelist AERO1</td>
</tr>
<tr>
<td>OEBUM1</td>
<td>T - prints out namelistNam1</td>
</tr>
<tr>
<td>OEBUM2</td>
<td>T - prints out namelistNam2</td>
</tr>
<tr>
<td>OEBUM3</td>
<td>T - prints out namelistNam3</td>
</tr>
<tr>
<td>OEBUM4</td>
<td>T - prints out namelistNam4</td>
</tr>
<tr>
<td>OEBUM6</td>
<td>T - prints out namelistNam6</td>
</tr>
<tr>
<td>JEBU14</td>
<td>T - prints out intermediate values in FGAERO</td>
</tr>
<tr>
<td>JEBU22</td>
<td>T - prints out intermediate values in HOCOH</td>
</tr>
<tr>
<td>JEBU26</td>
<td>T - prints out intermediate values in PIFMODL</td>
</tr>
<tr>
<td>JEBU27</td>
<td>T - prints out intermediate values in REMAT</td>
</tr>
<tr>
<td>JEBU31</td>
<td>T - prints out intermediate values in DISMOD</td>
</tr>
<tr>
<td>JEBU42</td>
<td>T - prints out intermediate values in ORMP1</td>
</tr>
<tr>
<td>JEBU43</td>
<td>T - prints out intermediate values in ORMP1</td>
</tr>
<tr>
<td>JEBU46</td>
<td>T - prints out intermediate values in DREG</td>
</tr>
<tr>
<td>JEBU48</td>
<td>T - prints out intermediate values in PIFRE</td>
</tr>
<tr>
<td>JEBU49</td>
<td>T - prints out intermediate values in PIFG</td>
</tr>
<tr>
<td>JEBU53</td>
<td>T - prints out intermediate values in DPIFS</td>
</tr>
<tr>
<td>DLT</td>
<td>EIGEN control law sampling interval</td>
</tr>
<tr>
<td>DER</td>
<td>R weight on elevator position</td>
</tr>
<tr>
<td>JER</td>
<td>RD weight on elevator rate</td>
</tr>
<tr>
<td>DFO</td>
<td>AERO elevator trim value</td>
</tr>
<tr>
<td>DREAL</td>
<td>RNINFO stability test parameter in ASYMREG, should be less than zero.</td>
</tr>
</tbody>
</table>

**Note:** All input quadratic weights are squared in CFWM when they are placed in Q, R, RD, OZ, or OQVE.
### TABLE 1. (CONTINUED) PIFCGT Variable Definitions

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRR</td>
<td>R weight on rudder position</td>
</tr>
<tr>
<td>DRRD</td>
<td>R weight on rudder rate</td>
</tr>
<tr>
<td>DRO</td>
<td>Rudder trim value</td>
</tr>
<tr>
<td>DTH</td>
<td>R weight on throttle position</td>
</tr>
<tr>
<td>DTHD</td>
<td>R weight on throttle rate</td>
</tr>
<tr>
<td>DTO</td>
<td>Throttle trim value</td>
</tr>
<tr>
<td>ECGT</td>
<td>Convergence test in solver in CGT for inverting matrix</td>
</tr>
<tr>
<td>ELOG</td>
<td>Convergence test for logx1, should be 1.0E-10</td>
</tr>
<tr>
<td>EPS</td>
<td>1.0E-09 not used</td>
</tr>
<tr>
<td>GKGGM</td>
<td>Field gain used to adjust GM so that steady state sideslip is zero if iChose = 7</td>
</tr>
<tr>
<td>GKDICP</td>
<td>Intercept gain for computing DO feedback gain value</td>
</tr>
<tr>
<td>GKDMAX</td>
<td>Minimum DO feedback gain value</td>
</tr>
<tr>
<td>GKDMSL</td>
<td>Slope gain for computing DO feedback gain</td>
</tr>
<tr>
<td>GKSPI</td>
<td>PSI error gain in iChose = 6, 15 and 16 design</td>
</tr>
<tr>
<td>GR</td>
<td>AERO gravity (ft/sec^2)</td>
</tr>
</tbody>
</table>

**ICGT(1)** = 1
- O11, O12, O21, O22 constructed
- FM, GM, HM, DM, S11, S12, S21, S22, FW, GW, A, B, C, E, S31, S32, S41, S42 must appear in calling sequence but are not used in program execution

**ICGT(2)** = 2
- S11, S21 constructed
- S21, S22, FW, GW, A, B, C, E, S31, S32, S41, S42 must appear in calling sequence but are not used in program execution

**ICGT(3)** = 3
- S11, S21, S22 constructed
- FW, GW, A, B, C, E, S31, S32, S41, S42 must appear in calling sequence but are not used in program execution

**ICGT(4)** = 4
- S31, S32, S41, S42 constructed
- FM, GM, HM, DM, S11, S12, S21, S22 must appear in calling sequence but are not used in program execution

**ICGT(5)** = 5
- Construct all feedforward matrices

**ICGT(4)** = 1
- Print out O11, O12, O21, O22 matrices

**ICGT(4)** = 2
- Print out O11, O12, O21, O22, S11, S12, S21, S22 matrices

**ICGT(1)** = 1
- Do not perform quad inverse O11, O12, O21, O22 are passed in calling sequence
**TABLE 1. (CONTINUED) PIFCGT Variable Definitions**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICH</td>
<td>EIGEN</td>
</tr>
<tr>
<td>ICHOOSE</td>
<td>RNINFO</td>
</tr>
<tr>
<td>ICHOOSE = 1</td>
<td></td>
</tr>
<tr>
<td>ICHOOSE = 3</td>
<td></td>
</tr>
<tr>
<td>ICHOOSE = 4</td>
<td></td>
</tr>
<tr>
<td>ICHOOSE = 6</td>
<td></td>
</tr>
<tr>
<td>ICHOOSE = 7</td>
<td></td>
</tr>
<tr>
<td>ICHOOSE = 9</td>
<td></td>
</tr>
<tr>
<td>ICHOOSE = 12</td>
<td></td>
</tr>
<tr>
<td>ICHOOSE = 14</td>
<td></td>
</tr>
<tr>
<td>ICHOOSE = 15</td>
<td></td>
</tr>
<tr>
<td>IOREG</td>
<td>RNINFO</td>
</tr>
<tr>
<td>IFREEG</td>
<td>EIGEN</td>
</tr>
<tr>
<td>IFREEG</td>
<td>EIGEN</td>
</tr>
<tr>
<td>IFREEP</td>
<td>EIGEN</td>
</tr>
<tr>
<td>IFREIG</td>
<td>EIGEN</td>
</tr>
<tr>
<td>IKDGM</td>
<td>RNINFO</td>
</tr>
<tr>
<td>IMAX</td>
<td>LABEL</td>
</tr>
<tr>
<td>IRUN</td>
<td>LABEL</td>
</tr>
<tr>
<td>ITFR</td>
<td>RNINFO</td>
</tr>
<tr>
<td>Variable</td>
<td>Definition</td>
</tr>
<tr>
<td>----------</td>
<td>------------</td>
</tr>
<tr>
<td>IX</td>
<td>AERO</td>
</tr>
<tr>
<td>IV</td>
<td>AERO</td>
</tr>
<tr>
<td>L</td>
<td>GLAD</td>
</tr>
<tr>
<td>LAMDA</td>
<td>RNINFO</td>
</tr>
<tr>
<td>LAMOAG</td>
<td>RNINFO</td>
</tr>
<tr>
<td>LANDAGS</td>
<td>RNINFO</td>
</tr>
<tr>
<td>M</td>
<td>GLAD</td>
</tr>
<tr>
<td>MLOG</td>
<td>RNINFO</td>
</tr>
<tr>
<td>%</td>
<td>GLAD</td>
</tr>
<tr>
<td>%M</td>
<td>GLAD</td>
</tr>
<tr>
<td>%M</td>
<td>GLAD</td>
</tr>
<tr>
<td>PBSQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>PBQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>PBHQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>PHIQ</td>
<td>AERO</td>
</tr>
<tr>
<td>PHQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>P5DQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>P5IQ</td>
<td>AERO</td>
</tr>
<tr>
<td>P5Q</td>
<td>QWHT</td>
</tr>
<tr>
<td>PQ</td>
<td>AERO</td>
</tr>
<tr>
<td>QBDQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>QBO</td>
<td>QWHT</td>
</tr>
<tr>
<td>QO</td>
<td>AERO</td>
</tr>
<tr>
<td>QDOQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>RBQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>RHO</td>
<td>AERO</td>
</tr>
<tr>
<td>RO</td>
<td>AERO</td>
</tr>
<tr>
<td>S</td>
<td>AERO</td>
</tr>
<tr>
<td>SS</td>
<td>EIGEN</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>T0DQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>THETAQ</td>
<td>AERO</td>
</tr>
<tr>
<td>TBO</td>
<td>QWHT</td>
</tr>
<tr>
<td>TOL</td>
<td>RNINFO</td>
</tr>
<tr>
<td>UBDQ</td>
<td>QWHT</td>
</tr>
<tr>
<td>J0Q</td>
<td>QWHT</td>
</tr>
<tr>
<td>UL</td>
<td>LABEL</td>
</tr>
<tr>
<td>JHAG</td>
<td>UDIR</td>
</tr>
<tr>
<td>UMR</td>
<td>EIGEN</td>
</tr>
<tr>
<td>UR</td>
<td>EIGEN</td>
</tr>
<tr>
<td>JSTP</td>
<td>UDIR</td>
</tr>
</tbody>
</table>
### TABLE 1. (CONCLUDED) PIFCGT Variable Definitions

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>USTRT</td>
<td>UDIR</td>
</tr>
<tr>
<td>UXMAG</td>
<td>LABEL</td>
</tr>
<tr>
<td>JO</td>
<td>AERO TRIM VALUE OF U</td>
</tr>
<tr>
<td>VDOTQ</td>
<td>QWHT O DWCE WEIGHTING OF VDOT</td>
</tr>
<tr>
<td>VHQ</td>
<td>QWHT Q WEIGHTING OF V</td>
</tr>
<tr>
<td>VTO</td>
<td>AERO TRIM VALUE OF VT</td>
</tr>
<tr>
<td>VTP</td>
<td>RNINFO TOTAL AIRSPEED VELOCITY IN FT/SEC**2. USED IN interp. TAIL VALUES AND AERODYNAMIC DATA ARE INTERPOLATED SO THAT AIRCRAFT MODEL IS FOUND FOR VTO = VTP.</td>
</tr>
<tr>
<td>VO</td>
<td>AERO TRIM VALUE OF V</td>
</tr>
<tr>
<td>VDOTQ</td>
<td>QWHT O DWCE WEIGHTING OF VDOT</td>
</tr>
<tr>
<td>VQ</td>
<td>QWHT Q WEIGHTING OF W</td>
</tr>
<tr>
<td>W</td>
<td>AERO TRIM VALUE OF W</td>
</tr>
<tr>
<td>XDQ</td>
<td>QWHT WEIGHTING ON XDOT POSITION</td>
</tr>
<tr>
<td>XQ</td>
<td>QWHT WEIGHTING ON X POSITION</td>
</tr>
<tr>
<td>XL</td>
<td>LABEL ARRAY CONTAINING THE LABELS FOR THE STATES</td>
</tr>
<tr>
<td>XR</td>
<td>EIGEN LOGICAL ARRAY WITH INFORMATION TO REFORMAT THE COMMAND MODEL SYSTEM MATRIX FM (NM X NM)</td>
</tr>
<tr>
<td>XS</td>
<td>HDXA LOCATION OF ACCELEROMETERS IN BODY AXIS, X-POSITION WITH REFERENCE TO CG</td>
</tr>
<tr>
<td>XXMAG</td>
<td>LABEL ARRAY CONTAINING THE SCALING VALUES FOR STATES USED IN DIPS WHEN METERS IS TRUE, AUTOMATICALLY SCALED</td>
</tr>
<tr>
<td>YDQ</td>
<td>QWHT WEIGHTING ON YDOT POSITION</td>
</tr>
<tr>
<td>YQ</td>
<td>QWHT WEIGHTING ON Y POSITION</td>
</tr>
<tr>
<td>YL</td>
<td>LABEL ARRAY CONTAINING THE LABELS FOR THE COMMANDS</td>
</tr>
<tr>
<td>YR</td>
<td>EIGEN LOGICAL ARRAY WITH INFORMATION TO REFORMAT THE OUTPUT MATRICES H,D,HM,OM (L X N), (L X M), (L X NM), (L X NM)</td>
</tr>
<tr>
<td>YS</td>
<td>HDXA LOCATION OF ACCELEROMETERS IN BODY AXIS, Y-POSITION WITH REFERENCE TO CG</td>
</tr>
<tr>
<td>YXMAG</td>
<td>LABEL ARAY CONTAINING THE SCALING VALUES FOR OUTPUTS (NOT USED)</td>
</tr>
<tr>
<td>ZCDMAX</td>
<td>RNINFO1 MULTIPLICATION FACTOR FOR CHANGING ZO MAXIMUM</td>
</tr>
<tr>
<td>ZCLMAX</td>
<td>RNINFO1 MULTIPLICATION FACTORS FOR CHANGING ZCLOSE</td>
</tr>
<tr>
<td>ZCLMIN</td>
<td>RNINFO1 MULTIPLICATION FACTORS FOR CHANGING ZCLOSE</td>
</tr>
<tr>
<td>ZCLDSE</td>
<td>RNINFO1 POSITION DIFFERENCE WHEN FULL DECELERATION BEGINS</td>
</tr>
<tr>
<td>ZDOMAX</td>
<td>RNINFO1 MAXIMUM ACCELERATION</td>
</tr>
<tr>
<td>ZDMMAX</td>
<td>RNINFO1 MAXIMUM MODEL RATE</td>
</tr>
<tr>
<td>ZFTA</td>
<td>RNINFO VELOCITY VECTOR HEADING ANGLE OF TRIMMED FLIGHT CONDITION (USUALLY 0.0)</td>
</tr>
<tr>
<td>ZETA1</td>
<td>RNINFO VELOCITY VECTOR HEADING ANGLE OF TRIMMED FLIGHT CONDITION (USUALLY 0.0)</td>
</tr>
<tr>
<td>ZIQQ</td>
<td>QWHT WEIGHTING ON ZDOT POSITION</td>
</tr>
<tr>
<td>ZIQ</td>
<td>QWHT WEIGHTING ON Z POSITION</td>
</tr>
<tr>
<td>ZS</td>
<td>HDXA LOCATION OF ACCELEROMETERS IN BODY AXIS, Z-POSITION WITH REFERENCE TO CG</td>
</tr>
<tr>
<td>ZIQ</td>
<td>QWHT Q WEIGHTING OF FIRST INTEGRATOR STATE (QZ REFORMATED USING YR)</td>
</tr>
<tr>
<td>Z2Q</td>
<td>QWHT Q WEIGHTING OF SECOND INTEGRATOR STATE (QZ REFORMATED USING YR)</td>
</tr>
<tr>
<td>Z3Q</td>
<td>QWHT Q WEIGHTING OF THIRD INTEGRATOR STATE (QZ REFORMATED USING YR)</td>
</tr>
<tr>
<td>Z4Q</td>
<td>QWHT Q WEIGHTING OF FOURTH INTEGRATOR STATE (QZ REFORMATED USING YR)</td>
</tr>
</tbody>
</table>
III. SUBROUTINE DOCUMENTATION

All subroutines used by PIFCGT are from ORACLS (ORACLIB in Appendix B) or from the PIF library (PIFLIBO in Appendix B). This chapter individually documents in alphabetical order the PIF library subroutines. The documentation follows the same format as ORACLS in Ref. 2.
A. BLOCK DATA (AEROD)

1. PURPOSE

Block data program AEROD initializes aerodynamic data in common blocks AERO and AER1. Other variables are initialized from common blocks listed in 2d below.

2. USAGE

Under the NOS 1.3 operating system, AEROD is placed in the submit file as shown in Appendix B.

a. Calling Sequence

None

b. Input Arguments

None

c. Output Arguments

None

d. Common Blocks

RNINFO, LABEL, EIGEN, GLAD, IDENTS, DEBUG, AERO, AER1, HDXA

e. Error Messages

None

f. Subroutines Employed by AEROD

None

g. Subroutines Employing AEROD

None

h. Concluding Remarks

None
B. SAMPLED-DATA REGULATOR (ALPHA AND BETA)

1. PURPOSE

The purpose of the subroutines ALPHA and BETA is to compute the following functions depending on the values of LLL and H,

If \( LLL = 2, \) and \( H = 0.0 \) then compute

\[
SUM = \int_0^{\Delta t} \int_0^T e^{ATs} ds \int_0^T e^{As} ds \, d\tau
\]  

If \( LLL = 1, \) and \( H = -1.0 \) then compute

\[
SUM = \int_0^{\Delta t} e^{AT\tau} Q \int_0^\tau e^{As} ds \, d\tau
\]

Subroutine ALPHA calls subroutine BETA as part of its execution.

2. USAGE

a. Calling Sequence

CALL ALPHA (N, A, AT, ASIG, SIG, 0, GEST, EPSLO, LLL, H, SUM, A1, SS1,  
SS, A2, BVEC, AVEC, XMLIH)

b. Input Arguments

N  Integer scalar value indicating matrix dimension; A, AT, ASIG,  
Q, SUM, A1, SS1, SS, A2, BVEC, AVEC, XMLIH - all \( N \times N \).

A  Matrix packed by column in one-dimensional array; not destroyed  
on return.

AT  Matrix packed by column in one-dimensional array; not destroyed  
on return. AT must contain the transpose of A.

ASIG, A1,  
SS1, SS,  
A2, BVEC,  
AVEC,  
XMLIH  Working space vectors of dimension at least \( N \times N \)

SIG  Real scalar indicating the sampling time \( \Delta t \).

GEST  An index for starting the test of convergence in computing the
integrals. A value of 5.0 is recommended.

EPSLO Real scalar indicating the convergence criterion in computing the integrals. A value of 0.00001 is recommended.

LLL, H An integer scalar and a real scalar, respectively, used to determine the integrals discussed in the PURPOSE section.

c. Output Arguments

SUM An N x N matrix packed by columns into a one-dimensional array containing the evaluation of the computed integrals.

d. Common Blocks

None

e. Error Messages

None

f. Subroutines Employed by ALPHA and BETA

ORACLS - MULT, UNITY

g. Subroutines Employing ALPHA and BETA

PIFLIB - QRMHAT

h. Concluding Remarks

None
C. SOLUTION OF $A*Y*B-X=C$ (AXBMXC)

1. PURPOSE

Subroutine AXBMXC solves the algebraic matrix equation

$$A*X*B - X = C \quad (1)$$

First the matrix $A$ is transformed to the lower real SCHUR form using the unitary transformation $U$. Next, the matrix $B$ is transformed to upper real SCHUR form using the unitary transformation $V$. The matrix $C$ is transformed using $U^T * C * V$. Equation 1 reduces to

$$U^T A U * (U^T X V) * (U^T B V) - (U^T X V) = U^T C V \quad (2)$$

or

$$A_s * X_s * B_s - X_s = C_s \quad (3)$$

The subroutine SHRSOL is used to solve Eq. 3 for $X_s$. The solution $X$, is reconstructed for $X_s$ as follows

$$X = U^T X_s * V^T \quad (4)$$

It is assumed that

$$\lambda_i^A \lambda_j^B \neq 1 \quad (i=1,2,\ldots,N; \quad j=1,2,\ldots,M) \quad (5)$$

where $\lambda_i^A$ and $\lambda_j^B$ are eigenvalues of $A$ and $B$ respectively. If Eq. 5 is true, Eq. 1 has a unique solution for $X$.

2. USAGE

a. Calling Sequence

CALL AXBMXC (A,U,B,V,C,N,NA,NU,M,NB,NV,NC,EPSA,EPSB,FAIL)

b. Input Arguments

$A,B,C$ Matrices stored in two-dimensional array. First dimensions
are NA, NB, and NC, respectively. Second dimensions are at least N, M, and N, respectively. Destroyed upon return.

N  Order of the matrix A

M  Order of the matrix B

NA, NB, NC,
NU, NV  Maximum first dimensions of the matrices A, B, C, U, and V, respectively, as given in the dimension statement of the calling program. It is required that NA .GT. N+1, NB .GT. M+1, NC .GT. N+1, NU .GT. N+1, and NV .GT. M+1

EPSA  Criterion for zeroing elements in the matrix A in the subroutine SCHUR. A(I,J) is considered effectively 0 if it is less than EPSA times the infinity norm of the upper Hessenberg form of A.

NOTE: If EPSA .LT. 0 it is assumed that A is already in lower real SCHUR form and U is the appropriate transformation matrix.

EPSB  Criterion for zeroing elements in the matrix B in the subroutine SCHUR. B(I,J) is considered effectively 0 if it is less than EPSB times the infinity norm of the upper Hessenberg form of B.

NOTE: If EPSB .LT. 0 it is assumed that B is already in upper real SCHUR form and V is the appropriate transformation matrix.

c. Output Arguments

FAIL  Integer flag indicating success of reduction of A and B to lower and upper real SCHUR form. If FAIL is not zero, SCHUR was unsuccessful.

U  Two-dimensional array containing the orthogonal matrix which reduces A to lower real SCHUR form. First dimension is NU.
Second dimension is at least $N$.

$V$ Two-dimensional array containing the orthogonal matrix which reduces $B$ to upper real SCHUR form. First dimension is $N\times V$. Second dimension is at least $M$.

$A, B$ Two-dimensional arrays containing the original $A$ and $B$ matrices transformed to lower and upper real SCHUR form, respectively.

$C$ Two-dimensional array containing the solution, $X$.

d. COMMON Blocks

None

e. Error Messages

None

f. Subroutines Employed by AXBMXC

ORACLS - BCKMLT, HSHLDR, SCHUR, SYSSLV

PIFLIB - SHRSOL

g. Subroutines Employing AXBMXC

PIFLIB - SOLVER

h. Concluding Remarks

It is recommended that the program SOLVER, which calls AXBMXC, be used to solve Eq. 1. This subroutine is a generalization of the algorithm reported in Ref. 3.
D. CREATE WEIGHTING MATRICES (CFWM)

1. PURPOSE

The subroutine CFWM constructs the diagonal weighting matrices used in the PIF control law quadratic cost function. The state weights UBQ, WBQ, QBQ, THQ, XIQ, ZIQ, VHQ, RBQ, PBQ, PHQ, PSQ, YIQ are squared and placed along the Q matrix diagonal. The control weights DTH, DER, DA, and DRR are squared and placed along the R matrix diagonal. The control rate weights DTHD, DERD, DARD, DRRD are squared and placed along the RD matrix diagonal. The integrator weights Z1Q, Z2Q, Z3Q and Z4Q are squared and placed along the QZ matrix diagonal. The state rate weights UBDQ, WBDQ, QBDQ, THDQ, XIDQ, ZIDQ, VBDQ, RBDQ, PBDQ, PHDQ, PSDQ and YIDQ are squared and placed along the W9(LW1) diagonal. The following matrix is constructed

\[
\begin{bmatrix}
   Q_1 & A_M^T \\
   A_M & R_1
\end{bmatrix} =
\begin{bmatrix}
   F^T \\
   G^T
\end{bmatrix} W9(LW1) \begin{bmatrix} F & G \end{bmatrix}
\]

The Q1 matrix is added to Q and the R1 matrix is added to R completing the weighting matrix constructions. F and G are the continuous-time plant system matrix and control effect matrix, respectively.

2. USAGE

a. Calling Sequence

CALL CFWM(F, NF, G, NG, Q, NQ, R, NR, AM, NAM, RD, NRD, QZ, NQZ, W9, NW9DIM, DEBUG)

b. Input Arguments

F, G Matrices packed by columns into one-dimensional arrays; not destroyed upon return.
Two dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N. Integer values must be: NF(12, 12), NG(12, 4), NQ(12, 12), NR(4, 4), NAM(4, 12), NRD(4, 4), NQZ(12, 12).

Work matrix packed by columns into a one-dimensional array. Dimension must be at least 4 * ((NF(1) + NG(2))**2 + 1).

Logical scalar indicating the following:

TRUE: Print out debug information
FALSE: Do not print out debug information

Integer scalar containing the maximum dimension of the W9 matrix.

c. Output Arguments

Q, R, AM,
RD, QZ

Weighting matrices packed by columns into one dimensional arrays.

d. COMMON Blocks

QWHT

e. Error Messages

If the W9 matrix is insufficiently large the message "THE W9 MATRIX IS SMALL" is printed and the program stops. If DEBUG is true then Q, R, RD, QZ, and W9(LW1) (QDVE) are printed after being diagonally constructed. After the state rate construction and assimilation Q, R, RD, QZ, and W9(LW1) (QDVE) are printed again along with AM.
f. Subroutines Employed by CFWM

ORACLS - JUXTC, ADD, PRNTTL, DIAGPUT, PRNT, MATBA, EXTR

g. Subroutines Employing QRMHAT

PIFLIB - PIFCGT

h. Concluding Remarks

None
E. FEEDFORWARD MATRICES (CGTPIF)

1. PURPOSE

Subroutine CGTPIF computes the feedforward matrices $S_{11}$, $S_{12}$, $S_{21}$, $S_{22}$, $S_{31}$, $S_{32}$, $S_{41}$, and $S_{42}$. The subroutine determines the feedforward matrices by solving the following matrix equation

$$
\begin{bmatrix}
(F-ssI) G & S_{11} & S_{12} & S_{31} & S_{32} \\
H & D & S_{21} & S_{22} & S_{41} & S_{42}
\end{bmatrix}
= \begin{bmatrix}
S_{11}(F-ssI) & S_{11}G & S_{31}(F-ssI)-AS_{31} & G_w & B \\
H_m & D_m & -C & -E
\end{bmatrix} \tag{1}
$$

The feedforward matrix equation is valid for both continuous-time models and discrete-time models. The scalar $ss$ in Eq. 1 is chosen by the designer under the following restrictions. $ss$ cannot equal a transmission zero of the plant. If the dynamical models are representations of continuous-time systems and $S_{12}$, $S_{22}$, $S_{32}$, or $S_{42}$ is to be determined, then $ss$ must be zero. If the dynamical models are representations of discrete-time systems and $S_{12}$, $S_{22}$, $S_{32}$ or $S_{42}$ is to be determined then $ss$ must be one.

The plant, disturbance, and model matrices in the feedforward matrix equations are assumed to satisfy the following dynamical equations

**CONTINUOUS TIME**

**PLANT**

$$\begin{align*}
\dot{x} &= Fx + Gu + Aw + Bs \\
y &= Hx + Du + Cw + Es
\end{align*} \tag{2}$$

**DISTURBANCE TO BE REJECTED**

$$\dot{w} = Fw + Gs \tag{4}$$
MODEL TO BE FOLLOWED

\[ \dot{x}_m = F_{m\times m}x_m + G_{m\times m}u_m \]  
\[ y_m = H_{m\times m}x_m + D_{m\times m}u_m \]  

DISCRETE TIME

PLANT

\[ x_{k+1} = F_{n\times n}x_k + G_{n\times m}u_k + A_{n\times n}w_{k} + B_{n\times n}s_k \]  
\[ y_k = H_{n\times m}x_k + D_{n\times m}u_k + C_{n\times n}w_k + E_{n\times n}s_k \]  

DISTURBANCE TO BE REJECTED

\[ w_{k+1} = F_{n\times n}w_k + G_{n\times m}s_k \]  

MODEL TO BE FOLLOWED

\[ x_{m,k+1} = F_{m\times m,k}x_{m,k} + G_{m\times m,k}u_{m,k} \]  
\[ y_{m,k} = H_{m\times m,k}x_{m,k} + D_{m\times m,k}u_{m,k} \]  

The program begins by forming the quad partition matrix on the left side of Eq. 1. If the number of outputs in \( y \) equal the number of controls in \( u \), the program attempts to invert the quad partition matrix using GELIM. If the quad partition matrix is not square or GELIM encountered an error in attempting to invert the quad partition matrix, CGTPIF attempts to find the pseudo-inverse of the quad partition matrix using singular value decomposition (SVDDEC). If SVDDEC also encounters an error, a message is printed and the program continues.
After obtaining the inverse (or pseudo-inverse) of Eq. 1, it can be expressed as:

\[
\begin{bmatrix}
(F - ssI) & G \\
H & D
\end{bmatrix}^{-1} = \begin{bmatrix}
\Omega_{11} & \Omega_{12} \\
\Omega_{21} & \Omega_{22}
\end{bmatrix}
\]

(12)

Eq. 1 is separated into eight algebraic matrix equations,

\[
S_{11} = \Omega_{11} S_{11} (F_m - ssI) + \Omega_{12} H_m
\]

(13)

\[
S_{12} = \Omega_{11} S_{11} G_m + \Omega_{12} D_m
\]

(14)

\[
S_{21} = \Omega_{21} S_{11} (F_m - ssI) + \Omega_{22} H_m
\]

(15)

\[
S_{22} = \Omega_{21} S_{11} G_m + \Omega_{22} D_m
\]

(16)

\[
S_{31} = \Omega_{11} S_{31} (F_w - ssI) + \Omega_{11} A - \Omega_{12} C
\]

(17)

\[
S_{32} = \Omega_{11} S_{31} G_w - \Omega_{11} B - \Omega_{12} E
\]

(18)

\[
S_{41} = \Omega_{21} S_{31} (F_w - ssI) + \Omega_{21} A - \Omega_{22} C
\]

(19)

\[
S_{42} = \Omega_{21} S_{31} G_m - \Omega_{21} B - \Omega_{22} E
\]

(20)

Equations 13 and 17 are solved using SOLVER. Equations 14, 15, 16, 18, 19, and 20 are solved using matrix multiplication.

2. USAGE

a. Calling Sequence


b. Input Arguments
Matrices packed by columns in one-dimensional arrays; not 
destroyed upon return.

Integer scalar values indicating matrix dimensions:

F-NxN, G-NxM, H-LxN, D-LxM, A-NxNW, B-NxMW, C-LxNW, E-LxMW,
FW-NWxNW, GW-NWxMW, FM-NMxNM, GM-NMxMM, HM-LxNM, DM-LxMM

Real scalar indicating convergence criteria for iterative 
refinement in SOLVER. Recommend 1.0E-10 for CDC machine.

Integer scalar indicating maximum number of iterative 
refinements in SOLVER. Recommend 10.

Real scalar indicating convergence criteria in SOLVER for 
inverting a matrix. Recommend 1.0E-10 for CDC machine.

Five dimensional integer option vector:

ICGT(1) = 1 011, 012, 021, 022 constructed.

FM,GM,GM,DM,S11,S12,S22,FW,GW,A,B,C,E,
S31,S32,S41,S42 must appear in calling sequence 
but are not used in program execution.

ICGT(1) = 2 S11, S21 constructed.

S21,S22,FW,GW,A,B,C,E,S31,S32,S41,S42 must 
appear in calling sequence but are not used in 
program execution.
ICGT(1) = 3  S11, S12, S21, S22 constructed.

FW, GW, A, B, C, E, S31, S32, S41, S42 must appear in
calling sequence but are not used in program
execution.

ICGT(1) = 4  S31, S32, S41, S42 constructed.

FM, GM, HM, DM, S11, S12, S21, S22 must appear in
calling sequence but are not used in program
execution.

ICGT(1) = 5  Construct all feedforward matrices.

ICGT(4) = 1  Print out 011, 012, 021, 022 matrices.

ICGT(4) = 2  Print out all matrices.

ICGT(5) = 1  Do not perform quad inverse. 011, 012, 021,

022, are passed in calling sequence.

SS  Scalar option

SS = 1.0 if ICGT(1) .GT. 2 and plant, model and disturbance
are discrete time models

SS = 0.0 if ICGT(1) .GT. 2 and plant, model and disturbance
are continuous time models

SS = Any number if ICGT(1) .LE. 2 except plant transmission
zeroes.

W1, W2, W3
W4, W5  Working space vectors of dimension at least:

W1 = \((2*(N+MAX(L,M)+2))**2, W2, W3, W4, W5 = \((N+MAX(L,M)+2))**2
c. Output Arguments

011,012,
021,022 Matrices packed by columns into one-dimensional arrays.

Upon normal return and depending on ICGT, the matrices contain the partitioned quad inverse

\[
\begin{bmatrix}
\Omega_{11} & \Omega_{12} \\
\Omega_{21} & \Omega_{22}
\end{bmatrix} =
\begin{bmatrix}
011 & 012 \\
021 & 022
\end{bmatrix}
\]

Dimensions of the matrices are at least:

011 - N**2, 012 - N*L, 021 - M*N, 022 - M*L.

S11,S12,
S21,S22,
S31,S32,
S41,S42 Matrices packed by columns into one-dimensional arrays.

Upon normal return and depending on ICGT, the matrices contain the feedforward matrices. Dimensions of the matrices are at least (N+MAX(NM,NW)+2)**2.

NS11,NS12,
NS21,NS22,
NS31,NS32,
NS41,NS42 Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N. Example,

\[
\text{NS11}(1) = N
\]
\[
\text{NS11}(2) = NM
\]

d. COMMON Blocks

None

e. Error Messages
If ICGT(1) .GT. 2 and SS .NE. 1 the message "???????????? SS IS INCORRECT IN CGT ????????????? BUT PROGRAM CONTINUES SS = ________." is printed.

If the plant is a continuous-time model and SS = 0.0, then the error should be ignored. CGTPIF cannot determine if the dynamic models are continuous or discrete and operates under the implicit assumption that the models are discrete.

If GELIM or SNVDEC encounter an error the message "SNVDEC ERROR IN CGT IERR = ________" is printed.

f. Subroutines Employed by CGTPIF

ORACLS - MULT, NULL, SCALE, SUBT, UNITY, ADD, NORMS, BCKMLT, HSHLDR,
SCHUR, SYSSLV, GELIM, DETFAC
PIFLIB - AXBMXC, SHRSOL, PRNT, PRNTITL.

g. Subroutines Employing CGTPIF

PIFLIB - PIFCGT, PIFPLC

h. Concluding Remarks

The theory and use of feedforward matrices are discussed in Refs. 4 and 5.
F. CLOSED-LOOP d MODEL (CLDMOD)

1. PURPOSE
Subroutine CLDMOD propagates the dynamic command model for the APR GS autopilot mode. The subroutine is similar to the APR GS command model in the flight computer code. A discussion of the equations used in CLDMOD is given in Ref. 1.

2. USAGE
a. Calling Sequence
CALL CLDMOD (Dl, DO, LAMDMAX, LAMIN, DEBUG, LAMDAGS, VTP, LAMC, GKDSL, GKD, GKDF, GKDMIN)

b. Input Arguments
Dl, DO  k and k-1 commanded d perpendicular position of aircraft from glideslope. Changed upon return.
LAMDMAX The maximum allowed flight path angle change in DELT seconds (ZDDMAX in input NAMELIST)
DELT The control law sampling interval
LAMAX The maximum allowed flight path angle. (ZCLMAX in input NAMELIST)
LAMIN The minimum allowed flight path angle. (ZCLMIN in input NAMELIST)
DEBUG Logical scalar indicating the following:
TRUE: Print out debug information
FALSE: Do not print out debug information
LAMDAGS The glideslope angle (positive)
VTP The total airspeed in ft/sec
LAMC The commanded model dynamics flight path angle at k-1.
   Changed upon return.

GKDSLPG The slope gain for computing the DO feedback gain.

GKDICP The intercept gain for computing the DO feedback gain.

GKDMAX The maximum DO feedback gain value.

GKDMIN The minimum DO feedback gain value.

c. Output Arguments

DL, DO k+1 and k commanded d perpendicular position of aircraft
   from glideslope.

LAMC The commanded model dynamics flight path angle at k.

d. COMMON Blocks

None

e. Error Messages

The first pass through the subroutine causes GK1, LAMAX, LAMIN and
LAMDAGS to be printed out. If DEBUG is TRUE, DL, DO, LAMDAGS, GK1,
and LAMC are printed out during the course of the simulation.

f. Subroutines Employed by CLDMOD

None

g. Subroutines Employing CLDMOD

PIFLIB - DPIFS

h. Concluding Remarks

None
G. CLOSED-LOOP Y MODEL (CLYMOD)

1. PURPOSE
Subroutine CLYMOD propagates the dynamic command model for the APR LOCI, APR LOCR, and APR LOCP autopilot modes. The subroutine is similar to the APR LOC command model propagation model in the flight computer code. A discussion of the equations used in CLYMOD is given in Ref. 1.

2. USAGE

a. Calling Sequence

CALL CLYMOD (Y1, Y0, PS11, PS10, YC, PSIC, AK2, ROLLCM, GKDSLP, GKDICP, GKDMAX, GKDMIN, DELT, DEBUG RDCMAX, VTP, ROLL)

b. Input Arguments

Y1, Y0 k and k-1 commanded Y position of aircraft from localizer beam center line. Changed upon return.

PS11, PS10 k and k-1 commanded yaw position of the aircraft relative to the runway. Changed upon return.

YC The commanded value of y which is set to 0.0.

PSIC The maximum allowed intercept angle, usually 45 deg.

AK2 The yaw error feedback gain for the roll angle command.

ROLLCM The maximum allowed value of the roll angle command during capture (ZDDMAX in input NAMELIST)

GKDLP The slope gain for computing the YO feedback gain.

GKDICP The intercept gain for computing the YO feedback gain.

GKDMAX The maximum YO feedback gain value.
GKDMIN  The minimum YO feedback gain value.
DELT    The control law sampling interval.
DEBUG   Logical scalar indicating the following:
        TRUE:  Print out debug information
        FALSE: Do not print out debug information
RDCMAX  The maximum allowed roll angle change in DELT sec (ZCDMAX in input NAMELIST)
VTP     The total airspeed in ft/sec
ROLLC   The commanded value of roll angle at k-1. Changed upon return.

  c. Output Arguments

  YI, YO  k+1 and k commanded y position of aircraft from beam center line.
  PSII, PSIO k+1 and k commanded yaw position of aircraft.
  ROLL   The commanded value of roll angle at k.

  d. COMMON Blocks

  None

  e. Error Messages

  The first pass through the subroutine causes PSIC, `AK2, ROLLCM and RDCMAX to be printed out. If DEBUG is TRUE, PSICM, YO, GK2, PSIO, ROLL, Xl, GKI and PSID are printed out during the course of the simulation.

  f. Subroutines Employed by CLYMOD

  None

  g. Subroutines Employing CLYMOD

  PIFLIB - DPIFS

  h. Concluding Remarks

  None
H. CLOSED-LOOP $\psi$ AND Z MODEL (CLXMOD)

1. PURPOSE

Subroutine CLXMOD propagates the dynamic command model for the HDG SEL and ALT SEL autopilot modes. The subroutine is similar to the HDG SEL and ALT SEL command model propagation models in the flight computer code. A discussion of the equations used in CLXMOD is given in Ref. 1.

2. USAGE

a. Calling Sequence

CALL CLXMOD (Zl, ZO, ZD1, ZD0, ZC1, ZCD, AK1, AK2, ZDDMAX, ACLOSE, ZDOMAX, DELT, ZCLMAX, ZCLMIN, ZCDEAD, DEBUG, ZCDMAX, ICHOOSE, VTP0

b. Input Arguments

Zl, ZO  k and k-1 commanded $z(\psi)$ position of aircraft
ZD1, ZD$\phi$  k and k-1 commanded $z(\phi)$ of aircraft.
ZC1  Commanded value of $z(\psi)$ position the model is required to track.
ZCD  Commanded value of $z(\phi)$, usually 0.0.
AK1, AK2  Position and velocity feedback gains for the model
ZDDMAX  The maximum allowed model acceleration ($\ddot{z}$ or $\dot{\phi}$)
DELT  The sampling interval of the control law
DEBUG  Logical scalar indicating the following:
        TRUE: Print out debug information
        FALSE: Do not print out debug information
ICHOSE  Integer scalar indicating the desired autopilot design (See HDCON)
VTP  The total airspeed in ft/sec.

c. Output Arguments

Zl, ZO  k+1 and k commanded z (ψ) position of aircraft

ZDl, ZDO  k+1 and k commanded z (ϕ) of aircraft

d. COMMON Blocks

None

e. Error Messages

The first pass through the subroutine causes AK1, AK2, ZDDMAX, and ZCLOSE to be printed out. If DEBUG is TRUE then ZC1, ZO, ZDG, ZD0 ZDD, Xl, GK1, and ZDOMAG are printed out.

f. Subroutines Employed by CLXMOD

None

g. Subroutines Employing CLXMOD

PIFLIB - DPIFS

h. Concluding Remarks

None
I. COMBINE MATRICES (CORNER)

1. PURPOSE
Subroutine CORNER inserts one matrix into another such that they have a common corner.

2. USAGE
a. Calling Sequence
CALL CORNER (A,NA,B,NB,IWHERE)

b. Input Arguments
A Matrix packed by columns; destroyed upon return.
B. Matrix packed by columns; not destroyed upon return.
NA,NB Two-dimensional vectors giving number of rows and columns of respective matrices, for example:
NA(1) = number of rows of A
NA(2) = number of columns of A
Not destroyed upon return.
IWHERE Option for combination of the two matrices: 1 use lower left corner as common, 2 use upper right corner as common.

c. Output Arguments
A Matrix packed by columns containing A & B combined.

d. COMMON Blocks
None

e. Error Messages
If either NA(1) < NB(1) or NA(2) < NB(2), or NA(1) x NA(2) < 1, or
NB(1) x NB(2) < 1, the message "DIMENSION ERROR IN CORNER NA = _____ _____,
NB = _____ _____" is printed and the program is returned to the calling point.
f. Subroutines Employed by CORNER

None

g. Subroutines Employing CORNER

PIFLIB - PIFMODL, QRMP1, QRMHAT, PIFG, FGAERO, PIFEIG

h. Concluding Remarks

Matrix B replaces a portion of A. Therefore A must be as big or bigger than B. This is essentially an overlaying process. If IWHERE is 1, B is positioned such that the lower left corner element of A is replaced with the lower left corner of B and so forth. IWHERE of 2 indicates that the upper right corner of A and B are made equal. If B is smaller than A, one corner of B will occur part way into A. See Diagram below.

\[ \text{IWHERE} = 1 \]

\[
\begin{bmatrix}
  \text{---} & A \\
  \mid & \mid \\
  \text{---} & \text{---}
\end{bmatrix}
\]

\text{B placed here}

\[ \text{IWHERE} = 2 \]

\[
\begin{bmatrix}
  A \\
  \text{---} & \text{---}
\end{bmatrix}
\]
J. DIAGONAL PARTITION (DIAGPAR)

1. PURPOSE

To combine two matrices such that they become the top left and bottom right matrices of a quad partition matrix. The top right and bottom left matrices are set to zero in the quad partition matrix.

2. USAGE

a. Calling Sequence

CALL DIAGPAR (A,NA,B,NB,C,NC)

b. Input Arguments

A,B Matrices packed by columns in one-dimensional arrays; not destroyed upon return.

NA,NB Two-dimensional vectors giving the number of rows and columns of respective matrices; for example:

NA(1) = Number of rows of A
NA(2) = Number of columns of A
Not destroyed upon return.

c. Output Arguments

C Matrix packed by columns in a one-dimensional array. Upon return A becomes the upper left hand matrix and B becomes the lower left hand matrix within the quad partition matrix C. If NB(1) = 0, then C = A.

NC Two dimensional vector: Upon return,

NC(1) = NA(1) + NB(1)
NC(2) = NA(2) + NB(2)
unless NB(1) = 0, then
NC(1) = NA(1)
NC(2) = NA(2)
d. COMMON Blocks

None

e. Error Messages

None

f. Subroutines employed by DIAGPAR

ORACLS - NULL and EQUATE.

g. Subroutines employing DIAGPAR

PIFLIB - PIFMODL, PIFG, FGAERO, PIFEIG.

h. Concluding Remarks

\[
C = \begin{bmatrix}
A & 0 \\
-1 & -1 \\
0 & B
\end{bmatrix}
\]
K. FORM A DIAGONAL MATRIX (DIAGPUT)

1. PURPOSE

Places elements of a vector along the diagonal of a square matrix.

2. USAGE

a. Calling Sequence

CALL DIAGPUT (A,NA,B,NB)

b. Input Arguments

A    Vector of elements to be placed on diagonal of output matrix.
NA   Number of elements in A.

c. Output Arguments

B    Matrix with diagonal elements from A packed by columns in one
dimensional array, off diagonal elements are zero.
NB   Two-dimensional vector containing number of rows and columns in B.
       NB(1) = NA
       NB(2) = NA

d. COMMON blocks

None

e. Error Messages

None

f. Subroutines employed by DIAGPUT:

ORACLS - NULL

g. Subroutines employing DIAGPUT:

PIFLIB - CFWM, PGAERO, PIFEIG.

h. Concluding Remarks

Example

A = \[
    a, b, c, d, e \\
\]

B = \[
    a, 0, 0, 0, 0 \\
    0, b, 0, 0, 0 \\
    0, 0, c, 0, 0 \\
    0, 0, 0, d, 0 \\
    0, 0, 0, 0, e
\]
L. DIMENSIONS (DIMSS)

1. PURPOSE

The subroutine DIMSS loads the integer dimensions into the two dimensional vectors.

2. USAGE

a. Calling Sequence

CALL DIMSS (N, M, L, NM, MM, NF, NQ, NAM, NR, NG, NH, ND, NFM, NGM, NHM, NDM, NPH, NGA, NQH, NRH, NAMH, NGK, NX, NPHC, NGKQ, NGKI, NQZ)

b. Input Arguments

N, M, L,
NM, MM Integer scalars indicating matrix dimensions. N is the number of plant states, M is the number of plant controls, L is the number of plant commands, NM is the number of model states and MM is the number of model controls.

c. Output Arguments

NF, NQ, NAM, NR,
NH, NG, ND, NFM,
NGM, NHM, NDM,
NPH, NGA, NQH,
NRH, NAMH, NGK,
NX, NPHC, NGKQ,
NGKI, NQZ Two dimensional integer vectors holding the number of rows and columns of the matrix shown after the letter N.

d.COMMON Blocks

None
e. Error Messages
None

f. Subroutines Employed by DIMSS

PIFLIB - PRNTITL

PIFLIB - PIFCGT

h. Concluding Remarks
None
M. DISCRETE MODEL CONSTRUCTION (DISCMOD)

1. PURPOSE

Subroutine DISCMOD transforms the continuous-time representations of the plant

\[ \dot{x} = Fx + Gu \]  

and model

\[ \dot{x}_m = F_m x + G_m u_m \]  

to the equivalent discrete representations

\[ x_{k+1} = D_F x_k + D_G u_k \]  

\[ x_{m,k+1} = D_{FM} x_{m,k} + D_{GM} u_{m,k} \]

The controls \( u_k \) and \( u_{m,k} \) are assumed constant over the uniform sampling interval, \( \Delta t \). The equations, using the plant dynamics as an example, are:

\[ D_F = e^{F \Delta t} \]  

\[ D_G = \left[ \int_0^{\Delta t} e^{F s} ds \right] G \]

2. USAGE

a. Calling Sequence

CALL DISCMOD (F, NF, G, NG, FM, NFM, GM, NGM, DF, DG, DFM, DGM, W9, NW9DIM, DELT, DEBUG, ICH)
b. Input Arguments

F, G, FM,
GM Matrices packed by columns into one-dimensional arrays.
Not destroyed upon return.
NF, NG,
NFM, NGM Two dimensional integer vectors holding the number of rows
and columns of the matrix shown after the letter N.
W9 Work matrix packed by columns into a one-dimensional array.
Dimension must be at least 2*(NF(1) ** 2 + 1)
ICH Two-dimensional logical matrix dimensioned 10 x 15. Sub-
routines in PIFCGT are called depending on the TRUE or
FALSE status of elements in ICH.
DEBUG Logical scalar indicating the following:
TRUE: Print out debug information
FALSE: Do not print out debug information
DELT Real scalar containing the sampling time.
NW9DIM Integer scalar containing the maximum dimension of the W9
matrix.
c. Output Arguments
DF, DG,
DFM, DGM Matrices packed by columns into one dimensional arrays.
If ICH (3,1) is TRUE then the matrices contain the discrete
plant representations. If ICH (3,1) is FALSE then DF,
DG, DFM, DGM are set equal to F, G, FM, GM.
d. COMMON Blocks
None
e. Error Messages
If DEBUG is TRUE then DFM, DGM, DF, DG are printed.
f. Subroutines Employed by DISCMOD

ORACLS - NORMS, EXPINT, UNITY, EQUATE, SCALE, NULL, ADD, LNCNT, MAXEL
PIFLIB - PRNTITL, PRNT

g. Subroutines Employing DISCMOD

PIFLIB - PIFCGT

h. Concluding Remarks

The ORACLS subroutine EXPINT cannot compute DFM if FM is the null matrix. Special logic is used in DISCMOD to circumvent the problem in EXPINT.
The purpose of DPIFS is to simulate the linear PIF control law tracking the output of the chosen autopilot model being designed. The output of DPIFS is a step by step print out of the simulation. The discrete equations simulated by DPIFS are as follows:

PLANT DYNAMICS

\[ x_{k+1} = DF x_k + DG u_k \]  

PLANT MEASUREMENTS

\[ z_k = HX x_k + DX u_k \]

MODEL DYNAMICS

\[ x_{m,k+1} = DFM x_{m,k} + DGM u_{m,k} \]

OR

CLXMLD, CLYMOD, or CLXMOD is called depending on the value of ICLOSE

PIF CONTROL LAW

\[ e_k = z_k - S_{m,m} x_{k-1} \]

\[ y_k = HX x_k + Du_k \]

\[ y_{m,k} = H x_{m,m,k} + Du_{m,k} \]

\[ v_k = (I - \Delta t C_4) v_{k-1} - C_3 (e_k - e_{k-1}) \]

\[ - \Delta t C_4 (H x_{k-1} + Du_{k-1} - H x_{m,m,k-1} - Du_{m,m,k}) + E_1 (u_{m,k+1} - u_{m,k}) \]

\[ u_{k+1} = u_k + \Delta t v_k + S_{21} (x_{m,k+1} - x_{m,k}) \]
RELATIONSHIP BETWEEN EQUATION VARIABLES AND DPIFS NOTATION

<table>
<thead>
<tr>
<th>EQUATION VARIABLE</th>
<th>DPIFS NOTATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_{11}$</td>
<td>S11</td>
</tr>
<tr>
<td>$I-\Delta t \cdot C_4$</td>
<td>S31</td>
</tr>
<tr>
<td>$S_{21}$</td>
<td>S21</td>
</tr>
<tr>
<td>$-C_3$</td>
<td>S32</td>
</tr>
<tr>
<td>$-\Delta t \cdot C_4$</td>
<td>S41</td>
</tr>
<tr>
<td>$E_1$</td>
<td>S42</td>
</tr>
<tr>
<td>$x_{m,k+1}$</td>
<td>XM</td>
</tr>
<tr>
<td>$x_{m,k}$</td>
<td>XM1</td>
</tr>
<tr>
<td>$x_{m,k-1}$</td>
<td>XM2</td>
</tr>
<tr>
<td>$u_{m,k+1}$</td>
<td>YS(1, IP+2)</td>
</tr>
<tr>
<td>$u_k$</td>
<td>PU(1, IP)</td>
</tr>
<tr>
<td>$y_k$</td>
<td>PY(1, IP)</td>
</tr>
<tr>
<td>$y_{m,k}$</td>
<td>YSS(1, IP)</td>
</tr>
<tr>
<td>$y_{k-1}$</td>
<td>YK1</td>
</tr>
<tr>
<td>$z^*<em>k = S</em>{11}x_{m,k} + S_{12}u_{m,k+1}$</td>
<td>YX(1, IP)</td>
</tr>
<tr>
<td>$z_k$</td>
<td>PX(1, IP)</td>
</tr>
<tr>
<td>$e_k$</td>
<td>YXXK</td>
</tr>
<tr>
<td>$e_{k-1}$</td>
<td>EK1</td>
</tr>
<tr>
<td>$u^*<em>k = S</em>{21}x_{m,k} + S_{22}u_{m,k+1}$</td>
<td>YU(1, IP)</td>
</tr>
<tr>
<td>$e_{k-1}$</td>
<td>EK2</td>
</tr>
<tr>
<td>$y_k$</td>
<td>VK1</td>
</tr>
<tr>
<td>$y_{k-1}$</td>
<td>VK2</td>
</tr>
<tr>
<td>$y_{m,k}$</td>
<td>YK</td>
</tr>
<tr>
<td>$y_{k-1}$</td>
<td>YMK</td>
</tr>
<tr>
<td>$y_{m,k-1}$</td>
<td>YM1</td>
</tr>
<tr>
<td>$z_{k-1}$</td>
<td>YM2</td>
</tr>
<tr>
<td></td>
<td>PX(1, IP-1)</td>
</tr>
</tbody>
</table>
The simulation begins by setting almost all state variables to zero, the initial condition starting point. If ICHOSE = 14 (APR GS), the initial value for d is loaded in appropriate locations. If ICHOSE = 9 (APR LOCI) or ICHOSE = 15 (APR LOCR or APR LOCOP), the initial value for y, and the intercept yaw angle are loaded in appropriate locations. The initial values for the commands are in UMAG(1,1) and UMAG(2,1), as discussed in HDCON. The starting time at which the command is changed from 0.0 to the UMAG value is in USTRT(1,1) and USTRT(2,1). The stopping time at which the command is returned to 0.0 is in USTP(1,1) and USTP(2,1). The simulation is from 0.0 seconds to (IMAX-2) * DELT seconds. All the data from the simulation are stored in two dimensional matrices (PX, YX, PU, YU, PY, YSL). The dimensions of these matrices limit the value of IMAX (300 for the NAVION autopilot designs).

2. USAGE

a. Calling Sequence

CALL DPIFS (DEBUG, METERS, DEBUG1, ICHOSE, ZDDMAX, ZCLOSE, ZDOMAX, ZCLMIN, ZCDMAX, VTP, LAMDA, LAMDAGS, GKDSLP, GKDICP, GKDMAX, GKDMIN)

b. Input Arguments

ZDDMAX, ZCLOSE,
ZDOMAX, AK1,
AK2, ZCLMAX,
ZCLMIN, LAMDA,
ZCDMAX, LAMDAGS,
GKDSLP, GKDICP,
GKDMAX,
GKDMIN Autopilot parameters specified by the designer in NAMELIST.
Definitions for these variables are provided in CLXMOD, CLYMOD, and CLDMOD documentation.

DEBUG,

DEBUG1 Logical scalars indicating the following:

TRUE: Print out debug information.

FALSE: Do not print out debug information.

DEBUG is used in the subroutines CLYMOD, CLXMOD, CLDMOD.

DEBUG1 is not used.

METERS If METERS is TRUE then state variables that have units of feet are converted to units of meters before print out and plotting.

ICHOSE Integer scalar indicating the desired autopilot design.

(See HDCON)

c. Output Arguments

None

d. Common Blocks

DISCSV, WORKS9, MODF, PLA, DIMN1, DIMN3, DIMN4, DIMN5, SMAT, PIFG,

EIGEN, LABEL, UDIR, GLAD, DISCM, HDXA

e. Error Messages

If DEBUG is TRUE then the following matrices are printed out, DFM,

DGM, UMAG, USTRT, USTP.

f. Subroutines Employed by DPIFS

ORACLS - NULL, MULT, ADD, SUBT, EQUATE

PIFLIB - PRNT2, PRNTITL, CLXMOD, CLYMOD, CLDMOD

g. Subroutines Employing DPIFS

PIFLIB - PIFCGT

54
h. Concluding Remarks

DPIFS does not include plotting.
0. DISCRETE RICCATI EQUATION SOLUTION (DREG)

1. PURPOSE

Subroutine DREG solves the discrete Riccati equation:

\[
X = \Phi^T X \Phi + Q - (\Gamma^T X \Phi + \Lambda^T)^T * \\
(R + \Gamma^T X \Gamma)^{-1} (\Gamma^T X \Phi + \Lambda^T)
\]  

(1)

for \(X\), then computes the feedback gain

\[
G = - (R + \Gamma^T X \Gamma)^{-1} * \\
(\Gamma^T X \Phi + \Lambda^T)
\]

(2)

The solution is obtained using NEWTON'S method as discussed in ORACL'S documentation of the ASYMREG subroutine. The cross weighting matrix AMHAT is eliminated from the quadratic scalar function before solving the Riccati equation using the procedure outlined in the ORACL'S subroutine PREFIL. If ICH(2,7) is TRUE, the input gain GK must cause the closed-loop plant system matrix

\[
\Phi_{CL} = \Phi + \Gamma G
\]

(3)

to be asymptotically zero. If ICH(2,7) is FALSE the subroutine attempts to find an initial starting stabilizing gain, G, using the ORACL'S subroutine DSTAB.

2. USAGE

a. Calling Sequence

CALL DREG (RHAT, NRH, AMHAT, NAM, QHAT, NQH, GAMA, NGA, X, NX, PHI,
NPH, GK, NGK, PHICL, NPHC, ICH, DEBUG, ZERO, NZERO, W9, NW9DIM, IOP

b. Input Arguments

RHAT, AMHAT,

QHAT, GAMA,

PHI Matrices packed by columns into one-dimensional arrays. Not destroyed upon return.

GK Matrix packed by column into a one-dimensional array. Destroyed upon return. GK is assumed to contain a stabilizing gain if ICH(2,7) is TRUE.

W9 Work matrix packed by columns into a one-dimensional array. Dimension must be at least 9*(NPH(1)*NPH(2) + 1)

ZERO Work matrix packed by columns into a one-dimensional array. All elements must be zero. Dimension must be at least NPH (1) * NPH(2) + 1

ICH Two-dimensional logical matrix dimensioned 10 x 15. Subroutines in PIFCGT are called depending on the TRUE or FALSE status of elements in ICH.

NGK, NX,

NQH, NAM, NPH,

NGA, NZERO,

NPHC Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N. NZERO is the only vector destroyed upon return.

DEBUG Logical scalar indicating the following:
TRUE: Print out debug information
FALSE: Do not print out debug information

DELT Real scalar containing the sampling time.
NW9DIM   Integer scalar containing the maximum dimension of the W9 matrix.

IOP(1)=0  Do not print results. Otherwise print input data, QHAT and final values of GK and X.

IOP(2)=0  Do not print at each iteration. Otherwise regardless of printing specified by IOP(1), print iteration count and value of X.

IOP(3)    Not Required.

IOP(4)=0  Do not compute the Riccati equation residual. Otherwise compute the residual and print.

IOP(5)=0  Compute but do not print the eigenvalues of X, the matrix (PHI - GAMA GK) and the eigenvalues of (PHI - GAMA GK).
Otherwise print these data after computation.

c. Output Arguments

X, GK,

PHICL    Matrices packed by columns into one-dimensional arrays containing respectively the discrete Riccati equation solution, the discrete feedback gain, and the discrete closed-loop system matrix.

NX, NGK,

NPHC     Two dimensional integer vectors holding the number of rows and columns of the matrix shown after the letter N.

d. COMMON Blocks

None

e. Error Messages

If the workspace in the W9 matrix is too small the message "WORK SPACE IS TOO SMALL FOR CALCULATION". WORK SPACE = _______. WC = _______.

58
The solution to the RICCATI equation \( (X) \) and the gain matrix \( (G_K) \) are always printed out.

f. Subroutines Employed by DREG

ORACLS - MULT, SUBT, TESTSTA, SCALE, RICTNWT, TRANP, ADD, EQUATE, EIGEN,
JUXTC, DSTAB, UNITY, NULL, ASYMREG,
PIFLIB - PRNTITL, PRNT

Subroutines Employing DREG

PFICGT

h. Concluding Remarks

DSTAB fails consistently in computing a stabilizing gain. An alternate method for computing the starting gain is performed by PIFCGT.
P. SUBSET OF A MATRIX (EXTR)

1. PURPOSE

Subroutine EXTR extracts a matrix from within another matrix.

2. USAGE

a. Calling Sequence

CALL EXTR(A,NA,B,NB,I,J)

b. Input Arguments

A     Matrix packed by columns in one dimensional array. Not destroyed upon return.

NA    2 element vector giving number of rows and columns respectively of A. Not destroyed upon return.

NB    2 element vector giving number of rows and columns to be extracted into B. Not destroyed upon return.

I & J  Define upper left hand corner position of B as a row and column position in A.

c. Output Arguments

B     Matrix that is extracted. Packed column wise into one dimensional array.

d. COMMON Blocks

None

e. Error Messages

If either NA(1) < NB(1), or NA(2) < NB(2), or NA(1) x NA(2) < 1, or NB(1) x NB(2) < 1, the message is "DIMENSION ERROR IN CORNER NA = _____ _____, NB = _____ _____" is printed and the program is returned to the calling point.
f. Subroutines employed by EXTR

None

g. Subroutines employing EXTR

PIFLIB - CFWM, PIFPLC, PIFG, HADAC.

h. Concluding Remarks

This routine is useful for extracting relevant sub-matrices from within a partitioned matrix as shown below.

\[
\begin{bmatrix}
A = I & J \\
NB(1) + I & \text{\dag} \\
\end{bmatrix}
\begin{bmatrix}
\text{\dag} & J \\
NB(2) + J \\
\end{bmatrix}
\]
Q. F AND G CONSTRUCTION FROM AERODYNAMIC COEFFICIENTS (FGAERO)

1. PURPOSE

Subroutine FGAERO accepts nondimensional aerodynamic coefficients, trim conditions, and a desired velocity magnitude and constructs the aircraft perturbation model matrices F and G at the trim condition specified by the desired aircraft velocity, where

\[ \Delta \dot{x} = F \Delta x + G \Delta u \]  

A derivation of the perturbation aircraft dynamics based on the non-linear dynamics is summarized in Ref. 1. Information on the notation used in the following discussion can be found in Ref. 1, Appendix A.

The F and G matrices are partitioned as follows:

\[
\begin{bmatrix}
\Delta \dot{\omega}_y \\
\Delta \dot{\psi}_z \\
\Delta \dot{\alpha}_B \\
\Delta \dot{\beta}_B
\end{bmatrix} =
\begin{bmatrix}
A11 & A12 & A13 & 0 \\
A21 & A22 & 0 & 0 \\
0 & A32 & A33 & 0 \\
A41 & 0 & A43 & 0
\end{bmatrix}
\begin{bmatrix}
\Delta \dot{\psi}_B \\
\Delta \dot{\psi}_B \\
\Delta \dot{\psi}_B \\
\Delta \dot{\psi}_B
\end{bmatrix}
\]

\[ + \begin{bmatrix}
\Delta u \\
\Delta u \\
0 \\
0
\end{bmatrix} + \begin{bmatrix}
F1 \\
F2 \\
0 \\
0
\end{bmatrix} \Delta \dot{\psi}_w \]

The state and control vectors are

\[
[\Delta \dot{\psi}_B^T \Delta \dot{\psi}_B^T \Delta \dot{\psi}_B^T \Delta \dot{x}_E^T] = [\Delta u \Delta v \Delta w \Delta p \Delta q \Delta r \Delta \phi \Delta \psi \Delta x \Delta y \Delta z] \]
\[ \Delta u^T = [\Delta \delta_T \ \Delta \delta_e \ \Delta \delta_a \ \Delta \delta_r] \quad (4) \]

\[ \Delta \dot{\omega} = [\Delta \dot{\phi} \ \Delta \dot{\psi}] \quad (5) \]

The effect of \( \Delta \dot{\omega} \) is included after the above partitions have been computed. The partitioned matrices in Eq. 2 satisfy the following:

\[ A_{11} = F_{BV} - W_{BI} + F_{BAB}^*A_{U} \quad (6) \]

\[ A_{12} = V_{BTC} + F_{BW} \quad (7) \]

\[ A_{13} = N_{BV} \quad (8) \]

\[ A_{21} = I_{B}^{-1}*(G_{BV} + G_{BAB}^*A_{U}) \quad (9) \]

\[ A_{22} = I_{B}^{-1}G_{BW} - I_{B}^2I_{B} \quad (10) \]

\[ A_{32} = L_{B}^{-1} \quad (11) \]

\[ A_{33} = L_{BLB} \quad (12) \]

\[ A_{41} = (H_{BI})^T = H_{IB} \quad (13) \]

\[ A_{43} = -(H_{IB}^*V_{B})*H_{IB}^*L_{B} \quad (14) \]

\[ F_{1} = 0 \quad (15) \]

\[ F_{2} = I_{B}^{-1}G_{BAB}D \quad (16) \]

\[ B_{11} = F_{BU} \quad (17) \]

\[ B_{21} = I_{B}^{-1}G_{BU} \quad (18) \]

\[ L_{B} = \begin{bmatrix} 1 & 0 & -\sin(\text{THETA}_0) \\ 0 & \cos(\text{PHIO}) & \sin(\text{PHIO}) \cos(\text{THETA}_0) \\ 0 & -\sin(\text{PHIO}) & \cos(\text{PHIO}) \cos(\text{THETA}_0) \end{bmatrix} \quad (19) \]

\[ \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = L_{B} \begin{bmatrix} P_{O} \\ Q_{O} \\ R_{O} \end{bmatrix} = L_{B} * W_{B} \quad (20) \]
\[ \mathbf{L} \mathbf{L} \mathbf{B} \mathbf{B} = \begin{bmatrix} \dot{\theta} \tan(\theta) & \dot{\psi} \sec(\theta) & 0 \\ -\dot{\psi} \cos(\theta) & 0 & 0 \\ \dot{\theta} \sec(\theta) & \dot{\psi} \tan(\theta) & 0 \end{bmatrix} \]  

(21)

\[ \mathbf{V} \mathbf{B} = \begin{bmatrix} U_0 \\ V_0 \\ W_0 \end{bmatrix} \]  

(22)

\[ \mathbf{V} \mathbf{B} \mathbf{T} = \begin{bmatrix} 0 & -W_0 & V_0 \\ W_0 & 0 & -U_0 \\ -V_0 & U_0 & 0 \end{bmatrix} \quad \sim \quad \mathbf{V} \mathbf{B} \]  

(23)

\[ \mathbf{W} \mathbf{B} = \begin{bmatrix} P_0 \\ Q_0 \\ R_0 \end{bmatrix} \]  

(24)

\[ \mathbf{W} \mathbf{B} \mathbf{I} = \begin{bmatrix} 0 & -R_0 & Q_0 \\ R_0 & 0 & -P_0 \\ -Q_0 & P_0 & 0 \end{bmatrix} \quad \sim \quad \mathbf{W} \mathbf{B} \]  

(25)

\[ \mathbf{H} \mathbf{B} \mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\phi) & \sin(\phi) \\ 0 & -\sin(\phi) & \cos(\phi) \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \]  

(26)

\[ \mathbf{G} \mathbf{G} = \begin{bmatrix} G_5 \\ G_6 \\ G_7 \end{bmatrix} = \mathbf{H} \mathbf{B} \mathbf{I} \begin{bmatrix} 0 \\ 0 \\ G \mathbf{R} \end{bmatrix} \]  

(27)
\( \mathbf{GB} = \begin{bmatrix} 0 & -G7 & G6 \\ G7 & 0 & -G5 \\ -G6 & G5 & 0 \end{bmatrix} = \mathbf{GG} \) (28)

(The subroutine \textsc{crospp} forms the cross produce equivalent matrix, \( \mathbf{GB} \), from the vector \( \mathbf{GG} \))

\[ \mathbf{IB} = \begin{bmatrix} \mathbf{IX} & 0 & 0 \\ 0 & \mathbf{IY} & 0 \\ 0 & 0 & \mathbf{IZ} \end{bmatrix} \] (29)

\[ \mathbf{IBIB} = \mathbf{IB}^{-1} \cdot \left[ \mathbf{WBI} \cdot \mathbf{IB} - (\mathbf{IB} \cdot \mathbf{WB}) \right] \] (30)

\( S = \text{RHO} \cdot \mathbf{A} \cdot 0.5 \) (31)

\( \text{AMASSI} = \frac{\text{GR}}{\text{WT}} \) (32)

\[ \mathbf{FBV} = 2 \cdot S \cdot \begin{bmatrix} \mathbf{CYO} \cdot \mathbf{UO} & \mathbf{CYO} \cdot \mathbf{VO} & \mathbf{CYO} \cdot \mathbf{WO} \\ \mathbf{CZO} \cdot \mathbf{UO} & \mathbf{CZO} \cdot \mathbf{VO} & \mathbf{CZO} \cdot \mathbf{WO} \end{bmatrix} \] (33)

\[ \mathbf{FBW} = 0.5 \cdot S \cdot \mathbf{VTO} \cdot \begin{bmatrix} \mathbf{B} \cdot \mathbf{CXP} & \mathbf{C} \cdot \mathbf{CXQ} & \mathbf{B} \cdot \mathbf{CXR} \\ \mathbf{B} \cdot \mathbf{CYP} & \mathbf{C} \cdot \mathbf{CYQ} & \mathbf{B} \cdot \mathbf{CYR} \\ \mathbf{B} \cdot \mathbf{CZP} & \mathbf{C} \cdot \mathbf{CZQ} & \mathbf{B} \cdot \mathbf{CZR} \end{bmatrix} \] (34)

\[ \mathbf{FBAB} = S \cdot \mathbf{VTO}^{\ast 2} \cdot \begin{bmatrix} \mathbf{CXB} & \mathbf{CXA} \\ \mathbf{CYB} & \mathbf{CYA} \\ \mathbf{CZB} & \mathbf{CZA} \end{bmatrix} \] (35)

\[ \mathbf{FBU} = S \cdot \mathbf{VTO}^{\ast 2} \cdot \begin{bmatrix} \mathbf{CXDT} & \mathbf{CXDE} & \mathbf{CXDA} & \mathbf{CXDR} \\ \mathbf{CYDT} & \mathbf{CYDE} & \mathbf{CYDA} & \mathbf{CYDR} \\ \mathbf{CZDT} & \mathbf{CZDE} & \mathbf{CZDA} & \mathbf{CZDR} \end{bmatrix} \] (36)

65
\[ GBV = 2S \begin{bmatrix} B\text{CLO}UO & B\text{CLO}VO & B\text{CLO}WO \\ C\text{CMO}UO & C\text{CMO}VO & C\text{CMO}WO \\ B\text{CNO}UO & B\text{CNO}VO & B\text{CNO}WO \end{bmatrix} \]

\[ GBAB = S^2VTO^2 \begin{bmatrix} C\text{CMB} & C\text{CMA} \\ B\text{CNB} & B\text{CNA} \end{bmatrix} \]

\[ GBW = 0.5S^2VTO \begin{bmatrix} B\text{**2CLP} & B\text{C\text{CLQ}} & B\text{B*CLR} \\ B\text{C\text{CMP}} & C\text{CMQ} & B\text{C\text{CMR}} \\ B\text{**2CNP} & B\text{C\text{CNQ}} & B\text{B\text{CNR}} \end{bmatrix} \]

\[ GBU = S^2VTO^2 \begin{bmatrix} B\text{*CLDT} & B\text{CLDE} & B\text{CLDA} & B\text{CLDR} \\ C\text{CMDT} & C\text{CMDE} & C\text{CMDA} & C\text{CMDR} \\ B\text{CNDT} & B\text{CNDE} & B\text{CNDA} & B\text{CNDR} \end{bmatrix} \]

\[ GBABD = 0.5S^2VTO \begin{bmatrix} B\text{B*CLBDOT} & C\text{CCLADOT} \\ B\text{C\text{CMBDOT}} & C\text{C\text{CMADOT}} \\ B\text{B\text{CNBDOT}} & B\text{C\text{CNADOT}} \end{bmatrix} \]

\[ AU = \begin{bmatrix} 0 & 1.0/VTO & 0 \\ 0 & 0 & UO \\ UO\times2+WO\times2 & 0 & UO\times2+WO\times2 \end{bmatrix} \]

\[ ABV = \begin{bmatrix} 0 & 1/VTO & 0 \\ 0 & 0 & 1/VTO \end{bmatrix} \]

\[ Z = I - ABV \times F1 \]

\[ G1 = Z^{-1} \begin{bmatrix} ABV\text{A11} & ABV\text{A12} & ABV\text{A13} & 0 \end{bmatrix} \]

\[ G2 = Z^{-1} \times ABV \times B1 \]
The effect $\Delta \mathbf{w}$ is included to produce the final form for $F$ and $G$

\[
\Delta \mathbf{w} = G_1 \mathbf{x} + G_2 \mathbf{u}
\] (46)

\[
F = F + \begin{bmatrix} 0 \\ F_2 \\ 0 \\ 0 \end{bmatrix} \cdot G_1
\] (47)

\[
G = G + \begin{bmatrix} 0 \\ F_2 \\ 0 \\ 0 \end{bmatrix} \cdot G_2
\] (48)

After $F$ and $G$ have been constructed the states are reordered as follows:

\[
\Delta \mathbf{x}^T = [\Delta u \ \Delta w \ \Delta q \ \Delta \theta \ \Delta z \ \Delta x \ \Delta v \ \Delta r \ \Delta \phi \ \Delta \psi \ \Delta y]
\] (49)

completing the linear time-invariant aircraft model construction. If $\text{ICH}(1,5)$ is TRUE, the units are changed from radians to degrees.

2. Usage

a. Calling Sequence

CALL FGAERO (DEBUG, ICH, HBI, NHBI, LB, NLB)

b. Input Arguments

DEBUG Logical scalar indicating the following:

\[
\begin{align*}
\text{TRUE:} & \quad \text{Print out debug information} \\
\text{FALSE:} & \quad \text{Do not print out debug information}
\end{align*}
\]

ICH Two-dimensional logical matrix dimensioned 10 x 15. Subroutines in PIFCGT are called depending on the TRUE or FALSE status of elements in ICH.

c. Output Arguments

HBI, LB Matrices packed by columns into one dimensional arrays.

NHBI, NLB Two dimensional integer vector holding the number of rows and columns of the matrix after the letter N.
d. COMMON Blocks

PLA, GLAD, DIMN4, AERO, WORKS9, HDXA

e. Error Messages

If the LB matrix is not invertible the message "LB IS A SINGULAR MATRIX" is printed and the program stops. If the Z matrix is not invertible the message "Z IS A SINGULAR MATRIX" is printed and the program stops. If DEBUG is TRUE the following statements and matrices are printed during the course of program execution

*** s =

*** AMASSI =

LB, HBI, GB, NBV, LB\(^{-1}\)(LB1), LBLB, IB, IB\(^{-1}\)(IB1), IBIB, FBV, FBW, FBAB, FBU, GBV, GBAB, GBW, GBU, GBBD, A11, A12, A13.

A1 The F matrix in Eq. 2, before including the effect of \(\Delta v_w\) and the position states \(\Delta x_E\)

B11, B21,

B31, B1 The G matrix in Eq. 2, before including the effect of \(\Delta v_w\) and the position states \(\Delta x_E\)

F11, F21, F31, F1, Z, Z\(^{-1}\)(ZINV), G1, G2,

FORD The A1 matrix including the effect of \(\Delta v_w\)

GHIA The B1 matrix including the effect of \(\Delta v_w\)

If ICH(1,5) is TRUE then the following are printed

FDEG The F matrix in Eq. 1 using degree units

GDEG The G matrix in Eq. 1 using degree units

f. Subroutines Employed by FGAERO

ORACL5 - EQUATE, MULT, UNITY, GELIM, DETFAC, GAUSEL, SUBT, SCALE, ADD,

JUXTA, JUXTR

PIFLIB - PRNTITL, PRNT, MOVEALL, CORNER, DIAGPAR, CROSSP, R3MAT

68
g. Subroutines Employing FGAERO

PIFLIB - PIFCGT

h. Concluding Remarks

None
1. PURPOSE

The purpose of FREEF is to compute eigenvalues, $\lambda$, and eigenvectors, $x$, of the matrix, $F$, where

$$Fx = \lambda x, \quad \lambda = R + jI$$

FREEF assumes $F$ is a closed-loop plant matrix

$$F = A + BK$$

so that eigenvalues of $F$ are interpreted as indications of closed-loop stability. FREEF uses the NASA Langley FTNMLIB subroutine CSQZ. CSQZ uses Householder transformations and a combination of the double shift QZ iteration and the single shift implicit QZ iteration. Consult Section 2h for further remarks.

2. USAGE

a. Calling Sequence

CALL FREEF (F, ER, EI, V, WK, IFREE, N)

b. Input Arguments

$F$ Matrix packed by columns into one-dimensional array. Destroyed upon return.

$\text{EFREE}$ Integer scalar for eigenvalue computations

0 - No calculation in FREEF
1 - Compute and print eigenvalues
2 - Compute and print eigenvalues and eigenvectors

$N$ Integer scalar indicating row dimension of the square matrix $F$. 

70
WK Work matrix of dimension at least 2*(N**2+1)

c. Output Arguments

ER Real vector containing the real part of the eigenvalues in ascending order of magnitude

EI Real vector containing the imaginary part of the eigenvalues in ascending order of magnitude

WK Real matrix containing the eigenvectors packed by columns into a one-dimensioned array. A real eigenvector is placed in one column. A complex eigenvector is placed in two columns. The real part is placed in the first column. The imaginary part of the positive complex eigenvector pair is placed in the second column. The arrangement is in ascending order of magnitude.

d. COMMON Blocks

None

e. Error Messages

After CSQZ is called, the following is printed:

OUTPUT OF CSQZ IN FREE

EPSA = _____ EPSB = _____ IERR = _____

Eigenvalues are calculated as (\alpha_1+j\alpha_2)/\beta. EPSA is the inaccuracy level of |\alpha_1+j\alpha_2|. (Digits below this level are known to be inaccurate). EPSB is the inaccuracy level of \beta. IERR is an error code.

IERR = 0 normal return.

= I Ith eigenvalue ratio has not been determined after 30 iterations.

If \beta is zero for eigenvalue I the following is printed:

EIGENVALUE PROBLEM IN CSQZ IN FREE: BETA IS ZERO I = __.
Additional information for each eigenvalue is printed with the following banners:

**EIGENVALUES**
The values for the eigenvalues R and jI

**TIME CONSTANT**
\(-1.0/R\) - Time to reach 63.212% of a step command

**50 PERCENT SET TIME**
\(0.6931472*(-1.0/R)\) - Time to reach 50% of a step command

**10 PERCENT SET TIME**
\(2.302582*(-1.0/R)\) - Time to reach 90% of a step command

**2 PERCENT SET TIME**
\(3.9120230*(-1.0/R)\) - Time to reach 98% of a step command

**UNDAMPED NAT FREQ**
\((R**2+I**2)\) - Undamped natural frequency of a complex eigenvalue

**RATIO**
\(-R/(R**2+I**2)\) - Damping ratio of a complex eigenvalue

**RISE TIME**
\(0.5*(6.28318531)/|I| - TAN^{-1} (|I|/R)/|I|\) - time for the step response of a complex eigenvalue to cross the step command the first time.

**PERIOD**
\(6.28318531/|I|\) - Time for a complex eigenvalue to complete one oscillation.

**OVERSHOOT**
\(100.0* e^{(0.5*R*6.28318531/|I|)}\) - The overshoot of a stable complex eigenvalue in response to a step input.

*f. Subroutines Employed by FREEF*

**ORACLS** - UNITY

**PIFLIB** - RESON

**FTNMLIB** - CSQZ

*g. Subroutines Employing FREEF*

None
h. Concluding Remarks

A user with access to the subroutine CSQZ should replace the call to FREEO with a call to FREEF in the following subroutines - PIFCGT, PIFEIG, PIFPLC, PIFFRE. A user without access to CSQZ should remove the subroutine FREEF from the software package.
S. EIGENVALUES/EIGENVECTORS OF CLOSED-LOOP PLANTS USING EIGEN (FREEO)

1. PURPOSE

The purpose of FREEO is to compute eigenvalues, \( \lambda \), and eigenvectors, \( x \), of the matrix, \( F \), where

\[
Fx = \lambda x, \quad \lambda = R + jI
\]  

\( \text{FREEO} \) assumes \( F \) is a closed-loop plant matrix

\[
F = A + BK
\]

so that eigenvalues of \( F \) are interpreted as indications of closed-loop stability. \( \text{FREEO} \) uses the \text{ORACLS} subroutine EIGEN. EIGEN balances the matrix and the eigenvalues are found by stabilized elementary similarity transformations and the double shift QR algorithm.

2. USAGE

a. Calling Sequence

CALL FREEO (F, ER, EI, V, WK, IFREE, N)

b. Input Arguments

\( F \)  
Matrix packed by columns into one-dimensional array; destroyed upon return.

\( \text{IFREE} \)  
Integer scalar for eigenvalue computations

0 - No calculation in FREEO

1 - Compute and print eigenvalues

2 - Compute and print eigenvalues and eigenvectors

\( N \)  
Integer scalar indicating row dimension of the square matrix \( F \)

\( \text{WK} \)  
Work matrix of dimension at least \( 2*(N**2+1) \)
c. Output Arguments

ER  Real vector containing the real part of the eigenvalues in ascending order of magnitude

EI  Real vector containing the imaginary part of the eigenvalues in ascending order of magnitude

WK  Real matrix containing the eigenvectors packed by columns into a one-dimensional array. A real eigenvector is placed in one column. A complex eigenvector is placed in two columns. The real part is placed in the first column. The imaginary part of the positive complex eigenvector pair is placed in the second column. The arrangement is in ascending order of magnitude.

d. COMMON Blocks

None

e. Error Messages

After EIGEN is called and if IERR (defined in EIGEN documentation) is nonzero the value of IERR is printed out. Additional information for each eigenvalue is printed with the following banners:

EIGENVALUES  The values for the eigenvalues R + jI

TIME CONSTANT  -1.0/R - Time to reach 63.212% of a step command

50 PERCENT SET TIME  0.6931472*(-1.0/R) - Time to reach 50% of a step command

10 PERCENT SET TIME  2.302582*(-1.0/R) - Time to reach 90% of a step command

2 PERCENT SET TIME  3.9120230*(01.0/R) - Time to reach 98% of a step command

UNDAMPED NAT FREQ  (R**2+I**2) - Undamped natural frequency of a complex eigenvalue
RATIO \(-\frac{R}{R^2+I^2}\) - Damping ratio of a complex eigenvalue

RISE TIME \(0.5\times(6.28318531/|I|) - \tan^{-1} \left(\frac{|I|/R}{|I|}\right)\) - Time for the step response of a complex eigenvalue to cross the step command the first time.

PERIOD \(6.28318531/|I|\) - Time for a complex eigenvalue to complete one oscillation.

OVERSHOOT \(100.0\times e^{(0.5\times R\times 6.28318531/|I|)}\) - The overshoot of a stable complex eigenvalue in response to a step input.

f. Subroutines Employed by FREEF

ORACLS - UNITY EIGEN

PIFLIB - RESPON

g. Subroutines Employing FREEF

PIFCGT, PIFEIG, PIFPLC, PIFFRE

h. Concluding Remarks

None
1. PURPOSE

HADAC constructs the 3x13 and 3x4 observation matrices HAC and DAC and the 3x12 observation matrix HAB. HAC and DAC initially are the body-mounted perturbation accelerometer observation matrices and satisfy

\[
\Delta a_B = HAC \Delta x + DAC \Delta u
\]

where \( \Delta a_B \) is the perturbation body-axis accelerometer output,

\[
\Delta a_B^T = [\Delta a_x \Delta a_y \Delta a_z]
\]

\( \Delta x \) is the perturbation aircraft state vector

\[
\Delta x^T = [\Delta u \Delta w \Delta v \Delta p \Delta q \Delta r \Delta \phi \Delta \theta \Delta \psi \Delta x \Delta y \Delta z]
\]

\( \Delta u \) is the perturbation aircraft control vector

\[
\Delta u^T = [\Delta \delta_T \Delta \delta_e \Delta \delta_a \Delta \delta_r]
\]

The units must be feet and radians.

HAB is the perturbation observation matrix for total velocity, \( \Delta v \), sideslip, \( \Delta \beta \) and angle of attack, \( \Delta \alpha \), where

\[
\Delta \eta = HAB \Delta x
\]

and

\[
\Delta \eta = \begin{bmatrix} \Delta v \\ \Delta \beta \\ \Delta \alpha \end{bmatrix}
\]

The construction of HAC and DAC proceeds by first partitioning the 12x12 plant system matrix, \( F \), and 12x4 control matrix, \( G \), as follows...
The expressions for HAC, DAC, and HAB become (see Ref.1 for a derivation)

\[
H_{AC} = \begin{bmatrix} H_1 & H_2 & H_3 & 0 \end{bmatrix}
\]

\[H_1 = FXX + TW - TSB \times FWX\]  
\[H_2 = T(TXB \times W) - TW \times TXB - TVB + FWX - TXB \times FWW\]  
\[H_3 = FXV - TXB \times FWV - GB \times LB\]

where

\[
LB = \begin{bmatrix} 1.0 & 0 & -\sin(\Theta) \\ 0 & \cos(\Phi) & \sin(\Phi) \times \cos(\Theta) \\ 0 & -\sin(\Phi) & \cos(\Phi) \times \cos(\Theta) \end{bmatrix}
\]

\[H_{BI} = H_{IB}^T\]

\[H_{IB} = T_1 \times T_2 \times T_3\]

\[
T_1 = \begin{bmatrix} \cos(\Psi) & -\sin(\Psi) & 0 \\ \sin(\Psi) & \cos(\Psi) & 0 \\ 0 & 0 & 1.0 \end{bmatrix}
\]

\[
T_2 = \begin{bmatrix} \cos(\Phi) & 0 & \sin(\Phi) \\ 0 & 1.0 & 0 \\ -\sin(\Phi) & 0 & \cos(\Phi) \end{bmatrix}
\]

78
\[ T_3 = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & \cos(\Theta) & -\sin(\Theta) \\ 0 & \sin(\Theta) & \cos(\Theta) \end{bmatrix} \] (18)

\[ TW = \begin{bmatrix} 0 & -RO & QO \\ RO & 0 & -PO \\ -QO & PO & 0 \end{bmatrix} \] (19)

\[ TXB = \begin{bmatrix} 0 & -ZS & XS \\ ZS & 0 & -YS \\ -XS & -YS & 0 \end{bmatrix} \] (20)

\[ VBT = \begin{bmatrix} 0 & -W0 & U0 \\ W0 & 0 & -V0 \\ -U0 & V0 & 0 \end{bmatrix} \] (21)

\[ \begin{bmatrix} g_1 \\ g_2 \\ g_3 \end{bmatrix} = HBI \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \] (22)

\[ GB = \begin{bmatrix} 0 & -g_3 & g_1 \\ g_3 & 0 & -g_2 \\ -g_1 & g_2 & 0 \end{bmatrix} \] (23)

\[ \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = TXB \begin{bmatrix} PO \\ QO \\ RO \end{bmatrix} \] (24)

\[ T(TXB*W) = \begin{bmatrix} 0 & -x_3 & x_1 \\ x_3 & 0 & -x_2 \\ -x_1 & x_2 & 0 \end{bmatrix} \] (25)
DAC = [GXU - TXB * GWU] \hfill (26)

where

\[ HAB = [JW * HWB \ 0 \ 0 \ 0] \] \hfill (27)

\[ JW = \begin{bmatrix} 1.0 & 0 & 0 \\ 0 & \frac{1.0*DPR}{V_{T_o}} & 0 \\ 0 & 0 & \frac{1.0*DPR}{(V_{T_o} \cos \beta_0)} \end{bmatrix} \] \hfill (28)

\[ HWB = HBW^T \] \hfill (29)

\[ HBW = \begin{bmatrix} \cos \beta_0 & 0 & -\sin \beta_0 \\ 0 & 1.0 & 0 \\ \sin \beta_0 & 0 & \cos \beta_0 \end{bmatrix} \begin{bmatrix} \cos \beta_0 & \sin \beta_0 & 0 \\ -\sin \beta_0 & \cos \beta_0 & 0 \\ 0 & 0 & 1.0 \end{bmatrix} \] \hfill (30)

\[ DPR = \frac{180.0}{\pi} \] \hfill (31)

\[ V_{T_o} = \sqrt{U_0^2 + V_0^2 + W_0^2} \] \hfill (32)

\[ \beta_0 = \sin^{-1} \left( \frac{V_0}{V_{T_o}} \right) \] \hfill (33)

\[ \alpha_0 = \tan^{-1} \left( \frac{W_0}{U_0} \right) \] \hfill (34)
If ICH(2,12) is TRUE, $\Delta a_x$ observation is replaced with velocity magnitude, $\Delta V$, observation in HAC DAC by replacing the first row in HAC with the first row in HAB and zeroing out the first row in DAC. In a similar manner if ICH(2,13) is TRUE, $\Delta a_y$ observation is replaced with sideslip observation, $\Delta \beta$, and if ICH(2,14) is TRUE, $\Delta a_z$ observation is replaced with angle-of-attack observation, $\Delta \alpha$.

2. USAGE

a. Calling Sequence

CALL HADAC (LB, NLB, VBT, NVBT, GB, NGB, DEBUG, F, NF, G, NG, XS,
YS, ZS, PO, QO, RO UO, VO, WO, HAC, DAC, ICH, NGAC, NDAC, HAB, NHAB)

b. Input Arguments

LB, GB, F, G, VBT Matrices packed by columns into one-dimensional arrays; not destroyed upon return. These matrices are constructed in FGAERO

NLB, NVBT,
NGB, NF, NG Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N.

XS, YS,

ZS Real scalars containing the x, y, and z body axis position of the accelerometers with reference to the aircraft center of gravity.

PO, QO, RO,
UO, VO, WO Real scalar trim conditions for roll rate, pitch rate, yaw rate, body-axis forward velocity, lateral velocity and vertical velocity, respectively.

ICH Two-dimensional logical matrix dimensioned 10x15.
Subroutines in PIFCGT are called depending on the TRUE or FALSE status of elements in ICH.

c. Output Arguments

HAC, DAC,

HAB Matrices packed by columns into one-dimensional arrays

NHAC, NDAC,

NHAB Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N.

d. COMMON Blocks

WORKS9

e. Error Messages

If DEBUG if TRUE the following matrices are printed our during the course of program execution: FXX, GXU, JW*HWB(JHWB), HAB, HAC, DAC.

f. Subroutines Employed by HADAC

ORACLS - NULL, MULT, SUBT, ADD, JUXTC, TRANP, UNITY

PIFLIB - CROSSP, EXTR, PRNT, PRNTITL

g. Subroutines Employing Hadac

PIFLIB - FGAERO

h. Concluding Remarks

None

1. PURPOSE

The purpose of HDCON is to construct the FM, GM, HM, DM, H and D matrices, to set up the XR, UR, YR, XMR, UMR logical vectors used by the REMAT subroutine, and to modify the XR, UR, YR, and UMR vectors as needed to reflect changes from initial specifications in AEROD (BLOCK DATA). The program has the necessary information for each command system that is chosen by the designer using ICROSE. For each value of ICROSE supported by the program, (ICROSE = 4 is available but not supported), the values for elements FM, GM, HM, H and D are shown in Ref. 1, Appendix C. The program constructs the matrices assuming the aircraft states and controls have the following order,

\[ \Delta x^T = [\Delta u \ \Delta w \ \Delta q \ \Delta \theta \ \Delta z \ \Delta v \ \Delta r \ \Delta \phi \ \Delta \psi \ \Delta y] \]  

\[ \Delta u^T = [\Delta \delta_e \ \Delta \delta_a \ \Delta \delta_r] \]  

The order for \( \Delta y \), \( \Delta y_m \), \( \Delta x_m \), and \( u_m \) are the same as the order in Ref. 1.

After constructing the matrices, the states and controls which are to be eliminated are identified by setting appropriate elements in XR, UR and YR to FALSE. The order for XR and UR are the same as Eqs. 1 and 2. The first two elements in YR are assumed to be longitudinal. The latter two elements in YR are assumed to be lateral.

The weighting elements for the integrator states, \( Z_{1Q}, Z_{2Q}, Z_{3Q}, Z_{4Q} \), have the same order as YR. Integrator states are eliminated in REMAT using the YR logical vector. The values for NM, the number of command model controls are specified in HDCON for each ICROSE value.
The values for L, N, and M are always 4, 12 and .4, respectively. REMAT changes these values depending on XR, UR, and YR. The plotting labels in YL must be changed to reflect different command systems. The first two rows in YL are assumed to be longitudinal. The latter two rows in YL are assumed to be lateral. Rows in YL are eliminated in REMAT using the YR logical vector.

2. USAGE
a. Calling Sequence.

CALL HDCON (H, NH, D, ND, FM, NFM, GM, NGM, HM, NHM, DM, NDM, ICHOSE, W9, NW9DIM, UO, WO, QO, RO, VTO, THETAO, PHI0, DEBUG, METERS, XR, UR, YR, XMR, UMR, ICH, HAC, NHAC, DAC, NDAC, GR, N, M, L, NM, MM, XL, YL, GKPSI, CKY, F, LAMDA, ZETA)

b. Input Arguments

F The 12'x 12 aircraft system matrix packed by columns in a one-dimensional array. The F matrix is constructed in FGAERO. The F matrix is altered if ICHOSE = 14.

ICH Two-dimensional logical matrix dimensioned 10 x 15. Subroutines in PIFCCT are called depending on the TRUE or FALSE status of elements in ICH.

W9 Work matrix packed by columns into a one-dimensional array. Dimension must be at least 7*(n + m + 1).

DEBUG Logical scalar indicating the following:
TRUE: Print out debug information
FALSE: Do not print out debug information.

NW9DIM Integer scalar containing the maximum dimension of the W9 matrix.
Trim values for the states. The trim values are obtained from linear interpolation in INTERP. \( \lambda \) is the flight path angle in degrees (\( \lambda = 0.0 \) deg usually). \( \zeta \) is the ground relative heading angle in deg (\( \zeta = 0.0 \) usually).

**ICHOSE**

Integer scalar indicating the desired autopilot designs as follows:

<table>
<thead>
<tr>
<th>ICHOOSE</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-BETA HOLD</td>
<td>Beta and roll angle constant command system for lateral dynamics. Beta and roll angle perturbation commands for DPIFS simulation are in ( \text{UMAG}(1,1) ) and ( \text{UMAG}(2,1) ), respectively.</td>
</tr>
<tr>
<td>6-HDG SEL</td>
<td>Yaw angle outer loop turning command system using the ROLL HOLD command system as the inner loop structure. Yaw angle and rudder position perturbation commands for DPIFS simulation are in ( \text{UMAG}(1,1) ) and ( \text{UMAG}(2,1) ).</td>
</tr>
<tr>
<td>7-ALT SEL</td>
<td>Altitude select command system for longitudinal dynamics. Altitude perturbation command is in ( \text{UMAG}(1,1) ).</td>
</tr>
<tr>
<td>9-APR LOCI</td>
<td>Localizer tracking model for lateral dynamics. Perturbation lateral position offset is in ( \text{UMAG}(1,1) ). Yaw intercept angle is in ( \text{UMAG}(2,1) ). ( \text{USTRT}(1,1) ) and ( \text{USTRT}(2,1) ) must be at 0.0. ( \text{USTP}(1,1) ) and ( \text{USTP}(2,1) ) must be at desired final time.</td>
</tr>
</tbody>
</table>
12-ROLL HOLD  Roll angle and rudder angle constant command system. Roll and rudder angle perturbation commands are in UMAG(1,1) and UMAG(2,1).

14-APR GS  Glideslope position error (d) tracking command system. UMAG(1,1) contains the d position offset at beginning of simulation. USTRT(1,1) should be 0.0. z dynamics in F and G are replaced with d dynamics in HDCON.

15-APR LOCR  Lateral position (y) localizer outer loop command system using the HDG SEL command system as the inner loop structure. Same simulation requirements as ICHOSE = 9.

16-APR LOCP  Lateral position (y) localizer outer loop command system using the ROLL HOLD command system as the inner loop structure. Same simulation requirements as ICHOSE = 9.

METERS  Logical scalar indicating when units of control law should be changed to meters and radians.

HAC, DAC  Matrices packed by columns into one-dimensional arrays. These matrices are determined in HADAC.

NHAC, NDAC  Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N.

GR  Gravity in ft/sec^2.

N, M, L, NM, MM  Integer values indicating initial values for dimension of matrices, N = 12, M = 4, L = 4, NM = 2, MM = 2. Changed upon return.
GKPSI  Outer loop yaw angle guidance gain for HDG SEL and APR LOCR autopilot modes.

SKY  Outer loop lateral position guidance gain for APR LOCR and APR LOCP autopilot modes.

XL, YL  Label arrays for printout in DPIFS. Some of the elements in the array are altered upon return.

c.  Output Arguments

H, D, FM,
GM, HM, DM  Matrices packed by columns into one-dimensional arrays.
NH, ND,
NFM, NGM,
NHM, NDM  Two-dimensional integer vector holding the number of rows and columns after the letter N.
XR, UR, YR,
XMR, UMR  Logical arrays containing TRUE and FALSE information for subroutine REMAT.
N, M, L,
NM, MM  Integer values indicating maximum dimensions for matrices.
XL, YL  Label values for plots matching choice of AUTOPilot mode.

d.  Common Blocks

None

e.  Error Messages

If the W9 matrix is too small the statement "THE W9 MATRIX IS TOO SMALL" is printed.

if ICHOSE = 1 and ICH(2,9) is FALSE or ICH(2,13) is FALSE then "ERROR ?? ICH BETA BAD" is printed.
If DEBUG is TRUE the following matrices are printed out during the course of program execution.

**VXYZ** - A $3 \times N$ matrix containing the $\Delta \dot{x}$, $\Delta \dot{y}$, $\Delta \dot{z}$ dynamics extracted from the $F$ matrix.

**LAMD** - A $1 \times N$ vector containing the row in $F$ that represents the $\Delta \dot{d}$ dynamics.

**H(HCON), D(DCON), FM(FMCO), GH(GMCO), HM(HMCO), DM(DMCO)**

f. Subroutines Employed by HDCON

**ORACLS** - MULT, NULL, PRNT, SCALE

**PIFLIB** - PRNTITL, PRNT

g. Subroutines Employing HDCON

**PIFLIB** - PIFCGT

h. Concluding Remarks

None
V. INTERPOLATE AERODYNAMIC DATA (INTERP)

1. PURPOSE

Given the aerodynamic coefficients and trim conditions of two flight conditions at different forward velocities, the subroutine INTERP interpolates between the data points to find estimated intermediate aerodynamic coefficients and trim conditions.

2. USAGE

a. Calling Sequence

   CALL INTERP (VTP)

b. Input Arguments

   VTP  Real scalar containing the forward velocity in fps at which the interpolation is to occur

   a. Calling Sequence

   CALL INTERP (VTP)

b. Input Arguments

   VTP  Real scalar containing the forward velocity in fps at which the interpolation is to occur

   c. Output Arguments

   None

d. COMMON Blocks

   AERO, AER1

   The interpolation overwrites the values in the AERO common block. The AERO common block is used elsewhere in the program called by PIFCGT

e. Error Messages

   None

f. Subroutine Employed by INTERP

   PIFLIB - PRNTITL

g. Subroutines Employing INTERP

   PIFCGT

h. Concluding Remarks

   None
W. MULTIPLY A TRANSPOSE TIMES B TIMES A (MATBA)

1. PURPOSE
Form Matrix Product $A^TBA$ where $B$ must be symmetric.

2. USAGE
a. Calling Sequence
CALL MATBA (A,NA,B,NB,C,NC,Wl,NWl)

b. Input Arguments
A,B Input matrices packed by columns in one-dimensional arrays; not destroyed. $B$ must be symmetrical.

NA,NB Two-dimensional vectors giving number of rows and columns of respective matrices; for example:
NA(1) = number of rows of $A$
NA(2) = number of columns of $B$
Not destroyed upon return.

Wl Input workspace vector of length $NB(1) \times NA(2)$ or greater

NWl Vector of length two for workspace

c. Output Arguments
C Product $A^TBA$ packed by columns in one-dimensional array.

NC Two-dimensional vector containing rows and columns in C.
NC(1) = NA(2)
NC(2) = NA(2)

d. COMMON Blocks
None

e. Error Messages
None
f. Subroutines employed by MATBA:

MULT

g. Subroutines employing MATBA:

PIFLIB - CFWM

h. Concluding Remarks

B is not checked to see if it is symmetric.
X. REARRANGE MATRIX (MOVEALL)

1. PURPOSE

Subroutine MOVEALL rearranges the rows (columns) of a matrix with the option of taking a subset of the rows (columns).

2. USAGE

a. Calling Sequence

CALL MOVEALL (A,NA,B,NB,IDX,IHOW)

b. Input Arguments

A Matrix to which manipulations are to be performed packed by columns in one dimensional array; not destroyed.

NA Two-dimensional vector giving number of rows and columns in A.

NA(1) = Number of rows in A
NA(2) = Number of columns in A

Not destroyed.

IHOW Flag for type of manipulation

IHOW = 1 - work with rows,
IHOW = 2 - work with columns.

IDX Position dependent index vector. IDX(1) contains the number of the row (column) in A that will become the first row (column) in B and so forth. All unused positions in IDX must be zero especially when working columns. Therefore if IDX has its first 5 elements non-zero then B will contain only 5 rows (columns) from A. IDX thus allows rearranging and deleting rows (columns) at the same time.

c. Output Arguments

B Contains the manipulated results from A packed by columns in a one-dimensional array.
NB Two-dimensional vector contain dimensions of B

NB(1) = Number of rows in B

NB(2) = Number of columns in B

d. COMMON Blocks

None

e. Error Messages

If NA(1) < NB(1) or NA(2) < NB(2), the message "DIMENSION ERROR IN MOVEALL
NA = _____ _____, NB = _____ _____" is printed and the program is returned
to the calling point.

f. Subroutines employed by MOVEALL

None

g. Subroutines employing MOVEALL

PIFLIB - REMAT, FGAERO

h. Concluding Remarks

Example: IDEX = [3, 1, 2, 0]

IHOW = 1

\[
A = \begin{bmatrix}
  a & b & c & d \\
  e & f & g & h \\
  n & p & q & r \\
  t & v & w & x
  \end{bmatrix}
\]

then

\[
B = \begin{bmatrix}
  n & p & q & r \\
  a & b & c & d \\
  e & f & g & h
  \end{bmatrix}
\]

It is good practice to zero out IDEX before constructing it since non-zero
elements detected after a zero element causes problems.
Y. PIF EIGENVALUE PLACEMENT (PIFEIG)

1. PURPOSE

The objective of PIFEIG is to compute a closed-loop model matrix, FM, with stable eigenvalues. Subroutine PIFEIG computes a desirable closed-loop model matrix for the implicit model following design method in the subroutine PIFPLC. Ultimately, PIFPLC computes an initial stabilizing feedback gain for the subroutine DREG.

If ICH(2,11) is TRUE, PIFEIG computes a random feedback gain, GK, using the subroutine RANF. If ICH(2,11) is FALSE, PIFEIG assumes GK contains an initial starting gain. The eigenvalues and eigenvectors of the closed-loop plant, F+G*GK are computed using the subroutine FREEF and satisfy the following relationship

\[
T \lambda T^{-1} = F + G \times GK
\]  

(1)

\(\lambda\) is a complex diagonal matrix containing the eigenvalues, \(\lambda_i\). \(T\) is a complex matrix containing the eigenvectors. If IE(i) is equal to 1, the \(i^{th}\) eigenvalue of \(\lambda\) is adjusted as follows:

- eigenvalue \(\lambda_i = a_i + j b_i\)
- stabilized \(a_i = -|a_i|\)

After eigenvalue stabilization, the real and imaginary parts of the eigenvector matrix, \(T = W9(LW8) \pm j W9(LW10)\), are used to form the real matrix, FM, as follows,

\[
\begin{bmatrix}
W9(LW8) & W9(LW10) \\
-W9(LW10) & W9(LW8)
\end{bmatrix}
\begin{bmatrix}
W9(LW1) & W9(LW2) \\
-W9(LW2) & W9(LW1)
\end{bmatrix}
\begin{bmatrix}
W9(LW8) & W9(LW10) \\
-W9(LW10) & W9(LW8)
\end{bmatrix}^{-1} =
\begin{bmatrix}
FM & 0 \\
0 & FM
\end{bmatrix}
\]  

(2)
The matrix W9(LW1) is a diagonal matrix containing the real part of the stabilized eigenvalues of F+G*GK. The matrix W9(LW2) is a diagonal matrix containing the imaginary part of the stabilized eigenvalues. The partitioned matrix manipulations make it possible to perform complex arithmetic without using complex numbers. The constructed FM matrix has the stable adjusted eigenvalues and the same eigenvectors as F+G*GK.

2. USAGE

a. Calling Sequences

CALL PIFEIG (F, NF, FM, NFM, W9, NW9DIM, DEBUG, IFREIG, ER, EI, ERV, IEV, G, NG, ICH, GK, NGK)

b. Input Arguments

F, G, GK Matrices packed by columns into one-dimensional arrays; F and G are not destroyed upon return. If ICH (2,11) is FALSE, GK contains an initial starting gain and is not destroyed upon return.

IE Integer vector of dimension at least NF(1). If the eigenvalue adjustment procedure in PIFEIG is to be performed on the \( i^{th} \) eigenvalue then \( IE(i) \) must be 1. If \( IE(i) \) is 0, the \( i^{th} \) eigenvalue in FM equals the \( i^{th} \) eigenvalue of F+G*GK. If the \( i^{th} \) eigenvalue is complex and the \( i^{th}+1 \) eigenvalue is the complex conjugate, then \( IE(i) \) and \( IE(i+1) \) must contain the same integer number (1 or 0). The order in IE must apriori agree with the order of the eigenvalues computed by FREEF. The IE feature is useful if a plant eigenvalue is stable but uncontrollable. If ICH(2,11) is TRUE, all elements in IE are automatically set to 1.
ER, EV  Real work vectors of dimension at least NF(1). During computation the adjusted real parts of the eigenvalues are placed in ER and the imaginary parts of the eigenvalues are placed in EV.

IEV  Integer vector of dimension at least NF(1). Each of the locations must contain the integer 1. If ICH(2,11) is TRUE, all IEV elements are automatically set to 1.

ERV  Real work matrix of dimension NF(1)*NF(1). ERV is not used by PIFEIG as long as IEV has all 1's.

W9  Work matrix packed by columns into a one-dimensional array. Dimension must be at least 16*(NF(1)*NF(2)+1)

ICH  Two-dimensional logical matrix dimensioned 10 x 15. Subroutines in PIFCGT are called depending on the TRUE or FALSE status of elements in ICH.

NF, NG, NGK  Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N.

DEBUG  Logical scalar indicating the following:
TRUE: Print out debug information
FALSE: Do not print out debug information

IFREIG  The results of the construction of FM can be observed by setting this variable to 0, 1, or 2
2- Compute and print eigenvalues and eigenvectors of FM
1- Compute and print eigenvalues of FM
0- No eigenvalue computations
c. Output Arguments

FM     Matrix packed by column into a one-dimensional array. FM contains the desired closed-loop PIF plant used in PIFPLC.

NFM    Two-dimensional integer vector holding the number of rows and columns of the matrix FM, NFM(1)=NF(1), NFM(2)=NF(2).

d. COMMON Blocks

None

e. Error Messages

If the W9 matrix is insufficiently large, the message "The W9 matrix is too small" is printed and the program stops. If DEBUG is TRUE, the following matrices are printed out during the course of the computation:

GKRA - the random or input GK gain matrix

REAL - the real part of the eigenvalues of F+G*GK

IMAG - the imaginary part of the eigenvalues of F+G*GK

ETMA - the 2*NF(1) x 2*NF(1) matrix containing the partitioned eigenvectors of F+G*GK

ER - the adjusted real part of the eigenvalues of F+G*GK

EI - the adjusted imaginary parts of the eigenvalues of F+G*GK

REAL - the real part of the eigenvalues to be placed in FM

IMAG - the imaginary part of the eigenvalues to be placed in FM.

EIMA - the 2*NF(1) x 2*NF(1) matrix containing the partitioned adjusted eigenvalues of FM.

IDEN - the inverse of ETMA is multiplied by ETMA and the result is printed out as the matrix IDEN.
The accuracy of the inversion can be determined by judging how close IDEN is to an identity matrix.

TLTI - the 2*NFI x 2*NFI matrix obtained by multiplying
ETMA*EIMA*ETMA. If the subroutine GELIM produces an error in inverting ETMA, the message "GELIM IN EIGP IERR = XXXX" is printed.

f. Subroutines Employed by PIFEIG

ODACLS - NULL, EQUIATE, MULT, ADD, GELIM, UNITY, SCALE, DETFAC, EIGEN

PIFLIB - DIAGPAR, DIAGPUT, CORNER, FREEF, RESPON, SMALL, PRNT, PRNTITL

g. Subroutines Employing PIFEIG

PIFGET

h. Concluding Remarks

In all applications of PIFEIG by the program PIFICGT, the locations in IE and IEV are all set to 1. It is not necessary to use PIFEIG and PIPPLC if DSTAB in ASYMREG in DREG finds an initial stabilizing gain for the closed-loop plant.
1. PURPOSE

PIFFRE takes the matrix log of the closed-loop PIF system matrix, PHICL, then computes the eigenvalues and eigenvectors of the resulting matrix.

2. USAGE

a. Calling Sequences

CALL PIFFRE (DELT, W9, DEBUG, NW9DIM, PHICL, NPHC, IFREEP, ICH, ELOG, MLOG)

b. Input Arguments

PHICL Matrix packed by columns into a one dimensional array; not destroyed upon return.

NPHC Two-dimensional integer vector holding the number of rows and columns of the matrix PHICL.

ELOG Real scalar convergence test for LOGX1. Recommend 1.0E-10.

MLOG Integer scalar indicating the maximum number of iterations used in the LOGX1 subroutine. Recommend 8.

IFREEP Integer scalar used to determine eigenvalue eigenvector calculations

=0 Do not perform calculations
=1 Compute eigenvalues
=2 Compute eigenvalues and eigenvectors

W9 Work matrix packed by columns into a one-dimensional array. Dimension must be at least 3*(NPHC(1)**2+1)
ICH Two-dimensional logical matrix dimensioned 10 x 15.
Subroutines in PIFCGT are called depending on the TRUE or FALSE status of elements in ICH.

DEBUG Logical scalar indicating the following:
TRUE: Print out debug information
FALSE: Do not print out debug information

DELT Real scalar containing the sampling time

NW9DIM Integer scalar containing the maximum dimension of the W9 matrix.

c. Output Arguments
None

d. COMMON Blocks
None

e. Error Messages
If the W9 matrix is insufficiently large, the message "THE W9 WORK MATRIX IS TOO SMALL" is printed and the program stops.

f. Subroutines Employed by PIFFRE
ORACL5 - EQUATE, EXPADE, MULT, NORMS, SCALE, SUBT, UNITY, EIGEN
PIFLIB - LOGX1, FREEF, RESPON

g. Subroutines Employing PIFFRE
PIFCGT

h. Concluding Remarks
None

100
1. PURPOSE

Subroutine PIFG performs three functions:

- The feedback gain, Riccati equation solution, and feedforward matrix solution are used to construct the matrices in the implementable PIF control law.
- The body-axis aircraft states used for feedback are replaced with available measurements. Gain matrices in the implementable PIF control law are reconfigured to accommodate the measurements.
- IF METERS is TRUE, the units in the PIF control law are changed from feet and degrees to meters and radians. Gain matrices in the implementable PIF control law are reconfigured to accommodate the change in units and printed out. The actual values of the PIF control law gains do not change units at the end of PIFG execution if METERS is TRUE. Program execution proceeds as follows:

The feedback gain, \( \text{GK} \), is partitioned into three gain sets and they are extracted from \( \text{GK} \).

\[
\text{GK} = \begin{bmatrix}
K1 & & \\
& K2 & \\
& & K3
\end{bmatrix}
\]  

(1)

The Riccati equation solution, \( \text{X} \), is partitioned into nine individual matrices and three of them (PZZ, PUZT, PXZT) are extracted from \( \text{X} \).

\[
\text{X} = \begin{bmatrix}
PXX & PXU & PXZ \\
PXUT & PUU & PUZ \\
PXZT & PUZT & PZZ
\end{bmatrix}
\]  

(2)
The matrix $A$ is computed

$$A = PZZ^{-1} \ast (PUZT \ast S22 + PXZT \ast S12)$$  \hspace{0.5cm} (3)$$

and used to construct $S42$

$$S42 = K3 \ast A - K2 \ast S22 - K1 \ast S12$$  \hspace{0.5cm} (4)$$

The matrix $HAC$ is computed

$$HAC = -K2 \ast S21$$  \hspace{0.5cm} (5)$$

The new measurements, $z$, are related to the aircraft states by constructing the $T$ matrix,

$$\begin{bmatrix} z \\ u \end{bmatrix} = \begin{bmatrix} HX & DX \\ 0 & I \end{bmatrix} \begin{bmatrix} X \\ u \end{bmatrix}$$  \hspace{0.5cm} (6)$$

$$T = \begin{bmatrix} HX & DX \\ 0 & I \end{bmatrix}$$  \hspace{0.5cm} (7)$$

The feedforward matrices are reconfigured to accommodate the new measurements

$$\begin{bmatrix} S11 & S12 \\ S21 & S22 \end{bmatrix}_{\text{new}} = \begin{bmatrix} HX & DX \\ 0 & I \end{bmatrix} \begin{bmatrix} S11 & S12 \\ S21 & S22 \end{bmatrix}_{\text{old}}$$  \hspace{0.5cm} (8)$$

The feedback gains $K1$ and $K2$ are reconfigured to accommodate the new measurements

$$[K1 \ K2]_{\text{new}} = [K1 \ K2]_{\text{old}} \begin{bmatrix} HX & DX \\ 0 & I \end{bmatrix}^{-1}$$  \hspace{0.5cm} (9)$$

102
The PIF gains S31 and S41 are constructed

\[ S31 = I + \text{DELT} \cdot K2 \]  
\[ S41 = \text{DELT} \cdot K3 \]

The resulting implementable PIF structure has the following form

\[ u_{k+1} = u_k + \text{DELT} \cdot v_k + S21 \left( u_{m,k+1} - u_{m,k} \right) \]  
\[ e_k = x_k - S11 \cdot x_{m,k} \]

\[ v_k = S31 \cdot v_{k-1} + \text{HAC}(x_{m,k} - x_{m,k-1}) + S32(e_k - e_{k-1}) \]

\[ + S41(Hx_{k-1} + Du_{k-1} - Hx_{m,m,k-1} - Du_{m,m,k}) + S42(u_{m,k+1} - u_{m,k}) \]  

The gain elements in the implementable PIF structure are modified if the METERS option is TRUE and printed out. The gain matrices at the end of the program execution remain in the units of feet and degrees.

2. USAGE

a. Calling Sequence

CALL PIFG (S11, NS11, S12, NS12, S21, NS21, S22, NS22, GK, NGK, X, NX, S31, NS31, S32, NS32, HAC, NHAC, S41, NS41, S42, NS42, DELT, DEBUG, W9, NW9DIM, HX, DX, NHX, NDX, METERS, ICHOOSE)

b. Input Arguments

S11, S12,
S21, S22,
GK, X,
HX, DX Matrices packed by columns into one-dimensional arrays; HX, DX, S21, and S22 are not destroyed upon return
NS11, NS12,
NS21, NS22,
NGK, NX,

NHX, NDX  Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N.

METERS  Logical scalar indicating when units of the control law should be changed to meters and radians.

ICHOSE  Integer scalar which indicates which command system is being designed.

W9  Work matrix packed by columns into a one-dimensional array. Dimension must be at least 9*(NX(1)*NX(2)+1).

DEBUG  Logical scalar indicating the following:
        TRUE:  Print out debug information
        FALSE: Do not print out debug information.

DELT  Real scalar containing the sampling time.

NW9DIM  Integer scalar containing the maximum dimension of the W9 matrix.

c. Output Arguments

S11, S31,
S32, HAC,
S41, S42,

S21  Matrices packed by columns into one-dimensional arrays constituting the PIF control law feedback and feedforward gains.

NS11, NS31,
NS32, NHAC,
NS41, NS42,

NS21  Two-dimensional integer vector holding the number of rows
and columns of the matrix shown after the letter N.

d. COMMON blocks

None

e. Error Messages and Printout

If the W9 matrix is insufficiently large the message "THE W9 WORK
MATRIX IS TOO SMALL" is printed and the program stops. If the value
in ICHOSE is not a recognized command system then the following is
printed "IN PIFG WITH METERS TRUE, ICHOSE = XXX IS NOT A KNOWN COMMAND
MODE. CHANGE PIFG TO ACCOMMODATE ICHOSE VALUE." IF DEBUG is TRUE, the
following matrices are printed during the course of program execution:
K1, K2, K3, PZZ, PZZ^{-1}(PZ-1), PUZT, PXZT, A, T, T^{-1}(T-1)
S11B - The S11 feedforward matrix before the new measurements are
accommodated.
S12B - The S12 feedforward matrix before the new measurements are
accommodated.
The following are always printed out:
Before changing units
S11, S12, S21, S22, S31, S32, S41, S42, HAC
After changing units if METERS is TRUE
S31M, S21M, HACM, S32M, S11M, S41M, S42M

f. Subroutines Employed by PIFG

ORACLS - MULT, ADD, SUBT, SCALE, UNITY, GELIM, JUXTC, EQUATE, DETFAC
PIFLIB - DIAGPAR, CORNER, PRNT, PRNTITL

g. Subroutines Employing PIFG

PIFLIB - PIFCGT

h. Concluding Remarks

The derivation used to obtain Eqs. 3 to 14 is shown in Chapter III
of Ref. 1.
BB. CONSTRUCT THE PIF MODELS (PIFMODL)

1. PURPOSE

The purpose of PIFMODL is to construct the following matrices

\[
\Phi = \begin{bmatrix} F & G & 0 \\ 0 & 0 & 0 \\ H & D & 0 \end{bmatrix}
\]

(1)

\[
\Gamma = \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix}
\]

(2)

\[
\Theta = \begin{bmatrix} A & AM & 0 \\ AM^T & R & 0 \\ 0 & 0 & QZ \end{bmatrix}
\]

(3)

\[
\Sigma = RD
\]

(4)

\[HX = I\]

(5)

\[DX = 0\]

(6)

HX and DX are NF(1) and NF(1) x NG(2) aircraft observation matrices. The order of the states is \(\Delta u \, \Delta w \, \Delta q \, \Delta \theta \, \Delta z \, \Delta x \, \Delta v \, \Delta r \, \Delta \phi \, \Delta \psi \, \Delta y\). If ICH(2,10) is TRUE, \(\Delta u\) observation in HX and DX is replaced with \(\Delta a_x\) observation. If ICH(2,9) is TRUE, \(\Delta v\) observation in HX and DX is replaced with \(\Delta a_y\) observation. If ICH(2,10) is TRUE, \(\Delta w\) observation in HX and DX is replaced with \(\Delta a_z\) observation. Variables in XL are changed depending on whether or not METERS, ICH(2,8), ICH(2,9) and ICH(2,10) are TRUE.
2. USAGE

a. Calling Sequences

CALL PIFMODL(F, NF, G, NG, H, NH, D, ND, PHI, NPH, CAMA, NGA, Q, NQ,
R, NR, AM, NAM, QZ, NQZ, QHAT, NQH, RD, NRD, RHT, NRH, ZERO, NZERO,
HX, NHX, DX, NDX, HAC, NHAC, DAC, NDAC, W9, NW9DIM, DEBUG, METERS,
ICH, XL, UL, YL)

b. Input Arguments

F, G,
H, D,
Q, AM,
R, QZ,
RD Matrices packed by columns into one-dimensional arrays; not
destroyed upon return.

HAC, DAC Matrices packed by columns into one-dimensional arrays; not
destroyed upon return. These are the observation matrices
for the body mounted accelerometers computed in HADAC.

XL, UL, YL Two dimensional arrays containing plotting labels for the
states, controls, and commanded outputs, respectively.

NF, NG, NH,
ND, NQ, NR,
NAM, NQZ,
NZERO, NHAC,
NDAC Two-dimensional integer vector holding the number of rows
and columns of the matrix shown after the letter N.

W9 Work matrix packed by columns into a one-dimensional array.
Dimension must be at least $2 \times (NF(1) + NG(2) + NH(1))^{*2+1}$

ZERO Work matrix packed by columns into a one-dimensional array.
All elements must be 0.0. Dimensions must be at least \( NPH(1) \times NPH(2) + 1 \).

**ICH**

Two-dimensional logical matrix dimensioned 10 x 15. Subroutines in PIFCGT are called depending on the TRUE or FALSE status of elements in ICH.

**DEBUG**

Logical scalar indicating the following:
- **TRUE**: Print out debug information.
- **FALSE**: Do not print out debug information.

**NW9DIM**

Integer scalar containing the maximum dimension of the W9 matrix.

c. **Output Arguments**

QHAT, RHAT,

HX, DX,

PHI, GAMA Matrices packed by columns into one dimensional arrays.

NQH, NRH,

NPH, NGA Two dimensional integer vectors holding the number of rows and columns of the matrix shown after the letter N.

XL Two dimensional array containing labels to be used in plotting; some are changed upon return depending on logical parameters METERS and ICH.

d. **COMMON Blocks**

None

e. **Error Messages**

If the W9 matrix is insufficiently large the message "THE W9 WORK MATRIX IS TOO SMALL" is printed and the program stops. If DEBUG is true then the following is printed out during program execution:
f. Subroutines Employed by PIFMODL

ORACLS - JUXTC, UNITY, JUXTR, TRANP, EQUATE

PIFLIB - DIAGPAR, CORNER, PRNT, PRNTITL

g. Subroutines employing PIFMODL

PIFCGT

h. Concluding Remarks

The perturbation relationship between the body mounted accelerometer outputs, $\Delta a_x$, $\Delta a_y$, and $\Delta a_z$, and the aircraft body-axis states is derived in Ref. 1, Appendix B.
CC.  PIF EIGENVALUE PLACEMENT (PIFPLC)

1. PURPOSE

In Ref. 6 a technique is given for computing a feedback gain which places eigenvalues and eigenvectors of a plant to those dictated by an ideal model. The purpose of PIFPLC is to compute a stabilizing feedback gain for the PIF control law using the eigenvalue eigenvector placement procedure in Ref. 6. The ideal model (PHICL) is computed in PIPEIG. The subroutine is only applicable to PIF model matrices. The subroutine first extracts the H matrix from the discrete PIF model matrix PHI. The discrete feedforward matrix equation

\[
\begin{bmatrix}
\Phi & \Gamma \\
H & 0
\end{bmatrix}
\begin{bmatrix}
S_{11} \\
S_{21}
\end{bmatrix} =
\begin{bmatrix}
S_{11} & \text{PHICL} \\
& H
\end{bmatrix}
\]  

is solved for \( S_{11} \) and \( S_{21} \) using CGTPIF. The stabilizing feedback gain is computed as

\[
G_K = S_{21} S_{11}^{-1}
\]  

2. USAGE

a. Calling Sequence

CALL PIFPLC (PHI, NPH, NH, GAM, NGA, W9, NW9DIM, X, PHICL, NFHC, ACL, GK, NGK, DEBUG, ELOG, MLOG, ICH, DELT, NS11, NS12, NS21, NS21, NS31, NS32, NS41, NS42, ICGT, TOL, ITER, ECGT)

b. Input Arguments

PHI, GAM, PHICL        Matrices packed by columns into one-dimensional arrays;

110
not destroyed upon return. PHI contains the discrete PIF system matrix, GAMA contains the discrete PIF control effect matrix, and PHICL contains the desired closed-loop plant matrix (FM in PIFEIG)

NS11, NS12,
NS21, NS22,
NS31, NS32,
NS41, NS42 Two dimensional work integer vectors.

ELOG Real scalar convergence test for LOGX1 subroutine.
X,ACL Work matrices of dimension at least NPH(1) * NPH (2) + 1
W9 Work matrix packed by columns into a one-dimensional array. Dimension must be at least 9*(NPH(1)**2+1) + 4 *((NPH(1) + NGA(2) + 2)**2+1) + 2*((NPH(1) + NGH(2) + 1)**2+1)
ICH Two-dimensional logical matrix dimensioned 10 x 15. Subroutines in PIFCGT are called depending on the TRUE or FALSE status of elements in ICH.

NH, NPH,
NGA, NPHC Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N.
TOL Real scalar indicating convergence criteria for iterative refinement in SOLVER. Recommend 1.0E-10
ITER Integer scalar indicating maximum number of interactive refinements in SOLVER. Recommend 10.
ECGT Real scalar indicating convergence criteria in SOLVER for inverting a matrix. Recommend 1.0E-10
ICGT Five dimensional integer option vector defined in CGTPIF.
ICGT(1) is set to 2, ICGT(4) is set to 0 and ICGT(5) is set to 0 automatically by the subroutine before calling PIFCGT. The original values of ICGT are restored at the end of subroutine execution.

**DEBUG**
Logical scalar indicating the following
TRUE: Print out debug information
FALSE: Do not print out debug information

**DELT**
Real scalar containing the sampling time.

**NW9DIM**
Integer scalar containing the maximum dimension of the W9 matrix.

c. **Output Arguments**

**GK**
Matrix packed by columns into a one-dimensional array containing the stabilizing feedback gain.

**NGK**
Two dimensional integer vector holding the number of rows and columns of the GK matrix.

d. **COMMON Blocks**

None

e. **Error Messages**

If the W9 matrix is insufficiently large the message "THE W9 MATRIX IS TOO SMALL" is printed and the program stops. If the subroutine GELIM produces an error in inverting Sll, the message "GELIM ERROR IN PIFPLC IERR = XX" is printed. If DEBUG is TRUE then the following is printed out during the course of program execution:

- **HDO** - The H matrix extracted from the PIF model matrix PHI.
- Matrices from the execution of CGTPIF (see CGTPIF document).
- **S11G** - The S11 feedforward matrix solution of the feedforward matrix equation.
S21G - The S21 feedforward matrix solution of the feedforward matrix equation.

I21G - S11 is multiplied times S11⁻¹ and the results, which should be an identity matrix, is printed out.

The equivalent continuous-time closed-loop eigenvalues and eigenvectors of PHI + GAMMA * GK.

f. Subroutines Employed by PIFPLC

ORACL5 - NULL, EQUATE, UNITY, GELIM, MULT, ADD, DETFAC, SCALE, SUBT, NORMS, BCKMLT, HSHLDR, SCHUR, SYSSLV, EXPADE, EIGEN

PIFLIB - PRNT, PRNTITL, EXTR, FREEF, RESPON, LOGX1, CGTPIF, AXBMXC,

SHRSOL

g. Subroutines Employing PIFPLC

PIFLIB - PIFCGT

h. Concluding Remarks

It is not necessary to use PIFEIG and PIFPLC if DSTAB in ASYMREG in DREG finds an initial stabilizing gain.
1. PURPOSE

Subroutine PRNT prints a single matrix with a descriptive heading on the same page. The descriptive heading is printed before each matrix and is of the form "NAM MATRIX NC(1) ROWS NA(2) COLUMNS." The matrix is printed by rows for seven columns at a time using the format, (I4,1P7E16.7), where the first number is an integer depicting the row number. A heading of integers corresponding to the column number is printed above the matrix.

2. USAGE

a. Calling Sequence

CALL PRNT (A,NA,NAM,IOP)

b. Input Arguments

A       Matrix packed by columns in a one-dimension array
NA      Two-dimensional vector giving the number of rows and columns of the matrix A:
        NA(1) = Number of rows of A;
        NA(2) = Number of columns of A
NAM     Hollerith characters giving matrix name. Generally NAM should contain 4 Hollerith characters and be written in the argument list as 4HXXXX.
IOP     Scalar print control parameter
        1. print heading and matrix on same page
        2. print heading and matrix on next page

c. Output Arguments

None

d. COMMON Blocks

None
e. Error Messages

If \( NA(1) < 1 \) or \( NA(2) < 1 \) the message "PROBLEM WITH NUMBER OF ROWS OR NUMBER OF COLUMNS NA(1) = ____ ____ , NA(2) = ____ ____" is printed, and the program is returned to the calling point.

f. Subroutine Employed by PRNT

None

g. PIFLIB subroutines Employing PRNT

PIFCGT, CGTPIF, HCON, CFWM, PIFMODL, PIFPLC, REMAT, DISCMOD, QRMP1, QRMHAT, DREG, PIFG, DPIFS, FGAERO, HADAC, PIFEIG

h. Concluding Remarks

None
1. PURPOSE

Subroutine PRNTITL is used to print the name of a subroutine at the beginning and end of each pass through the subroutine.

2. USAGE

a. Calling Sequence

CALL PRNTITL (NAM1, NAM2)

b. Input Arguments

NAM1  Hollerith characters for title to be printed out. The first ten Hollerith characters in the title are placed in NAM1 in the form 10HXXXXXXXXXX.

NAM2  Hollerith characters for title to be printed out. The second ten Hollerith characters in the title are placed in NAM2 in the form 10HXXXXXXXXXX.

c. Output Arguments

None

d. COMMON Blocks

None

e. Error Messages

None

f. Subroutines Employed by PRNTITL

None
g. PIFLIB subroutines Employing PRNTITL

DIMSS, HDCON, CFWM, PIFMODL, PIFPLC, REMAT, DISCMOD, QRMP1, QRMHAT,
DREG, PIFG, DPIFS, FGAERO, HADAC, PIPEIG, RUNINFO, INTERP, PIFFRE

h. Concluding Remarks

None
1. PURPOSE
Subroutine PRNT2 prints out one matrix in columns of seven with labels or numbers, or it prints out two matrices in columns of seven with labels for comparison. PRNT2 also has the option of printing a third one dimensional array as a counter instead of row numbers.

2. USAGE
a. Calling Sequence
CALL PRNT2 (A, NA, NAM, B, NB, NAMB, IOP, C, LABEL, LABEL1, LABEL2, M)

b. Input Arguments
A  Matrix packed by columns into a one-dimensional array.
    A is the first matrix to be printed.

B.  Matrix packed by columns into a one-dimensional array.
    B is the second matrix to be printed.

NA  Two-dimensional vector giving the number of rows and columns of matrix A.

NB  Two-dimensional vector giving the number of rows and columns of matrix B.
    NA(1) - number of rows of A
    NA(2) - number of columns of A
    NB(1) - number of rows of B = NA(1)
    NB(2) - number of rows of B = NA(2)

NAM  Hollerith characters giving matrix A's name. Generally NAM should contain 4 Hollerith characters and be written in the argument list as 4HXXXX.
This is the name of the matrix B and follows the same rules as NAM.

IOP  Scalar print control parameter.

IOP = 0    PRINT ONE MATRIX WITH LABELS - SAME PAGE
IOP = 1    PRINT ONE MATRIX WITH LABELS - NEW PAGE
IOP = 2    PRINT ONE MATRIX WITH NUMBERS - SAME PAGE
IOP = 3    PRINT ONE MATRIX WITH NUMBERS - NEW PAGE
IOP = 4    PRINT TWO MATRICES WITH LABELS - SAME PAGE
IOP = 5    PRINT TWO MATRICES WITH LABELS - NEW PAGE

C    One-dimensional counter array; dimension = NA(1)

LABEL     LABELS for Array A, dimensioned (M,2)
LABEL1    LABELS for Array B, dimensioned (M,2)
LABEL2    LABELS for Array C, dimensioned (1,2)

M    First Dimension of LABEL array = NA(2)

c. Output Arguments

None

d. COMMON Blocks

None

e. Error Messages

If NA(1) < 1 or NA(2) < 1 the message "PROBLEM WITH THE NUMBER OF ROWS OR NUMBER OF COLUMNS NA(1) = _____, NA(2) = _____" is printed, and the program is returned to the calling point.

f. Subroutines Employed by PRNT2

None

g. Subroutines Employing PRNT2

PIFLIB - DPIFS
h. Concluding Remarks

For printing one matrix - B, NB, NAMB and LABEL1 are dummy arguments, but they must be in the call to PRNT2. For printing without labels - C, LABEL, LABEL1, and LABEL2 are dummy arguments, but they must be in the call to PRNT2.
1. PURPOSE

The purpose of QRMHAT is to compute the sampled-data regulator corresponding to a linear time-invariant continuous system and its associated linear quadratic cost function

\[ \dot{x} = Fx + Cu \quad (1) \]

\[ J = \int_{0}^{\infty} x^T Qx + u^T Ru \, dt \quad (2) \]

The control, \( u \), is assumed to be constant over the sampling interval \( \Delta t \). The discrete plant and cost function are given by

\[ x_{k+1} = \text{PHI} \, x_k + \text{GAMA} \, u_k \quad (3) \]

\[ J = \sum_{k=0}^{\infty} (x_k^T \text{QHAT} \, x_k + 2x_k^T \text{AMHAT} \, u_k + u_k^T \text{RHAT} \, u_k) \quad (4) \]

The matrices in the sampled-data regulator problem are computed using the following equations

\[ \text{PHI} = \sum_{k=0}^{\infty} \frac{(F \Delta t)^k}{k!} = e^{F \Delta t} \quad (5) \]

\[ \text{GAMA} = \int_{0}^{\Delta t} \frac{(FT)^k}{k!} \, d\tau = \int_{0}^{\Delta t} e^{FT} \, d\tau \quad (6) \]

\[ \text{QHAT} = \int_{0}^{\Delta t} e^{FT} \, Q e^{FT} \, d\tau \quad (7) \]

\[ \text{AMHAT} = \int_{0}^{\Delta t} e^{FT} Q (\int_{0}^{T} e^{FS} \, ds) d\tau \quad (8) \]

\[ \text{RHAT} = \Delta t \, R + G^T \left[ \int_{0}^{\Delta t} (\int_{0}^{T} e^{AS} \, ds) Q (\int_{0}^{T} e^{AT} \, ds) d\tau \right] G \quad (9) \]
QHAT is computed using the ORACLS subroutine SAMPL. PHI and GAMA are computed using the ORACLS subroutine EXPINT. AMHAT and RHAT are computed using the subroutine ALPHA. If ICH(2,7) is TRUE, a matrix similar to PHI is computed using the matrix PHICL and the ORACLS subroutine EXPSER. The continuous time model for "F" used in computing PHICL is determined in the subroutine PIFEIG as the matrix FM. The last part of QRMHAT is specially adapted to PIF controller computation. For PIF, F and G in the calling program have the following internal form

\[
F = \begin{bmatrix} A & B & 0 \\ 0 & 0 & 0 \\ H & D & 0 \end{bmatrix}
\]

(10)

\[
G = \begin{bmatrix} 0 \\ I \\ 0 \end{bmatrix}
\]

(11)

After PHI and GAMA are computed using EXPINT, they are altered from

\[
PHI = \begin{bmatrix} \text{PHIA} & \text{GAMAB} & 0 \\ 0 & I & 0 \\ X_1 & X_2 & I \end{bmatrix}
\]

(12)

\[
GAMA = \begin{bmatrix} X_3 \\ \Delta t I \\ X_4 \end{bmatrix}
\]

(13)

to

\[
PHI = \begin{bmatrix} \text{PHIA} & \text{GAMAB} & 0 \\ 0 & I & 0 \\ \Delta t H & \Delta t D & I \end{bmatrix}
\]

(14)

\[
GAMA = \begin{bmatrix} 0 \\ \Delta t I \\ 0 \end{bmatrix}
\]

(15)
2. USAGE

a. Calling Sequence

CALL QRMHAT (H, NH, QHAT, NQH, D, ND, RHAT, NRH, AMHAT, NAM, PHI, NPH,
GAMA, NGA, DLT, ICH, DEBUG, W9, NW9DIM, ZERO, NZERO, PHICL, NPHC)

b. Input Arguments

H, QHAT,
D, RHAT,
PHI, GAMA,

PHICL   Matrices packed by columns in one-dimensional arrays. H and D
        are not destroyed upon return. The rest are destroyed upon
        return. Using the notation in the PURPOSE section, QHAT
        contains Q, RHAT contains R, PHI contains F, GAMA contains
        G, and PHICL contains the desired continuous-time closed-
        loop system (FM) constructed in PIFEIG.

W9      Work matrix packed by columns into a one-dimensional array.
Dimension must be at least 14* (NPH(1) * NPH(2) + 1)

ZERO    Work matrix packed by columns into a one-dimensional array.
All elements must be zero. Dimension must be at least
        NPH(1) * NPH(2) + 1

ICH     Two-dimensional logical matrix dimensioned 10 x 15. Sub-
        routines in PIFCGT are called depending on the TRUE or FALSE
        status of elements in ICH.

NH, NQH,
ND, NRH,
NAM, NPH,
NGA, NZERO,

NPHC    Two-dimensional integer vector holding the number of rows and
columns of the matrix shown after the letter N. NZERO is the only vector destroyed upon return.

**DEBUG** Logical scalar indicating the following:
- **TRUE**: Print out debug information
- **FALSE**: Do not print out debug information

**DELT** Real scalar containing the sampling time.

**NW9DIM** Integer scalar containing the maximum dimension of the W9 matrix.

c. **Output Arguments**

- **QHAT**, **RHAT**,  
- **AMHAT**, **PHI**,  
- **GAMA**, **PHICL**  
  Matrices packed by columns into one dimensional arrays containing the discrete plant representation shown in the **PURPOSE** section.

d. **COMMON Block**

None

e. **Error Messages**

If the W9 matrix is insufficiently large the message "THE W9MATRIX IS TOO SMALL" is printed and the program stops. If ICH(4,2) and ICH(4,3) are both true the message "BOTH ICH(4,2) AND ICH(4,3) ARE TRUE CONFLICT IN CALCULATIONS OF GAMA, PHI, QHAT, RHAT, AND AMHAT" is printed and the program stops. If **DEBUG** is true the print option in **ORACLS** subroutines **SAMPL** and **EXPINT** are activated and the following matrices at the end of subroutine execution are printed: **PHI**, **GAMA**, **QHAT**, **RHAT**, **AMHAT**,  
(AMHA), **PHICL** (PHCQ).
f. Subroutines Employed by QRMHAT

ORACL - EQUATE, SAMPL, EXPINT, MULT, EXPSER, TRANP, SCALE, ADD, UNITY,
      JUXTC, JUXTR

PIFLIB - PRNTITL, PRNT, CORNER, ALPHA, BETA

g. Subroutines Employing QRMHAT

PIFCGT

h. Concluding Remarks

ORACL software has a problem when computing the sampled data regulator
representation when NGA(2) = 1. QRMHAT serves as backup software to
QRMP1 when the latter fails.
1. PURPOSE

The purpose of QRMP1 is to compute the sampled-data regulator corresponding to a linear time-invariant continuous system and its associated linear quadratic cost function.

\[ \dot{x} = Fx + Gu \]  
\[ J = \int_0^\infty x^T Qx + u^T Ru \, dt \]

The control, \( u \), is assumed to be constant over the sampling interval, \( \Delta t \). The discrete plant and cost function are given by

\[ x_{k+1} = \Phi x_k + \Gamma u_k \]
\[ J = \sum_{k=0}^{\infty} \left( x_k^T \tilde{Q} x_k + 2x_k^T \tilde{A} u_k + u_k^T \tilde{R} u_k \right) \]

The matrices in the sampled-data regulator problem are computed using the following equations

\[ \Phi = \sum_{k=0}^{\infty} \frac{(F\Delta t)^k}{k!} = e^{F\Delta t} \]  
\[ \Gamma = \int_0^{\Delta t} e^{F\tau} \, d\tau = \int_0^{\Delta t} e^{F\tau} \, d\tau \]  
\[ \tilde{Q} = \int_0^{\Delta t} e^{FT\tau} Q e^{FT\tau} \, d\tau \]  
\[ \tilde{A} = \int_0^{\Delta t} e^{FT\tau} \left[ \int_0^{\tau} e^{F\sigma} \, d\sigma \right] Q \left[ \int_0^{\tau} e^{F\sigma} \, d\sigma \right] e^{FT\tau} \, d\tau \]  
\[ \tilde{R} = \Delta t \, \tilde{R} + G^T \int_0^{\Delta t} \left[ \int_0^{\tau} e^{A\tau} \, d\tau \right] Q \left[ \int_0^{\tau} e^{A\sigma} \, d\sigma \right] e^{A\tau} \, d\tau \]
QHAT, RHAT AND AMHAT are computed using the ORACLS subroutine SAMPL. PHI and GAMA are computed using the ORACLS subroutine EXPINT. If ICH(2,7) is TRUE a matrix similar to PHI is computed using the matrix PHICL and the ORACLS subroutine EXPSER. The continuous time model for "F" used in computing PHICL is determined in the subroutine PIFFEIG as the matrix FM. The last part of QRMP1 is specially adapted to PIF controller computation. For PIF, F and G in the calling program have the following form:

\[
F = \begin{bmatrix}
A & B & 0 \\
0 & 0 & 0 \\
H & D & 0
\end{bmatrix}
\]  \hspace{1cm} (10)

\[
G = \begin{bmatrix}
0 \\
I \\
0
\end{bmatrix}
\]  \hspace{1cm} (11)

After PHI and GAMA are computed they are altered from

\[
PHI = \begin{bmatrix}
PHIA & GAMAB & 0 \\
0 & I & 0 \\
x_1 & x_2 & I
\end{bmatrix}
\]  \hspace{1cm} (12)

\[
GAMA = \begin{bmatrix}
x_3 \\
\Delta t I \\
x_4
\end{bmatrix}
\]  \hspace{1cm} (13)

to

\[
PHI = \begin{bmatrix}
PHIA & GAMAB & 0 \\
0 & I & 0 \\
\Delta t H & \Delta t D & I
\end{bmatrix}
\]  \hspace{1cm} (14)

\[
GAMA = \begin{bmatrix}
0 \\
\Delta t I \\
0
\end{bmatrix}
\]  \hspace{1cm} (15)
The alteration implies the particular digital implementation used in the PIF control law as described in Ref. 1, Chapter III.

2. USAGE

a. Calling Sequence

CALL QRMP1 (H, NH, QHAT, NQH, D, ND, RHAT, NRH, AMHAT, NAM, PHI, NPH, GAMA, NGA, DELT, ICH, DEBUG, W9, NW9DIM, ZERO, NZERO, PHICL, NPHC)

b. Input Arguments

H, QHAT,
D, RHAT,
PHI, GAMA,

PHICL Matrices packed by columns in one-dimensional arrays. H and D are not destroyed upon return. The rest are destroyed upon return. Using the notation in the PURPOSE section, QHAT contains Q, RHAT contains R, PHI contains F, GAMA contains G and PHICL contains the desired continuous-time closed-loop system (FM) constructed in PIFEIG.

W9 Work matrix packed by columns into a one-dimensional array. Dimension must be at least 14 * (NPH(1) * NPH(2) + 1)

ZERO Work matrix packed by columns into a one-dimensional array. All elements must be zero. Dimension must be at least NPH(1) * NPH(2) + 1

ICH Two-dimensional logical matrix dimensioned 10 x 15. Subroutines in PIFCGT are called depending on the TRUE or FALSE status of elements in ICH.
NH, NQH, ND,
NRH, NAM,
NPH, NGA,
NZERO, NPHC, Two-dimensional integer vector holding the number of rows and columns of the matrix shown after the letter N. NZERO is the only vector destroyed upon return.
DEBUG Logical scalar indicating the following:
TRUE: Print out debug information
FALSE: Do not print out debug information
DELT Real scalar containing the sampling time.
NW9DIM Integer scalar containing the maximum dimension of the W9 matrix.
c. Output Arguments
QHAT, RHAT,
AMHAT, PHI,
GAMA, PHICL Matrices packed by columns into one dimensional arrays containing the discrete plant representation shown in the PURPOSE section.
d. COMMON Blocks
None
e. Error Messages
If the W9 matrix is insufficiently large the message "THE W9 MATRIX IS TOO SMALL" is printed and the program stops. If DEBUG is true the following matrices at the end of subroutine execution are printed: PHI, GAMA, QHAT, RHAT, AMHAT (AMHA), PHICL (PHCQ)
f. Subroutines Employed by QRMP1

ORACLS - EQUATE, SAMPL, EXPINT, MULT, EXPSER, TRANP, SCALE, ADD,
UNITY, JUXTC, JUXTR

PIFLIB - PRNTITL, PRNT, CORNER

g. Subroutines Employing QRMP1

PIFLIB - PIFCGT

h. Concluding Remarks

ORACLS subroutine SAMPL has a problem when computing the sampled data regulator representation when NGA(2) = 1. QRMHAT serves as back up software to QRMP1 when the later fails.
II. REDUCE MATRICES (REMAT)

1. PURPOSE

The purpose of REMAT is to eliminate rows and columns of matrices, and reduce the matrix dimensions. The information used by REMAT to eliminate rows and columns is given in the logical vectors XR, UR, YR, XMR, and UMR. The values in the logical vectors are specified in the subroutine HDCON. If a logical variable entry is TRUE, the corresponding state or control or command is to be retained in the PIF design problem. If a logical variable entry is FALSE, the corresponding state or control or command is to be eliminated. The correspondence between logical vector elements and variables is as follows:

\[
\begin{align*}
\Delta u & \quad \Delta w & \Delta q & \Delta \theta & \Delta z & \Delta x & \Delta v \\
XR(1) & XR(2) & XR(3) & XR(4) & XR(5) & XR(6) & XR(7) \\
\Delta r & \Delta p & \Delta \phi & \Delta \psi & \Delta y \\
XR(8) & XR(9) & XR(10) & XR(11) & XR(12) \\
\Delta \delta_1 & \Delta \delta_2 & \Delta \delta_3 & \Delta \delta_4 \\
UR(1) & UR(2) & UR(3) & UR(4) \\
YLN1 & YLN2 & YLAT1 & YLAT2 \\
YR(1) & YR(2) & YR(3) & YR(4) \\
XMR(1) & XMR(2) & XMR(3) & \ldots & XMR(NM) \\
UM(1) & UM(2) & UM(3) & \ldots & UM(NM)
\end{align*}
\]
XR is used to eliminate rows in the following matrices: F, G, Q, AM, PHI, QHAT, GAMA, XL, HX, HAB

XR is used to eliminate columns in the following matrices: F, Q, H, PHI, QHAT, HX, DX

UR is used to eliminate rows in the following matrices: UL, R, RD, RHAT, PHI, GAMA, QHAT

UR is used to eliminate columns in the following matrices: G, D, R, AM, RD, RHAT, GAMA, PHI, QHAT

UR is used to eliminate rows in the following matrices: YL, H, D, QZ, HM, DM, PHI, GAMA, QHAT

YR is used to eliminate columns in the following matrices: QZ, PHI, QHAT

XMR is used to eliminate rows in the following matrices: FM, GM

XMR is used to eliminate columns in the following matrices: FM, HM

UMR is used to eliminate columns in the following matrices: GM, DM

2. USAGE
   a. Calling Sequence

CALL REMAT (N, M, L, NM, MM, DEBUG)

   b. Input Arguments

   N, M, L, NM, MM

   Real scalars indicating matrix dimensions before reduction; changed upon return

   DEBUG

   Logical scalar indicating the following:

   TRUE: Print out debug information

   FALSE: Do not print out debug information

   c. Output Arguments
N, M, L,
NM, MM Real scalars indicating matrix dimensions after reduction.
d. COMMON Blocks
DISCSV, WORKS9, WCHTS, MODF, DISCM, PLA, DIMN1, DIMN2, DIMN3, DIMN4,
EIGEN, LABEL
e. Error Messages and Printout
If DEBUG is TRUE then following sequence of matrices are printed out
after they have been reduced: F, HX, HAB, Q, G, DX, R, AM, RD, H,
D, QZ, FM, HM, CM, DM, PHI, QHAT, GAMA, RHAT
f. Subroutines Employed by REMAT
ORACL5 - EQUATE
PIFLIB - PRNT, PRNTITL, MOVEALL
g. Subroutines Employing REMAT
PIFCGT
h. Concluding Remarks
None
JJ.  STEP RESPONSE EIGENVALUE INFORMATION (RESPON)

1. PURPOSE

RESPON in association with FREEF and FREEO computes step response information for a given eigenvalue.

2. USAGE

a. Calling Sequence

CALL RESPON (RR, RI, FREQNA, DAMRAT, THALF, PERIOD, TAU, T90, T98, T100, OVERSH)

b. Input Arguments

RR       Real part of an eigenvalue
RI       Imaginary part of an eigenvalue

c. Output Arguments

FREQNA   Natural frequency
DAMRAT   Damping ratio
THALF    Time to reach 50% of a step command
PERIOD   Time for a complex eigenvalue to complete one oscillation
TAU      Time constant
T90      Time to reach 90% of a step command
T98      Time to reach 98% of a step command
T100     Time for the step response of a complex eigenvalue to cross the step command the first time.
OVERSH   The overshoot of a stable complex eigenvalue in response to a step input.

d. COMMON Blocks

None

134
e. Error Messages
None

f. Subroutines Employed by RESPON
None

g. Subroutines Employing RESPON
PIFLIB - FREEF, FREEO

h. Concluding Remarks
None
1. **PURPOSE**

Subroutine RUNINFO prints out the values of all variables used in the PIF design. RUNINFO contains a listing of all available subroutine calls and a listing of all variables used by the PIF program along with their definitions.

2. **USAGE**

a. **Calling Sequence**

CALL RUNINFO (NUM)

b. **Input Arguments**

NUM: Directs RUNINFO to print out one of two kinds of information
- =1 Print out subroutine call information
- =2 Print out variable input information

c. **Output Arguments**

None

d. **COMMON Blocks**

WORKS9, RNINFO, RNINFO1, RNINFO2, QWHT, EIGEN, LABLE, UDIR, GLAD, DEBG, AERO, EIGS, HDXA

e. **Error Messages**

None

f. **Subroutines Employed by RUNINFO**

None

g. **Subroutines Employing RUNINFO**

PIFLIB - PIFCGT

h. **Concluding Remarks**

None
1. **PURPOSE**

Three row vectors A, B, and C are scaled by SC1, SC2, and SC3, respectively. The three row vectors are grouped into the matrix D,

\[ D = \begin{bmatrix} A \\ B \\ C \end{bmatrix} \]  

(1)

The matrix D is scaled by the scalar S.

2. **USAGE**

a. **Calling Sequence**

CALL R3MAT (A, B, C, D, ND, NW1, SC1, SC2, SC3, S, W9)

b. **Input Arguments**

A, B, C  One dimensional vectors

NW1    Two dimensional integer vector containing the number of rows and columns of A, B, and C:

NW1(1) = 1
NW1(2) = N

SC1, SC2, SC3  Scale factors for A, B, and C, respectively

S  Scale factor for the matrix D after creation

W9  Work space
c. **Output Arguments**

D  A 3 x N matrix packed by columns into a one-dimensional array with rows of A, B, and C, respectively.

ND  Number of rows and columns of D

ND(1) = Number of rows of D

ND(2) = Number of columns of D
d. COMMON Blocks

None

e. Error Messages

None

f. Subroutines Employed by R3MAT

ORACLS - JUXTR, SCALE

g. Subroutines Employing R3MAT

PIFLIB - FGAERO

h. Concluding Remarks

None
1. PURPOSE

Subroutine SHRSOL solves the algebraic matrix equation

\[ A \times X \times B - X = C \]  

where it is assumed that the matrix A is in lower real SCHUR form and the matrix B is in upper real SCHUR form. Subroutine SYSSLV is used in SHRSOL to solve a system of order 2 or 4 by Crout elimination. Common block SLVBLK is used to communicate with SYSSLV.

2. USAGE

a. Calling Sequence

CALL SHRSOL (A,B,C,N,M,NA,NB,NC)

b. Input Arguments

A,B,C Matrices stored in two-dimensional array. First dimensions are NA, NB and NC, respectively. Second dimensions are at least N, M, and N. Matrix C is destroyed upon return.

N Order of the matrix A.

M Order of the matrix B.

NA,NB, NC Maximum first dimensions of the matrices A, B, and C, respectively, as given in the DIMENSION statement of the calling program.

c. Output Arguments
C  Two-dimensional array containing the solution, X.

d. COMMON Blocks

SLVBLK

e. Error Messages

If the subroutine fails to find a solution the message "ALGORITHM FAILED TO PRODUCE SOLUTION" is printed and the program terminates execution.

f. Subroutines Employed by SHRSOL

ORACLS - SYSSLV

g. PIFLIB subroutines Employing SHRSOL

AXBMXC

h. Concluding Remarks

It is recommended that the program SOLVER which calls AXBMXC which subsequently calls SHRSOL be used to solve Eq. 1. This subroutine is a generalization of algorithms reported in Ref. 3.
NN. ROUND OFF SMALL NUMBERS (SMALL)

1. PURPOSE

To zero out elements in a matrix with very small values.

2. USAGE

a. Calling Sequence

CALL SMALL (A,NA)

b. Input Arguments

A    Input matrix packed by columns in a one dimensional array.
NA   Two-dimensional vector contains number of rows and columns in A
     NA(1) = number of rows in A
     NA(2) = number of columns in A

c. Output Arguments

A    If |A(i,j)| < 1 \times 10^{-8} then A(i,j) = 0. No action otherwise.
NA   No change

1. Error Messages

None

f. Subroutines employed by SMALL

None

g. Subroutines employing SMALL

PIFLIB - PIFEIG

h. Concluding Remarks

Helps in interpreting output if very small elements are zero.
00. **ITERATIVE REFINEMENT SOLUTION OF $A * X * B - X = C$ (SOLVER)**

1. **PURPOSE**

Subroutine SOLVER solves the algebraic matrix equation

$$A * X * B - X = C$$  \hspace{1cm} (1)

for the matrix $X$ using the program AXBMXC. SOLVER is primarily used to determine a highly accurate solution to Eq. 1 using iterative refinement. After AXBMXC computes the first solution to (1), the following matrices are computed using double precision,

$$\text{ERROR} = C + X - A * X * B$$  \hspace{1cm} (2)

$$A * X_1 * B - X_1 = \text{ERROR}$$  \hspace{1cm} (3)

If the Euclidean norm of $X_1$ is less than some tolerance (TOL) the iterative refinement procedure is terminated and $X$ is the solution. If the norm is not less then some tolerance, $X$ is updated,

$$X = X + X_1$$  \hspace{1cm} (4)

substituted back into Eq. 2 and the iterative refinement procedure is performed again. If two successive values of the Euclidean norm of $X_1$ show the norms are increasing, the iterative refinement procedure is terminated.
2. USAGE

a. Calling Sequence

CALL SOLVER (A,ADUM,U,N,NA,B,DBUM,V,M,NB,C,CDUM,NC,X,WORK1,EPSA,
EPSB,XDUM,TOL,ITER)

b. Input arguments

A,B,C Matrices packed by columns in one-dimensional arrays. Not
destroyed upon return. Dimensions are A - N*N, B - M*M,
C - N*M.

ADUM,
BDUM,
CDUM,
WORK1,
XDUM,
U,V Working space vectors of dimensions at least, ADUM - NA*NA,
BDUM - NB*NB, CDUM - NC*NC, XDUM - N*M, WORK1 - 2*(N*M),
U - N*N, V - M*M

ITER Integer scalar indicating the maximum number of iterative
refinements attempted by the subroutine. If ITER .LT. 0,
no refinement iterations are performed.

TOL Real scalar indicating the tolerance value used to determine
if the norm of the equation error matrix is sufficiently small.

N,M,NA,
NB,NC Integer scalars indicating matrix dimensions. It is required
that NA .GT. N+1, NB .GT. M+1, and NC .GT. (MAX(N,M)+1).

EPSA,
EPSB See AXBMXC subroutine description.
c. Output Arguments

X
Upon normal return, X contains the solution to Eq. 1 packed by columns in a one-dimensional array.

d. COMMON Blocks

None

e. Error Messages

If the number of iterative refinements exceeds ITER or the iterative refinement proceed is unstable, the message "SYSTEM FAILED TO CONVERGE AFTER _________ ITERATIONS. RESIDUAL NORM = _________" is printed.

f. Subroutines Employed by SOLVER

ORACLS - ADD, NORMS, BCKMLT, HSHLDR, SCHUR, SYSSLV

"PIFLIB - AXBMXC, SHRSOL

g. Subroutines Employing SOLVER

PIFLIB - CGTPIF

h. Concluding Remarks

This subroutine is based on the algorithm reported in Ref. 3.
IV. SUMMARY - DESIGN PROCEDURE

The design procedure for determining a PIF autopilot control law using the program PIFCGT is outlined below:

Choose the control law sampling interval - DELT

Specify convergence criteria - EPS, ECGT, TOL, ELOG

Specify number of iterative steps - ITER MLOG, IKDGM

Choose autopilot design - ICHOSE

Specify ICH logicals

Specify DEBUG logicals for print out

Specify ICGT for feedforward matrix construction

Determine the new values for the aerodynamic coefficients and reprogram AEROD accordingly

Determine the BETA HOLD feedforward matrices for a number of different forward velocities. Compute a straight line approximation to the $\phi_c$ to $\delta_r$ feedforward matrix element in S22 using forward velocity as the independent variable.

Reprogram HDCON based on the new straight line approximation.

Choose the number of simulation time steps - IMAX

Choose the simulation command magnitude - UMAG

Choose the starting and stopping time for the simulation command - USTRT, USTP

Specify the eigenvalue eigenvector printout - IFREEF, IFREEP

Choose stabilization criteria - DSTAB, DREAL

Choose the design flight condition by specifying forward velocity - VTP

Guess at the quadratic cost weighting matrices
Guess at the command model design parameters - GKDGm, GKPSI, ZDDMAX, ZCLOSE, ZDOMAX, AK1, AK2, ZCLMAX, ZCLMIN, ZCDMAX, GKMn, GKDSLP, GKDICP, GKDMAX

Iterate at guessing command model design parameters until desired command model response is obtained.

Iterate at guessing cost function weighting matrices until a desired tradeoff is made between gain magnitudes, command response and closed-loop eigenvalues.

If the NAMELISTS in Appendix A are used, the only choices that should be required by the designer are for DELT, VTP, and the quadratic weights. AEROD and HDCON need not be modified if the NAVION aircraft is used for control design.
REFERENCES


2. Armstrong, E. S., ORACLS - A Design System for Linear Multivariable Control, Marcel - Dekker, Inc., New York, 1980. ORACLS can be obtained from the NASA sponsored Computer Software Management and Information Center (COSMIC), Suite 12, Barrow Hall, University of Georgia, Athens, GA 30602, by requesting program LAR-12313.


7. CYBER LOADER VERSION 1 REFERENCE MANUAL, Control Data Corporation (CDC), Publication No. 60429800, 215 Moffett Park Drive, Sunnyvale, California 94086.
APPENDIX A

NAMELISTS FOR AUTOPILOT DESIGNS

This appendix presents the nine NAMELISTS used by PIFCGT to design the autopilots in Ref. 1. Initial data values, gain values and quadratic weights are identified in the NAMELISTS.
PIF CONTROL DESIGN FOR BETA HOLD AUTOPILOT

$NAM1$

M = 4, L = 4, NM = 2,

ICH(1, 4) = T, ICH(1, 5) = F, ICH(1, 6) = F, ICH(1, 7) = F,

ICH(1, 8) = F, ICH(1, 9) = F, ICH(1, 10) = T, ICH(1, 11) = T,

ICH(1, 12) = F, ICH(1, 13) = F, ICH(1, 14) = F, ICH(1, 15) = T,

ICH(2, 2) = T, ICH(2, 3) = F, ICH(2, 4) = T, ICH(2, 5) = T,

ICH(2, 6) = T, ICH(2, 7) = T, ICH(2, 8) = T, ICH(2, 9) = T,

ICH(2, 10) = T, ICH(2, 11) = T, ICH(2, 12) = F, ICH(2, 13) = T,

ICH(2, 14) = F, ICH(2, 15) = F, ICH(2, 16) = T,

ICH(4, 2) = T, ICH(4, 3) = F, ICH(4, 4) = T, ICH(4, 5) = T,

ICH(4, 6) = T, ICH(4, 7) = T, ICH(4, 8) = T, ICH(4, 9) = T,

ICH(4, 10) = F, ICH(4, 11) = F, ICH(4, 12) = T, ICH(4, 13) = T,

ICH(4, 14) = T, ICH(4, 15) = T, ICH(4, 16) = F,

ICH(5, 1) = F, ICH(5, 2) = F, ICH(5, 3) = F, ICH(5, 4) = F,

ICH(5, 5) = T, ICH(5, 6) = T,

SS = 1.0, EPS = 1.0E-12, ECGT = 1.0E-12, TOL = 1.0E-12,

ITEP = 12, VTP = 175.0, IC = 5.0, IC4 = 3.0, IG = 4.0,

ELOG = 1.0E-10, MLOG = 7, DREAL = 0.90, IDREG = 5.0,

XM = 12.0, F, UR = 4.0, F, YR = 4.0, F, XMR = 12.0, F,

$NAM2$

IMAX = 192, IRUN = 1, UMAG = 16.0, USTR = 16.0, USTRT = 16.0,

USTP = 16.0, UMA = 16.0, USTR = 16.0, USTRT = 16.0,

$NAM3$

IFREE = 0, FREIG = 0, IFREE = 2, DELE = 0.1,

DEBUM1 = F, DEBUM2 = F, DEBUM3 = F, DEBUM4 = F,

DEBUM5 = F, DEBUM6 = F, DEBUM7 = F, DEBUM8 = F,

DEGU2 = F, DEGU25 = F, DEGU26 = F, DEGU27 = F,

DEGU3 = F, DEGU4 = F, DEGU5 = F, DEGU6 = F,

DEGU4 = F, DEGU46 = F, DEGU48 = F, DEGU5 = F,

DEGU49 = T, DEGU53 = T, DEGU54 = F,

$NAM4$
<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNAM5</td>
<td>F=81<em>0.0, G=36</em>0.0, H=48<em>0.0, D=16</em>0.0,</td>
</tr>
<tr>
<td></td>
<td>FM=4<em>0.0, GM=4</em>0.0, HM=8<em>0.0, DM=8</em>0.0,</td>
</tr>
<tr>
<td>SNAM6</td>
<td>UBQ=0.00E-00, WBQ=0.00E-00, QBQ=0.00E-00, THQ=0.00E-00,</td>
</tr>
<tr>
<td></td>
<td>XIQ=0.0, ZIQ=0.0, VHQ=0.00E-00, RBQ=11.0E-00,</td>
</tr>
<tr>
<td></td>
<td>PBQ=0.00E-00, PHQ=10.0E-00, PSQ=10.0E-00, YIQ=0.00E-00,</td>
</tr>
<tr>
<td></td>
<td>DTH=0.00E-00, DER=0.00E-00, DA=0.10E-00, DRR=0.10E-00,</td>
</tr>
<tr>
<td></td>
<td>DTHD=0.0E-00, DERD=0.00E-00, DARD=7.00E-00, DRRD=7.00E-00,</td>
</tr>
<tr>
<td></td>
<td>ZIQ=0.00E-00, ZZQ=0.00E-00, Z30=3.00E-00, Z4Q=2.50E-00,</td>
</tr>
<tr>
<td></td>
<td>UBDO=0.00E-00, WBDO=0.00E-00, QBDO=0.00E-00, THDO=0.00E-00,</td>
</tr>
<tr>
<td></td>
<td>XIDO=0.00E-00, ZIDO=0.00E-00, VBDO=0.00E-00, RBDQ=0.00E-00,</td>
</tr>
<tr>
<td></td>
<td>PBDO=0.00E-00, PHDO=0.00E-00, PSDQ=0.00E-00, YIDO=0.0,</td>
</tr>
</tbody>
</table>

$\text{AERO1}$

$\text{AERO2}$

$\text{SAER01}$

$\text{SAER02}$

$\text{SNAM7}$

<table>
<thead>
<tr>
<th>Variable</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>EI=9<em>0.0, ER=9</em>0.0, I=9*0.0,</td>
<td></td>
</tr>
<tr>
<td>ERV=400<em>0.0, IEV=9</em>0.0,</td>
<td></td>
</tr>
</tbody>
</table>
PIF CONTROL DESIGN FOR ROLL SEL

SNAM1

N=9,
MM=2,
ICH=12,
ZDMD=20,
AK=9,
ICH=250*T,
ICH(1,4)=T,
ICH(1,8)=F,
ICH(1,12)=F,
ICH(2,2)=T,
ICH(2,7)=T,
ICH(2,11)=T,
ICH(4,2)=T,
ICH(4,6)=T,
ICH(4,10)=F,
ICH(5,1)=F,
ICH(5,5)=T,
ICH(1,11)= T,
ICH(2,16)=F,
ICH(4,3)=F,
ICH(4,7)=T,
ICH(4,11)=F,
ICH(5,2)=F,
ICH(5,6)=T,
ICH(2,15)=F,
ICH(4,4)=T,
ICH(4,8)=T,
ICH(4,15)=T,
ICH(5,3)=T,
ICH(5,4)=T,
ICH(2,14)=F,
ICH(2,19)=F,
ICH(2,13)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
ICH(2,12)=F,
ICH(2,16)=F,
ICH(2,12)=F,
ICH(2,17)=F,
<table>
<thead>
<tr>
<th>SNAM5</th>
<th></th>
<th>SNAM6</th>
<th></th>
<th>SNAM7</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$F=81*0.0$</td>
<td>$G=36*0.0$</td>
<td>$H=48*0.0$</td>
<td>$D=16*0.0$</td>
<td>$E=9*0.0$</td>
<td>$E=9*0.0$</td>
</tr>
<tr>
<td>$FM=4*0.0$</td>
<td>$GM=4*0.0$</td>
<td>$HM=8*0.0$</td>
<td>$DM=8*0.0$</td>
<td>$ERV=400*0.0$</td>
<td>$IEV=9*0.0$</td>
</tr>
<tr>
<td>$UBQ=0.00E-00$</td>
<td>$WBQ=0.00E-00$</td>
<td>$QBQ=0.00E-00$</td>
<td>$THQ=0.00E-00$</td>
<td>$D=16*0.0$</td>
<td>$E=9*0.0$</td>
</tr>
<tr>
<td>$XIQ=0.0$</td>
<td>$ZIQ=0.0$</td>
<td>$VHQ=0.00E-00$</td>
<td>$RBQ=8.00E-00$</td>
<td>$ERV=400*0.0$</td>
<td>$IEV=9*0.0$</td>
</tr>
<tr>
<td>$PBQ=0.00E-00$</td>
<td>$PHQ=6.00E-00$</td>
<td>$PSQ=4.00E-00$</td>
<td>$YIQ=0.00E-00$</td>
<td>$D=16*0.0$</td>
<td>$E=9*0.0$</td>
</tr>
<tr>
<td>$DTH=0.00E-00$</td>
<td>$DER=0.00E-00$</td>
<td>$DA=3.50E-00$</td>
<td>$YIQ=0.00E-00$</td>
<td>$ERV=400*0.0$</td>
<td>$IEV=9*0.0$</td>
</tr>
<tr>
<td>$DTQD=0.00E-00$</td>
<td>$DQRD=0.00E-00$</td>
<td>$DARD=3.50E-00$</td>
<td>$DQRD=4.00E-00$</td>
<td>$D=16*0.0$</td>
<td>$E=9*0.0$</td>
</tr>
<tr>
<td>$ZIQ=0.00E-00$</td>
<td>$ZIQ=0.00E-00$</td>
<td>$Z3Q=3.00E-00$</td>
<td>$ZIQ=2.50E-00$</td>
<td>$ERV=400*0.0$</td>
<td>$IEV=9*0.0$</td>
</tr>
<tr>
<td>$UBDQ=0.00E-00$</td>
<td>$WBDQ=0.00E-00$</td>
<td>$QBDQ=0.00E-00$</td>
<td>$THDQ=0.00E-00$</td>
<td>$D=16*0.0$</td>
<td>$E=9*0.0$</td>
</tr>
<tr>
<td>$XIDQ=0.00E-00$</td>
<td>$ZIDQ=0.00E-00$</td>
<td>$VBDQ=0.00E-00$</td>
<td>$RBDQ=0.00E-00$</td>
<td>$D=16*0.0$</td>
<td>$E=9*0.0$</td>
</tr>
<tr>
<td>$PBQ=0.00E-00$</td>
<td>$PHBQ=0.00E-00$</td>
<td>$PSDQ=0.00E-00$</td>
<td>$PSDQ=0.00E-00$</td>
<td>$D=16*0.0$</td>
<td>$E=9*0.0$</td>
</tr>
</tbody>
</table>
PIF CONTROL DESIGN FOR HDG SEL AUTOPILOT

**SNAM1**

<table>
<thead>
<tr>
<th>N=9,</th>
<th>M=4,</th>
<th>L=4,</th>
<th>NM=2,</th>
</tr>
</thead>
<tbody>
<tr>
<td>MM=2,</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ICHOS=6,</td>
<td>IKDM=1,</td>
<td>GKDGM=0.0020,</td>
<td>GKPSI=10.00,</td>
</tr>
<tr>
<td>ZDMMAX=5.0,</td>
<td>ZCLOSE=9.00,</td>
<td>ZDOMAX=15.0,</td>
<td>AK1=1.0,</td>
</tr>
<tr>
<td>AK2=2.00,</td>
<td>ZCLMAX=0.3,</td>
<td>ZCLMIN=0.1,</td>
<td>ZCDMAX=0.8,</td>
</tr>
<tr>
<td>ZCDMAX = 0.8,</td>
<td>ICH=250* T,</td>
<td>ICH(1,4)= T,</td>
<td>ICH(1,5)= T,</td>
</tr>
<tr>
<td>ICH(1,6)= T,</td>
<td>ICH(1,7)= T,</td>
<td>ICH(1,8)= F,</td>
<td>ICH(1,9)= F,</td>
</tr>
<tr>
<td>ICH(1,10)= T,</td>
<td>ICH(1,11)= T,</td>
<td>ICH(1,12)= F,</td>
<td>ICH(1,13)= F,</td>
</tr>
<tr>
<td>ICH(1,14)= F,</td>
<td>ICH(1,15)= T,</td>
<td>ICH(1,16)= T,</td>
<td>ICH(1,2)= F,</td>
</tr>
<tr>
<td>ICH(1,3)= T,</td>
<td>ICH(2,1)= T,</td>
<td>ICH(2,2)= T,</td>
<td>ICH(2,3)= F,</td>
</tr>
<tr>
<td>ICH(2,4)= T,</td>
<td>ICH(2,5)= T,</td>
<td>ICH(2,6)= T,</td>
<td>ICH(2,7)= T,</td>
</tr>
<tr>
<td>ICH(2,8)= T,</td>
<td>ICH(2,9)= T,</td>
<td>ICH(2,10)= T,</td>
<td>ICH(2,11)= F,</td>
</tr>
<tr>
<td>ICH(2,12)= F,</td>
<td>ICH(2,13)= F,</td>
<td>ICH(2,14)= F,</td>
<td>ICH(2,15)= F,</td>
</tr>
<tr>
<td>ICH(2,16)= F,</td>
<td>ICH(3,1)= F,</td>
<td>ICH(3,2)= T,</td>
<td>ICH(3,3)= T,</td>
</tr>
<tr>
<td>ICH(3,4)= T,</td>
<td>ICH(3,5)= T,</td>
<td>ICH(3,6)= T,</td>
<td>SS=1.0,</td>
</tr>
<tr>
<td>EPS=1.0E-12,</td>
<td>ECGT=1.0E-12,</td>
<td>TOL=1.0E-12,</td>
<td>ITER=12,</td>
</tr>
<tr>
<td>ICFT=5*1,</td>
<td>ICFT(1)=3,</td>
<td>ICFT(4)=0,</td>
<td>ICFT(5)=0,</td>
</tr>
<tr>
<td>ELOG=1.0E-10,</td>
<td>MLOG=7,</td>
<td>DREAL=0.90,</td>
<td>IDREG=5*0,</td>
</tr>
</tbody>
</table>

**SNAM2**

| XR=12* F, | UR=4* F, | YR=4* F, | XMR=12* F, |

**SNAM3**

| IMAX=292, | IUR=1, | UMAG=16*0.0, | USRT=16*0.0, |
| USTP=16*0.0, | UMAG(1,1)=-45, | USTRT(1,1)=0.5, | USTP(1,1)=30.0, |

**SNAM4**

| IFREEF=2, | IFREEG=0, | IFREEP=2, | DELT=0.1, |
| DEBUM1= F, | DEBUM2= F, | DEBUM3= F, | DEBUM4= F, |
| DEBUM6= F, | DEBUM14= F, | DEBUMA= F, | DEBU22= F, |
| DEBU24= F, | DEBU25= F, | DEBU26= F, | DEBU27= F, |
| DEBU31= F, | DEBU42= F, | DEBU43= F, | METERS= T, |
| DEBU44= F, | DEBU46= F, | DEBU48= T, | DEBU416= T, |
| DEBU49= F, | DEBU53= T, | DEBU54= F, |
$\textit{SNAM5}$

\begin{align*}
F &= 81 \times 0.0, \\
G &= 36 \times 0.0, \\
H &= 48 \times 0.0, \\
D &= 16 \times 0.0, \\
FM &= 4 \times 0.0, \\
GM &= 4 \times 0.0, \\
HM &= 8 \times 0.0, \\
DM &= 8 \times 0.0,
\end{align*}

$\textit{SNAM6}$

\begin{align*}
UBQ &= 0.00E-00, \\
WBQ &= 0.00E-00, \\
QBQ &= 0.00E-00, \\
THQ &= 0.00E-00, \\
XIQ &= 0.0, \\
ZIQ &= 0.0, \\
VHQ &= 0.00E-00, \\
RBQ &= 11.00E-00, \\
PQB &= 0.00E-00, \\
PHQ &= 10.00E-00, \\
PSQ &= 10.00E-00, \\
YIQ &= 0.00E-00, \\
DTH &= 0.00E-00, \\
DER &= 0.00E-00, \\
DA &= 0.10E-00, \\
DRR &= 0.10E-00, \\
DTHD &= 0.00E-00, \\
DERD &= 0.00E-00, \\
DARD &= 7.00E-00, \\
DRRD &= 7.00E-00, \\
Z1Q &= 0.00E-00, \\
Z2Q &= 0.00E-00, \\
Z3Q &= 3.00E-00, \\
Z4Q &= 2.50E-00, \\
UBDQ &= 0.00E-00, \\
WBDQ &= 0.00E-00, \\
QBDQ &= 0.00E-00, \\
THDQ &= 0.00E-00, \\
XIDQ &= 0.00E-00, \\
ZIDQ &= 0.00E-00, \\
VBDQ &= 0.00E-00, \\
RBDQ &= 0.00E-00, \\
PBDQ &= 0.00E-00, \\
PHDQ &= 0.00E-00, \\
PSDQ &= 0.00E-00, \\
YIDQ &= 0.00E-00,
\end{align*}

$\textit{SNAM7}$

\begin{align*}
EI &= 9 \times 0.0, \\
ER &= 9 \times 0.0, \\
IE &= 9 \times 0.0, \\
ERV &= 400 \times 0.0, \\
IEV &= 9 \times 0.0,
\end{align*}
PIF CONTROL DESIGN FOR APR LOCI AUTOPILOT

$NAM1$

M=4,

N=9,

L=4,

MM=2,

ICH=9,

IKDGM=10,

ZDDMAX=10,

AK2=1.20,

GKDMIN=-0.11,

ICH=250*T,

ICH(1,4)=T,

ICH(1,8)=F,

ICH(1,12)=F,

ICH(2,2)=T,

ICH(2,7)=T,

ICH(2,11)=T,

ICH(2,15)=F,

ICH(4,2)=T,

ICH(4,6)=F,

ICH(4,10)=F,

ICH(4,14)=F,

ICH(5,1)=F,

ICH(5,5)=T,

ICH(5,9)=T,

ICH(5,13)=T,

ICH(1,5)=T,

ICH(1,9)=F,

ICH(1,13)=F,

ICH(2,3)=F,

ICH(2,8)=T,

ICH(2,12)=F,

ICH(2,16)=F,

ICH(4,3)=F,

ICH(4,7)=T,

ICH(4,11)=F,

ICH(4,15)=T,

ICH(5,2)=F,

ICH(5,6)=T,

ICH(5,10)=F,

ICH(5,14)=F,

ICH(1,6)=F,

ICH(1,10)=T,

ICH(1,14)=F,

ICH(2,4)=T,

ICH(2,9)=T,

ICH(2,13)=F,

ICH(2,17)=T,

ICH(4,4)=T,

ICH(4,8)=T,

ICH(4,12)=T,

ICH(4,16)=T,

ICH(5,3)=T,

ICH(5,7)=T,

ICH(5,11)=F,

ICH(5,15)=F,

ICH(1,7)=F,

ICH(1,11)=T,

ICH(1,15)=T,

ICH(1,19)=T,

ICH(2,5)=T,

ICH(2,19)=T,

ICH(4,5)=T,

ICH(4,9)=T,

ICH(4,13)=T,

ICH(4,17)=T,

ICH(5,4)=T,

ICH(5,8)=T,

ICH(5,12)=T,

ICH(5,16)=T,

ICH(1,16)=T,

ICH(1,20)=T,

ICH(2,6)=F,

ICH(2,10)=T,

ICH(2,14)=F,

ICH(2,18)=F,

ICH(4,6)=T,

ICH(4,10)=T,

ICH(4,14)=T,

ICH(5,2)=F,

ICH(5,6)=T,

ICH(5,10)=F,

ICH(5,14)=F,

ICH(1,8)=T,

ICH(1,12)=T,

ICH(1,16)=T,

ICH(1,20)=T,

ICH(2,4)=T,

ICH(2,8)=T,

ICH(2,12)=F,

ICH(2,16)=F,

ICH(4,4)=T,

ICH(4,8)=T,

ICH(4,12)=T,

ICH(4,16)=T,

ICH(5,3)=T,

ICH(5,7)=T,

ICH(5,11)=F,

ICH(5,15)=F,

ICH(1,9)=F,

ICH(1,13)=F,

ICH(1,17)=F,

ICH(1,21)=F,

ICH(2,5)=T,

ICH(2,9)=T,

ICH(2,13)=F,

ICH(2,17)=T,

ICH(4,5)=T,

ICH(4,9)=T,

ICH(4,13)=T,

ICH(4,17)=T,

ICH(5,4)=T,

ICH(5,8)=T,

ICH(5,12)=T,

ICH(5,16)=T,

ICH(1,10)=T,

ICH(1,14)=F,

ICH(1,18)=F,

ICH(1,22)=F,

ICH(2,6)=F,

ICH(2,10)=T,

ICH(2,14)=F,

ICH(2,18)=F,

ICH(4,6)=T,

ICH(4,10)=T,

ICH(4,14)=T,

ICH(5,2)=F,

ICH(5,6)=T,

ICH(5,10)=F,

ICH(5,14)=F,

ICH(1,7)=F,

ICH(1,11)=T,

ICH(1,15)=T,

ICH(1,19)=T,

ICH(2,5)=T,

ICH(2,19)=T,

ICH(4,5)=T,

ICH(4,9)=T,

ICH(4,13)=T,

ICH(4,17)=T,

ICH(5,4)=T,

ICH(5,8)=T,

ICH(5,12)=T,

ICH(5,16)=T,

ICH(1,8)=T,

ICH(1,12)=T,

ICH(1,16)=T,

ICH(1,20)=T,

ICH(2,4)=T,

ICH(2,8)=T,

ICH(2,12)=F,

ICH(2,16)=F,

ICH(4,4)=T,

ICH(4,8)=T,

ICH(4,12)=T,

ICH(4,16)=T,

ICH(5,3)=T,

ICH(5,7)=T,

ICH(5,11)=F,

ICH(5,15)=F,

ICH(1,9)=F,

ICH(1,13)=F,

ICH(1,17)=F,

ICH(1,21)=F,

ICH(2,5)=T,

ICH(2,9)=T,

ICH(2,13)=F,

ICH(2,17)=T,

ICH(4,5)=T,

ICH(4,9)=T,

ICH(4,13)=T,

ICH(4,17)=T,

ICH(5,4)=T,

ICH(5,8)=T,

ICH(5,12)=T,

ICH(5,16)=T,

ICH(1,10)=T,

ICH(1,14)=F,

ICH(1,18)=F,

ICH(1,22)=F,

ICH(2,6)=F,

ICH(2,10)=T,

ICH(2,14)=F,

ICH(2,18)=F,

ICH(4,6)=T,

ICH(4,10)=T,

ICH(4,14)=T,

ICH(5,2)=F,

ICH(5,6)=T,

ICH(5,10)=F,

ICH(5,14)=F,

ICH(1,7)=F,

ICH(1,11)=T,

ICH(1,15)=T,

ICH(1,19)=T,

ICH(2,5)=T,

ICH(2,19)=T,

ICH(4,5)=T,

ICH(4,9)=T,
$\text{SNAM5}$
\begin{align*}
F &= 81.0 \times 0.0, \\
G &= 36.0 \times 0.0, \\
H &= 48.0 \times 0.0, \\
D &= 16.0 \times 0.0, \\
FM &= 4.0 \times 0.0, \\
GM &= 4.0 \times 0.0, \\
HM &= 8.0 \times 0.0, \\
DM &= 8.0 \times 0.0,
\end{align*}

$\text{SNAM6}$
\begin{align*}
UBQ &= 0.000E-00, \\
WBQ &= 0.000E-00, \\
Q8Q &= 0.000E-00, \\
THQ &= 0.000E-00, \\
XIQ &= 0.0, \\
ZIQ &= 0.0, \\
VHQ &= 0.000E-00, \\
RBQ &= 11.0E-00, \\
P8Q &= 0.000E-00, \\
PHQ &= 10.0E-00, \\
PSQ &= 14.0E-00, \\
YIQ &= 0.000E-00, \\
DTH &= 0.000E-00, \\
DER &= 0.000E-00, \\
DA &= 0.100E-00, \\
DRR &= 0.100E-00, \\
DTHD &= 0.000E-00, \\
DERD &= 0.000E-00, \\
DARD &= 7.000E-00, \\
DRRD &= 7.000E-00, \\
Z1Q &= 0.000E-00, \\
Z2Q &= 0.000E-00, \\
Z3Q &= 2.500E-00, \\
Z4Q &= 0.500E-00, \\
UBDQ &= 0.000E-00, \\
WBDQ &= 0.000E-00, \\
QBDQ &= 0.000E-00, \\
THDQ &= 0.000E-00, \\
XIDQ &= 0.000E-00, \\
ZIDQ &= 0.000E-00, \\
VBDQ &= 0.000E-00, \\
RBDQ &= 0.000E-00, \\
PBDQ &= 0.000E-00, \\
PHDQ &= 0.000E-00, \\
PSDQ &= 0.000E-00, \\
YIDQ &= 0.0,
\end{align*}

$\text{SAFO1}$

$\text{SAFO2}$

$\text{SNAM7}$
\begin{align*}
EI &= 9.0 \times 0.0, \\
ER &= 9.0 \times 0.0, \\
ERV &= 400.0 \times 0.0, \\
IEV &= 9.0, \\
EI &= 9.0 \times 0.0,
\end{align*}
PIF CONTROL DESIGN FOR APR LOC R AUTOPILOT

$SNAM1
N=9, M=4, L=4, NM=2,
MM=2,
ICH=OSE=15, IKDGM=1, GKDGM=0.0020, GKP=1=3.00,
ZDOMAX=10, ZCLOSE=15, ZDOMAX=360, AK1=0.114,
AK2=1.20, ZCLMIN=0.3, ZCLMIN=0.1, ZDOMAX=0.5,
GKDMIN = 0.11, GKDLP=0.0020, GKDLP=0.20, GKDLP=0.22,
ICH=250*T, GKY=0.30,
ICH(1,4)=.T., ICH(1,5)=.T., ICH(1,6)=.F., ICH(1,7)=.F.,
ICH(1,8)=.F., ICH(1,9)=.F., ICH(1,10)=.T., ICH(1,11)=.F.,
ICH(1,12)=.F., ICH(1,13)=.T., ICH(1,14)=.F., ICH(1,16)=.T.,
ICH(2,1)=.T., ICH(2,2)=.T., ICH(2,3)=.F., ICH(2,4)=.T., ICH(2,5)=.T.,
ICH(2,6)=.T., ICH(2,7)=.T., ICH(2,8)=.T., ICH(2,9)=.T., ICH(2,10)=.T.,
ICH(2,12)=.T., ICH(2,13)=.F., ICH(2,14)=.F., ICH(2,15)=.T.,
ICH(2,16)=.F., ICH(2,10)=.T.,
ICH(3,1)=.T., ICH(3,2)=.T., ICH(3,3)=.F., ICH(3,4)=.T., ICH(3,5)=.F.,
ICH(3,6)=.T., ICH(3,7)=.T., ICH(3,8)=.T., ICH(3,9)=.T., ICH(3,10)=.T.,
ICH(4,1)=.F., ICH(4,2)=.T., ICH(4,3)=.F., ICH(4,4)=.T., ICH(4,5)=.T.,
ICH(4,6)=.F., ICH(4,7)=.T., ICH(4,8)=.T., ICH(4,9)=.T.,
ICH(4,10)=.F., ICH(4,11)=.T., ICH(4,12)=.T., ICH(4,13)=.T., ICH(4,14)=.T.,
ICH(5,1)=.F., ICH(5,2)=.F., ICH(5,3)=.T., ICH(5,4)=.T.,
ICH(5,5)=.T., ICH(5,6)=.T.,
SS=1.0, EPS=1.0E-10, EC=1=1.0E-10, TOL=1.0E-10,
ITER=10, VIP=145.0, IC=1=5, IC=1=3, IC=1=4, IC=1=0,
ELOG=1.0E-10, MLOG=7, DREAL=0.90, IDREG=5*0,
$
$SNAM2
XR=12*F, UR=4*F, YR=4*F, XMR=12*F,
$
$SNAM3
IMAX=346, IRUN=1, UMAG=16*0.0, USTR=16*0.0,
USTP=16*0.0, UMAG(1,1)=500.0, USTR(1,1)=0.0, USTR(1,1)=30.0,
UMAG(2,1)=45, USTR(2,1)=0.0, USTR(2,1)=30.0,
$
$SNAM4
IFREEF=0, IFREEF=1, DELT=0.1,
DEBU1=F, DEBU2=F, DEBU3=F, DEBU4=F, DEBU5=F,
DEBU6=F, DEBU7=F, DEBU8=F, DEBU9=F, METERS=F,
DEBU10=F, DEBU11=F, DEBU12=F, DEBU13=F, DEBU14=F,
DEBU15=F, DEBU16=F, DEBU17=F, DEBU18=F, DEBU19=F,
DEBU20=F, DEBU21=F, DEBU22=F, DEBU23=F, DEBU24=F,
DEBU25=F, DEBU26=F, DEBU27=F, DEBU28=F, DEBU29=F,
DEBU30=F, DEBU31=F, DEBU32=F, DEBU33=F, DEBU34=F,
DEBU35=F, DEBU36=F, DEBU37=F, DEBU38=F, DEBU39=F,
DEBU40=F, DEBU41=F, DEBU42=F, DEBU43=F, DEBU44=F,
DEBU45=F, DEBU46=F, DEBU47=F, DEBU48=F, DEBU49=F,
DEBU50=F, DEBU51=F, DEBU52=F, DEBU53=F, DEBU54=F,
$SNAH5$

F=81*0.0, G=36*0.0, H=48*0.0, D=16*0.0,
FM=4*0.0, GM=4*0.0, HM=8*0.0, DM=8*0.0,

$\$SNAH6$

UBQ=0.00E-00, WBQ=0.00E-00, QBQ=0.00E-00, THQ=0.00E-00,
XI=0.0, ZI=0.0, VHQ=0.00E-00, RBQ=11.0E-00,
PBQ=0.00E-00, PHQ=1.00E-00, PSQ=11.0E-00, YIQ=0.00E-00,
DTH=0.00E-00, DER=0.00E-00, DA=0.10E-00, DRR=0.10E-00,
DTHD=0.00E-00, DERD=0.00E-00, DARD=7.00E-00, DRRD=7.00E-00,
Z1Q=0.00E-00, Z2Q=0.00E-00, Z3Q=1.00E-00, Z4Q=2.20E-00,
UBDO=0.00E-00, WBDQ=0.00E-00, QBDQ=0.00E-00, THDQ=0.00E-00,
XIDQ=0.00E-00, ZIDQ=0.00E-00, VBDQ=0.00E-00, RBDQ=0.00E-00,
PBDQ=0.00E-00, PHDQ=0.00E-00, PSDQ=0.00E-00, YIDQ=0.0,

$\$AERO1$

$\$AERO2$

$\$SNAH7$

EI=9*0.0, ER=9*0.0, E1=9*0.0,
ERV=400*0.0, IEV=9*0.0,
PIF CONTROL DESIGN FOR APR LOCP AUTOPILOT

SNAM1

N=9,
M=4,
L=4,
NM=2,
MM=2,
ICHQSE=16,
ZDDMAX=10,
AK2=1.20,
ZCDMAX=.5,
GKDMIN=.11,
ICH=250*.T.,
ICH(1,4)=.T.,
ICH(1,8)=.F.,
ICH(1,12)=.F.,
ICH(2,2)=.T.,
ICH(2,7)=.T.,
ICH(2,12)=.T.,
ICH(2,11)=.T.,
ICH(4,2)=.T.,
ICH(4,6)=.T.,
ICH(4,10)=.F.,
ICH(5,1)=.F.,
ICH(5,5)=.T.,
SS=1.0,
ITER=10,
ICGT=5*1,
ELOG=1.0E-10,
X=12*.F.,
UR=4*.F.,
UR=4*.F.,
XR=12*.F.,
UMR=4*.F.,
UR=4*.F.,
XR=12*.F.,
XMR=12*.F.,

SNAM2

XR=12*.F.,
UR=4*.F.,
XR=12*.F.,
XMR=12*.F.,

SNAM3

IMAX=346,
UMAG=16*0.0,
USTP=16*0.0,
UMAG(1,1)=500,
USTP(1,1)=0.0,
UMAG(2,1)=45,
USTP(2,1)=30.0,

SNAM4

IFREEF=0,
IFREIG=0,
IFREEP=1,
DELT=0.1,
DEB=1*.F.,
DEBM=2*.F.,
DEB=3*.F.,
DEBM=4*.F.,
DEBU=6*.F.,
DEBU=14*.F.,
DEBU=16*.F.,
DEBU=24*.F.,
DEBU=25*.F.,
DEBU=26*.F.,
DEBU=27*.F.,
DEBU=31*.F.,
DEBU=42*.F.,
DEBU=43*.F.,
DEBU=48*.F.,
DEBU=49*.F.,
DEBU=53*.T.,
DEBU=54*.F.,

SNAM5

159
SNAM5
F=81*0.0, G=36*0.0, H=48*0.0, D=16*0.0,
FM=4*0.0, GM=4*0.0, HM=8*0.0, DM=8*0.0,

SNAM6
UBQ=0.000E-00, WBQ=0.000E-00, QBQ=0.000E-00, THQ=0.000E-00,
XIQ=0.00, ZIQ=0.00, VHQ=0.000E-00, RBQ=11.00E-00,
PBO=0.000E-00, PHQ=8.00E-00, PSQ=14.00E-00, YIQ=0.000E-00,
DTH=0.000E-00, DER=0.000E-00, DA=0.10E-00, DRR=0.10E-00,
DTHD=0.000E-00, DERD=0.000E-00, DARD=7.00E-00, DRD=7.00E-00,
ZIQ=0.000E-00, Z2Q=0.000E-00, Z3Q=1.00E-00, Z4Q=2.20E-00,
UBDO=0.000E-00, WBDQ=0.000E-00, QBDQ=0.000E-00, THDQ=0.00E-00,
XIDQ=0.000E-00, ZIDQ=0.000E-00, VBDQ=0.000E-00, RBDQ=0.00E-00,
PBDQ=0.000E-00, PHDQ=0.000E-00, PSDQ=0.00E-00, YIDQ=0.00,

SNAM7
EI=9*0.0, ER=9*0.0, EI=9*0.0,
ERV=400*0.0, IEV=9*0.0,
PIF CONTROL DESIGN FOR PITCH SEL AUTOPILOT

$NAM1$

$N=9,$  $M=4,$  $L=4,$  $NM=2,$

$IM=2,$

$ICH=5,$  $IKDGM=1,$  $GKDGM=0.0020,$  $GKPSI=10.00,$

$ZD=0.75,$  $ZCLOD=19.98,$  $ZDOMAX=8.30,$  $AK1=0.8,$

$AK2=2.00,$  $ZCLOM=2.0,$  $ZCLOM=0.4,$  $ZCDM=0.3,$

$GKDMIN=0.04,$  $GKDOLP=-0.00046,$  $GKDICTP=0.07,$  $GKDICTM=0.07,$

$ICH=250*.*,$

$ICH(1,1)=T,$  $ICH(1,2)=T,$  $ICH(1,3)=T,$  $ICH(1,4)=F,$

$ICH(1,5)=T,$  $ICH(1,6)=T,$  $ICH(1,7)=T,$  $ICH(1,8)=F,$

$ICH(1,9)=F,$  $ICH(1,10)=T,$  $ICH(1,11)=T,$  $ICH(1,12)=F,$

$ICH(1,13)=F,$  $ICH(1,14)=F,$  $ICH(1,15)=T,$  $ICH(1,16)=T,$

$ICH(2,1)=T,$  $ICH(2,2)=T,$  $ICH(2,3)=F,$  $ICH(2,4)=T,$

$ICH(2,5)=T,$  $ICH(2,6)=T,$  $ICH(2,7)=T,$  $ICH(2,8)=T,$

$ICH(2,9)=T,$  $ICH(2,10)=T,$  $ICH(2,11)=T,$  $ICH(2,12)=T,$

$ICH(2,13)=F,$  $ICH(2,14)=F,$  $ICH(2,15)=F,$  $ICH(2,16)=F,$

$ICH(3,1)=T,$  $ICH(3,2)=T,$  $ICH(3,3)=T,$  $ICH(3,4)=T,$

$ICH(3,5)=T,$  $ICH(3,6)=T,$  $ICH(3,7)=T,$  $ICH(3,8)=T,$

$ICH(3,9)=T,$  $ICH(3,10)=T,$  $ICH(3,11)=T,$  $ICH(3,12)=T,$

$SS=1.0,$  $EPS=1.0E-10,$  $TOL=1.0E-10,$  $TOL=1.0E-10,$

$ITER=10,$  $G=150.0,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$

$ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,$  $ICG=10.1,
$\text{SNAM5}$

\begin{align*}
F &= 81 \times 0.0, \\
G &= 36 \times 0.0, \\
H &= 48 \times 0.0, \\
D &= 16 \times 0.0, \\
\end{align*}

$\text{SNAM6}$

\begin{align*}
UBQ &= 0.00 \times 0.0, \\
WBD &= 0.00 \times 0.0, \\
QBD &= 0.00 \times 0.0, \\
THQ &= 8.00 \times 0.0, \\
XIQ &= 0.0, \\
ZIQ &= 0.5, \\
VBQ &= 0.00 \times 0.0, \\
PBQ &= 0.00 \times 0.0, \\
PSQ &= 0.00 \times 0.0, \\
YIQ &= 0.00 \times 0.0, \\
DTH &= 0.00 \times 0.0, \\
DER &= 0.00 \times 0.0, \\
DA &= 0.00 \times 0.0, \\
DRR &= 0.00 \times 0.0, \\
ZIQ &= 3.00 \times 0.0, \\
Z2Q &= 3.00 \times 0.0, \\
Z3Q &= 0.00 \times 0.0, \\
Z4Q &= 0.00 \times 0.0, \\
UBD &= 0.00 \times 0.0, \\
WBD &= 0.00 \times 0.0, \\
QBD &= 0.00 \times 0.0, \\
THD &= 0.00 \times 0.0, \\
XID &= 0.00 \times 0.0, \\
ZID &= 0.00 \times 0.0, \\
VBD &= 0.00 \times 0.0, \\
RBD &= 0.00 \times 0.0, \\
PB &= 0.00 \times 0.0, \\
PHP &= 0.00 \times 0.0, \\
PBD &= 0.00 \times 0.0, \\
YID &= 0.0, \\
\end{align*}

$\text{SNAM7}$

\begin{align*}
EI &= 9 \times 0.0, \\
ER &= 9 \times 0.0, \\
IE &= 9 \times 0.0, \\
ERV &= 400 \times 0.0, \\
IEV &= 9 \times 0.0, \\
\end{align*}
PIF CONTROL DESIGN FOR ALT SEL AUTOPILOT

$NAM1$

N=9, M=4, L=4, NM=2

MM=2,
ICHOOSE=7, IKDGM=1, GKDGM=0.0020, GPKPSI=10.00
ZDOMAX=0.75, ZDCLOSE=19.68, ZDOMAX=8.30, AK1=0.8
AK2=2.00, ZCLMA=2.0, ZCLMIN=0.4, ZCDMAX=0.3
GKDMIN=0.04, GKDLP=-0.0004, GKDIP=0.07, GKDMAX=0.07
ICH=250*T,
ICH(1,4)=T, ICH(1,5)=T, ICH(1,6)=F, ICH(1,7)=F
ICH(1,8)=F, ICH(1,9)=F, ICH(1,10)=T, ICH(1,11)=T
ICH(1,12)=F, ICH(1,13)=F, ICH(1,14)=F, ICH(1,15)=T
ICH(1,16)=T,
ICH(2,1)=T, ICH(2,2)=F, ICH(2,3)=F, ICH(2,4)=T, ICH(2,5)=T
ICH(2,6)=F, ICH(2,7)=T, ICH(2,8)=T, ICH(2,9)=T, ICH(2,10)=T
ICH(2,11)=T, ICH(2,12)=T, ICH(2,13)=F, ICH(2,14)=F, ICH(2,15)=F
ICH(2,16)=F,
ICH(4,2)=F, ICH(4,3)=T, ICH(4,4)=T, ICH(4,5)=T
ICH(4,6)=T, ICH(4,7)=T, ICH(4,8)=T, ICH(4,9)=T
ICH(4,10)=F, ICH(4,11)=F,
ICH(5,1)=F, ICH(5,2)=F, ICH(5,3)=T, ICH(5,4)=T
ICH(5,5)=T,
SS=1.0, EPS=1.0E-10, ECGT=1.0E-10, TOL=1.0E-10
ITER=10, VTP=150.0, MLOG=7, CREAL=0.90, IDREG=5*0

$\$

$NAM2$

XR=12*F, UR=4*F, YR=4*F, XMR=12*F

$\$

$NAM3$

IMAX=298, IRUN=1, UMAG=16*0.0, USTR=16*0.0
USTP=16*0.0, UMAG(1,1)=100, USTR(1,1)=1.0, USTP(1,1)=30.0

$\$

$NAM4$

IFREE=1, IFREIG=0, IFREEP=1, DELT=0.1
DEBUM1=F, DEBUM2=F, DEBUM3=F, DEBUM4=F
DEBUM6=F, DEBU14=F, DEBU15=F, DEBU22=F
DEBU24=F, DEBU25=F, DEBU26=F, DEBU27=F
DEBU31=F, DEBU42=F, DEBU43=F, METERS=F
DEBU44=T, DEBU46=F, DEBU48=F, DEBU49=F
DEBU49=F, DEBU53=T, DEBU54=F

$
PIF CONTROL DESIGN FOR APR GS AUTOPILOT

$\text{SNAM1}$

\begin{align*}
N &= 9, \\
M &= 4, \\
L &= 4, \\
NM &= 2, \\
MM &= 2, \\
\text{ICHOS} &= 14, \\
\text{IKDGM} &= 1, \\
GKDGM &= 0.0020, \\
\text{GKPS} &= 10.00, \\
\text{ZDDMAX} &= 0.03, \\
ZCLOSE &= 0.00, \\
ZDOMAX &= 0.00, \\
\text{AK1} &= 0.04, \\
\text{AK2} &= 0.00, \\
\text{ZCLMIN} &= -5.0, \\
\text{ZCDMAX} &= 3.0, \\
\text{GKDMIN} &= 0.04, \\
\text{GKDLP} &= -0.0004, \\
\text{GKDIP} &= 0.07, \\
\text{GKDMAX} &= 0.07, \\
\text{ICH} &= 250*T, \\
\text{ICH}(1,4) &= T, \\
\text{ICH}(1,5) &= T, \\
\text{ICH}(1,6) &= F, \\
\text{ICH}(1,7) &= F, \\
\text{ICH}(1,8) &= F, \\
\text{ICH}(1,9) &= F, \\
\text{ICH}(1,10) &= T, \\
\text{ICH}(1,11) &= T, \\
\text{ICH}(1,12) &= T, \\
\text{ICH}(1,13) &= F, \\
\text{ICH}(1,14) &= F, \\
\text{ICH}(1,15) &= F, \\
\text{ICH}(1,16) &= F, \\
\text{ICH}(4,2) &= F, \\
\text{ICH}(4,3) &= T, \\
\text{ICH}(4,4) &= T, \\
\text{ICH}(4,5) &= T, \\
\text{ICH}(4,6) &= T, \\
\text{ICH}(4,7) &= T, \\
\text{ICH}(4,8) &= T, \\
\text{ICH}(4,9) &= T, \\
\text{ICH}(4,10) &= F, \\
\text{ICH}(4,11) &= F, \\
\text{ICH}(4,12) &= F, \\
\text{ICH}(4,13) &= F, \\
\text{ICH}(4,14) &= F, \\
\text{ICH}(4,15) &= F, \\
\text{ICH}(4,16) &= F, \\
\text{ICH}(5,1) &= F, \\
\text{ICH}(5,2) &= F, \\
\text{ICH}(5,3) &= T, \\
\text{ICH}(5,4) &= T, \\
\text{ICH}(5,5) &= T, \\
\text{ICH}(5,6) &= T, \\
\text{SS} &= 1.0, \\
\text{EPS} &= 1.0E-10, \\
\text{ECGT} &= 1.0E-10, \\
\text{TOL} &= 1.0E-10, \\
\text{ITER} &= 10, \\
\text{VTP} &= 150.0, \\
\text{ICGT} &= 5*1, \\
\text{MLOG} &= 7, \\
\text{DLOG} &= 1.0E-10, \\
\text{IDREG} &= 5*0, \\
\text{UR} &= 4*F, \\
\text{YR} &= 4*F, \\
\text{XMR} &= 12*F. \\
\end{align*}

$\text{SNAM2}$

\begin{align*}
\text{XR} &= 12*F, \\
\text{UMR} &= 4*F. \\
\end{align*}

$\text{SNAM3}$

\begin{align*}
\text{IMAX} &= 292, \\
\text{IRUN} &= 1, \\
\text{UMAG} &= 16.00, \\
\text{USTRT} &= 16.00, \\
\text{USTP} &= 16.00, \\
\text{UMAG(1,1)} &= -100, \\
\text{USTRT(1,1)} &= 0, \\
\text{USTP(1,1)} &= 30.0, \\
\end{align*}

$\text{SNAM4}$

\begin{align*}
\text{IFREEF} &= 0, \\
\text{IFREIG} &= 0, \\
\text{IFREEP} &= 2, \\
\text{DELT} &= 0.1, \\
\text{DEBUM} &= .F, \\
\text{DEBUM2} &= .F, \\
\text{DEBUM3} &= .F, \\
\text{DEBUM4} &= .F, \\
\text{DEBU} &= .F, \\
\text{DEBU14} &= .F, \\
\text{DEBUA} &= .F, \\
\text{DEBU22} &= .T, \\
\text{DEBU27} &= .F, \\
\text{DEBU31} &= .F, \\
\text{DEBU42} &= .F, \\
\text{DEBU43} &= .F, \\
\text{METERS} &= .T, \\
\text{DEBU44} &= .F, \\
\text{DEBU46} &= .F, \\
\text{DEBU48} &= .F, \\
\text{DEBU16} &= .T, \\
\text{DEBU49} &= .T, \\
\text{DEBU53} &= .T, \\
\text{DEBU54} &= .F. \\
\end{align*}
SNAM5
F = 81 * 0.0, G = 36 * 0.0, H = 48 * 0.0, D = 16 * 0.0,
FM = 4 * 0.0, GM = 4 * 0.0, HM = 8 * 0.0, DM = 8 * 0.0,
$
SNAM6
UBQ = 0.00E-00, WBQ = 0.00E-00, QBQ = 0.00E-00, THQ = 12.5E-00,
XIQ = 0.0, ZIQ = 0.5, VHQ = 0.00E-00, RBQ = 0.00E-00,
PBD = 0.00E-00, PHQ = 0.00E-00, PSQ = 0.00E-00, YIQ = 0.00E-00,
DTH = 0.00E-00, DERR = 0.00E-00, VA = 0.00E-00, DRR = 0.00E-00,
DTHD = 0.00E-00, DERD = 7.00E-00, DARD = 0.00E-00, DRRD = 0.00E-00,
ZIQ = 0.00E-00, Z2Q = 0.25E-00, Z3Q = 0.00E-00, Z4Q = 0.00E-00,
UBDQ = 0.00E-00, HBDQ = 0.00E-00, QBDQ = 0.00E-00, THDQ = 0.00E-00,
XIDQ = 0.00E-00, ZIDQ = 1.00E-00, VBDQ = 0.00E-00, RBDQ = 0.00E-00,
PBDQ = 0.00E-00, PHDQ = 0.00E-00, PSDQ = 0.00E-00, YIDQ = 0.0,
$
$AERO1
LAMDA = 0.0, ZETA = 0.0, LAMDAGS = 3.0,
$
$AERO2
LAMDA1 = 0.0, ZETA1 = 0.0,
$
$SNAM7
EI = 9 * 0.0, ER = 9 * 0.0, EI = 9 * 0.0,
ERV = 400 * 0.0, IEV = 9 * 0,
APPENDIX B

SUBMIT FILE FOR AUTOPILOT DESIGNS

The following is a typical submit file containing the JOB CONTROL CARDS to execute the PIFCGT program on the NASA Langley Computer Complex using the NOS. 1.3 operating system (\ is a blank space).

/JOB

PIFNAM7,T100,CM160000.~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~BIN15DT DBTAYLOR
USER,XXXXXX.

CHARGE,XXXXXX,LRC.

GET,TAPE20=PIFNAM7.

GET,PIFLIBO/UN=1003675C.

GET,ORACLIP/UN=017545N.

MAP,OFF.

LDSET,LIB=PIFLIBO.

LDSET,USEP=AEROD.

LDSET,LIB=ORACLIP.

LDSET,PRESETA=NGINF.

PIFCGT.

BOMB.

EXIT.

ROUTE,OUTPUT,DC=PR,‘,ID,DEF.

SRUCOMP.

REWIND,OUTPUT.

COPY,OUTPUT,TAPE86
REWIND,TAPE86.
REPLACE,TAPE86.
BOMB.
EXIT.
DAYFILE,BADP.
REPLACE,BADP.
EXIT.
/EOR
/PIF/CONTROL/DESIGN/PROGRAM
/EOF

To use the subroutine FREEF, the statements
ATTACH,FTNMLIB/UN=LIBRARY,NA.
LDSET,LIB=FTNMLIB

must be included. To use the subroutine DPIFSP for plotting, the statements
GET,ICSLIB/UN=004675C.
LDSET,LIB=ICSLIB.
REPLACE,TAPE1=PIFPLT7,

must be included in appropriate locations.
APPENDIX C

SEGMENTATION DIRECTIVES

PIFCGT is a large TOP-DOWN program. Central memory would be inefficiently used if the entire program had to be loaded for the program to execute. With segmentation, only certain portions of the executing program need to be in central memory concurrently. Segmented loading is initiated by the execution of a SEGLOAD control statement that causes SEGLOAD directives to be processed as discussed in Ref. 7. The SEGLOAD directives for PIFCGT are (~ is a blank space).

```
#INCLUDE AEROD
#TREE PIFCGT-(DIMSS,HDCON,CFWM,PIFMOIDL,PIFPLC,REMAT,DISCMOD,QRMP1,
ORMHAT,DREG,PIFG,DPIFS,DPIFSP,FGAERO,PIFEIG,RUNINFO,INTERP,PIFFRE)
PIFCGT#GLOBAL#DISCSV,WORKS9,WGHTS,MODF,DISCM,PLA,DIMN1,DIMN2,DIMN3,
DIMN4,DIMN5,SMAT,PFIG,QWHT,EIGEN,LABEL,GLAD,LINES,IDENTS,AERO,FORM
TOL,CONV,SLVBLK,UDIR,DEBG,EIGS,HDXA,AEER1,RNINFO,RNINFO1,RNINFO2,
STP.END,FCL.C.,Q8.IO.,FCL=ENT,FDL.COM
```

#END

Alternate forms of memory allocation may have to be developed by a user depending on machine configuration.
This report documents the computer program PIFCGT. The FORTRAN program, PIFCGT, is a computer design aid for determining Proportional-Integral-Filter (PIF) control laws for aircraft autopilots implemented with a Command Generator Tracker (CGT). The program uses Linear-Quadratic-Regulator synthesis algorithms to determine feedback gains, and includes software to solve the feedforward matrix equation which is useful in determining the command generator tracker feedforward gains. The program accepts aerodynamic stability derivatives and computes the corresponding aerodynamic linear model. The nine autopilot modes that can be designed include four maneuver modes (ROLL SEL, PITCH SEL, HDG SEL, ALT SEL), four final approach models (APR GS, APR LOC1, APR LOC2, APR LOC3), and a BETA HOLD mode. The program has been compiled and executed on a CDC computer.
End of Document