AFFDL-TR-73-142
Volume II

DESIGN METHODS FOR SPECIFYING HANDLING QUALITIES FOR CONTROL CONFIGURED VEHICLES

Volume II Mcpilot Program User's Manual

D.W. VOGT

Approved for public release; distribution unlimited.

FOREWORD

This report was prepared for the United States Air Force by the McDonnell Aircraft Company (MCAIR), a division of the McDonnell Douglas Corporation, P.O. Box 516, St. Louis, Missouri 63166 and covers the work performed under Air Force Contract F33615-73-C-3064, Project 8219, Task 821903. The work was performed from 15 January to 15 Novermber 1973.

The investigation reported here was performed under the sponsorship of the Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio. The Air Force Project Engineers were Mr. David Mayhew (AFFDL/FGL) and Capt. David Ullman.

Mr. Stanley A. LaFavor was the MCAIR program manager for this study. The principal investigator was Robert V. Brulle, from MCAIR Aerodynamics Department, and he was assisted by Daniel C. (Chuck) Anderson from the Guidance and Control Department. The McPilot computer program was written and assembled by Donald W. Vogt and R. Dale Turner. The criticism and suggestions of John Hodgkinson are acknowledged.

This report was submitted by the authors in September 1973 and consists of two volumes. Volume I, titled Technical Discussion, presents the technique and rationale for the development of the McPilot computer program and the pilot rating correlations achieved. Volume II, titled McPilot Program Users Manual, describes the McPilot program, shows how to use it, and presents a complete program listing.

This technical report has been reviewed and is approved.

C. B. WESTBROOK, Chief
Control Criteria Branch
Flight Control Division
AF Flight Dynamics Laboratory

ABSTRACT

A method for predicting aircraft pilot ratings which is applicable to control configured vehicles (CCV's) is developed based on aircraft performance and pilot workload. A discussion of this method of handling quality analysis, and a description of four random tracking pilot rating correlations developed using the method are presented in Volume I of the report. A literature survey classifying the most recent reports applicable to flying qualities is tabulated. A user's manual for McPilot, a general purpose digital computer program developed to implement the method and correlate pilot rating data, is presented in Volume II. The correlations that were developed are used to evaluate the pilot ratings of two conventional contemporary fighter aircraft configurations. The correlations are also applied to two CCV fighter configurations for which limited amounts of handling quality data had been obtained through manned aircraft simulations.

TABLE OF CONTENTS

TABLE OF CONTENTS (CONTINUED)

LIST OF ILLUSTRATIONS

1. INTRODUCTION

The purpose of Volume II of this report is to provide a User's Manual for the McPilot computer program which was developed to automate handling qualities calculations for conventional and CCV (Control Configured Vehicle) aircraft. The theoretical developments and technical discussion covering the material on which McPilot is based is presented in Volume I - Technical Discussion.

McPilot is a very flexible, general purpose computer program composed of a number of subprograms which perform the calculations required to evaluate aircraft handling qualities using the approach presented in Volume I. It is structured so that aircraft and pilot models, expressed in the form of linear differential equations, may be evaluated in terms of system performance functions and pilot workload, and pilot rating predictions can be calculated using pilot rating correlations developed in this study. Alternatively, the program can be used to develop new pilot rating correlations from available data. The program also performs many of the auxiliary computations necessary for engineering analysis of the pilot/aircraft systems under study.

System equation inputs to the program are in matrix form:

$$A \underline{x} = B \underline{u}$$

where $\underline{x}$ is a vector of important pilot/aircraft system states, $\underline{u}$ is an input vector composed of positive or negative powers of the complex Laplace operator (S) and A and B are matrices whose elements are polynomials composed of positive powers of the Laplace variable (S). Program input, output, and computational features and options are fully described in this volume.

For reference purposes, the equations describing the system inputs, cost functions, and pilot rating formulas are tabulated in Appendix A and a complete McPilot program listing is presented in Appendix B. For a complete description of the study and McPilot program see Volume I.

2. PROGRAM INPUT

This section contains information regarding all cards required, and their placement to create a deck ready to be read into a CDC 6600 computer. This assumes that the McPilot program has been loaded onto a device (tape or disk pack) which is accessible by said computer.

2.1 Symbolic Coded Input

Symbolic coded input is used for the bulk of the input data. This includes all data that remains fixed for the entire job, initial starting values for those parameters that may be varied during the optimization process, indicators, title card and classification as required.

2.1.1 Card Format

The program input routine (READA) expects the following format.

Card Columns	1-6	7	8-10	11	12-66	67-72	73-80
Field	I	II	III	IV	V	VI	VII

Card Field I Contains the symbolic name of the variable into which data contained in Field V begins loading.

Example: Card Column 1 12

```
Card Column 1        12
            GAMA     -1.23
            V        500.
```

Card Field II Not used.

Card Field III Contains the words DEC, BCD, TRA, INT, or is blank depending on the type of data to be loaded. The word BCD specifies that N binary coded decimal words of six characters each (N punched in column 12) beginning in column 13 are to be loaded. The word TRA denotes to the input routine that all data has been input and to return control to the calling program. The word DEC and blank are equivalent and specify that data loaded is decimal data.

BCD Example:

```
Card Column 1     8       12
            TITLE BCD     2AN^bbbb ALPHD
```

The 2 in column 12 specifies 2 words where each word is considered to be 6 characters including blanks. The largest number of 6 character words that can be loaded from one card is 9. The user should be very careful to see that the BCD information does not get punched into Field VI. This will cause an input error.

DEC Example:

```
Card Column 1        8       12
            CNSTMX   DEC     1000.,5.,10.
```

If the card is punched as above the numbers will be loaded into the machine as binary floating point numbers. If anything other than BCD, INT, TRA or blank appears in Field II then the word DEC is assumed.

INT Example:

```
Card Column 1        8       12
            MØDNØR   INT     1
            MØDNØR           1
            MØDNØR   INT     1,1,1,1
```

When the word INT is used it is assumed that all numbers on the card will be loaded as integers. If only one integer is punched per card the INT may be punched or omitted.

Card Field IV Not used.

Card Field V The actual input data to the program is punched in the Field V. DEC and INT data must always be left adjusted; that is it must start in column 12 on the input card. All numbers are separated by a "comma" and the field terminates with the first blank. Since Field V ends with the first blank, the user may punch any comments in the remainder of the field.

Card Field VI This field specifies the relocation subscript for the first piece of data in Field V. If this field is blank, an initial subscript of 1 is implied. The subscript may appear anywhere within the field.

Example:

Card Column 1	12	67
AINPUT	4.,0.,20.,10.,18.	1 or blank
AINPUT	20.,17.,30.,15.	6

In the example above the number 4 is loaded into the first cell of the array AINPUT. On the second card 20. is loaded in the sixth cell of the array. The one and six punched in Field VI indicate the subscript for the array AINPUT.

Card Field VII Not used as far as the input routine is concerned. This may be used as a sequence number for the card.

2.1.2 Data Input - Some general notes concerning data input.

1. Nominal value for all input is zero unless noted differently.
2. If the nominal value(s) is satisfactory, no card is required. Do not insert blank cards.
3. If two or more cards are entered with the same variable name and contain the same relocation subscript, only the last will be used.
4. All variable names beginning with the letters I thru N must be whole numbers and contain no decimal unless noted. Likewise all others must contain a decimal unless noted.
5. Cards may be input in any order followed by a 'TRA' card and a END OF FILE control card.

2.1.2.1 Guide - For ease in setting up a case, the data has been grouped according to the McPilot option selected. This grouping is shown in Figure 1, and is detailed in the following sections.

2.1.2.2 GROUP #1 General Input and Indicators

Name	Nominal Value	Description
TITLE	-	May be omitted or if desired insert 'BCD' in columns 8, 9, 10 and a '9' in column 12 followed by up to 54 alphanumeric characters (include blanks)
INDAIR	0	Indicator for aero. characteristics 0 - preset equations used 1 - all matrix data will be input

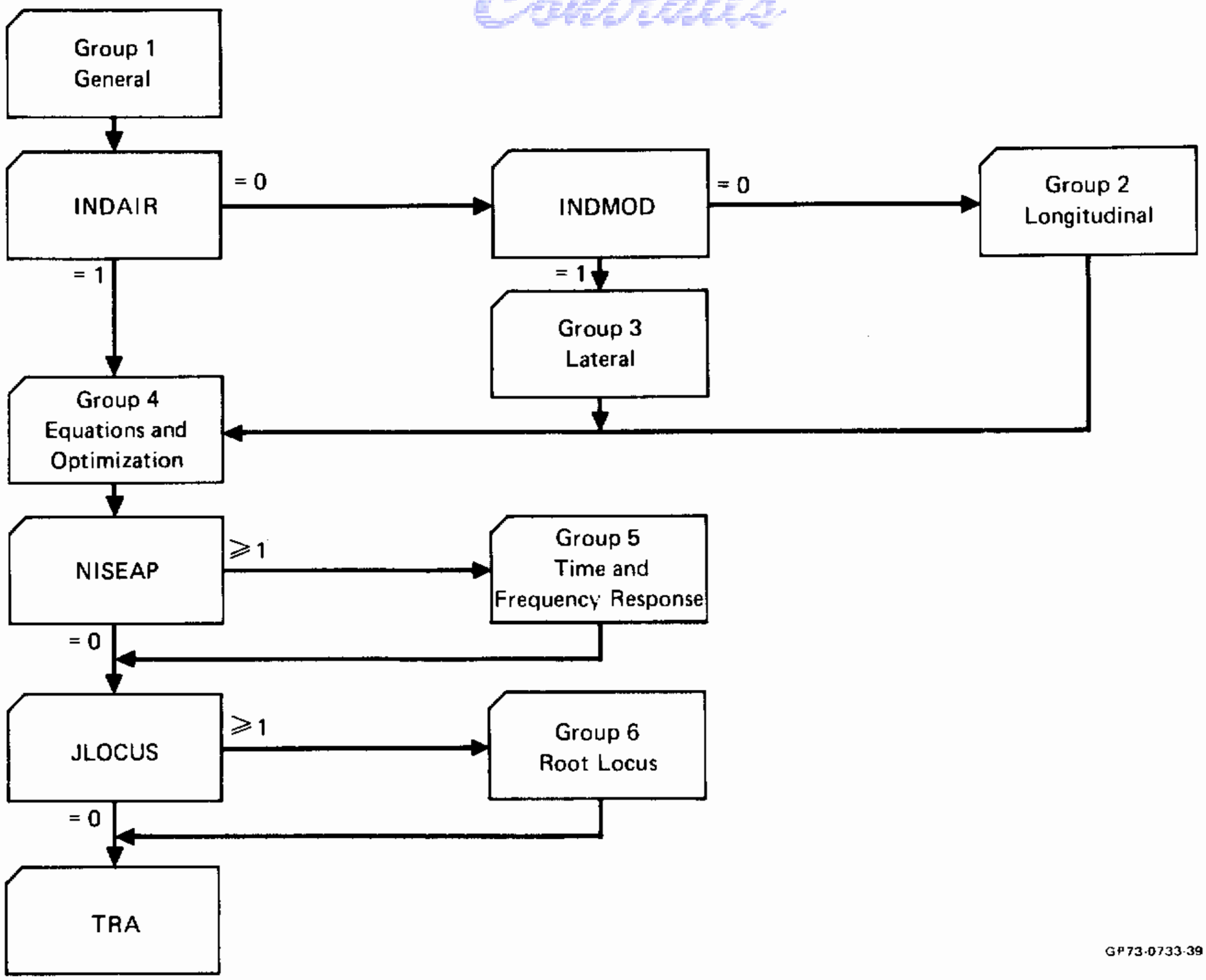


FIGURE 1
SYMBOLIC INPUT DATA GROUPING

Name	Nominal Value	Description
INDMØD	0	Indicator for mode 0 - longitudinal mode 1 - lateral directional mode (This indicator not used if INDAIR = 1)
INDCLS	0	Indicator for classification 0 - unclassified 1 - confidential 2 - secret 3 - top secret 4 - private 5 - proprietary
INDGST	0	Indicator for gusts 0 - no gusts 1 - include gust equations (This indicator not used if INDAIR = 1)
INDTST	0	Indicator for thunderstorm condition 0 - no thunderstorm 1 - thunderstorm condition (This indicator not used if INDGST = 0)
NISEAP	0	Number of ISE terms for which auxiliary program calculations are desired. (Namely, time response and frequency response). 0 - skip
JLØCUS	0	Which pilot function P(N) term will be varied (N=1,2,3,4, or 5) in root locus auxiliary program calculations. 0 - skip root locus
KTIME	0	Print time in seconds. Nominally (KTIME = 0) all intermediate optimization printout will be suppressed - unless elapsed time is within 10 sec. of total time estimated for the job at which time all intermediate results are printed out. This 10 seconds is a built in safety and any value of KTIME is an addition to the 10 seconds already built in.

2.1.2.3 GROUP #2 Input for Aerodynamic Data for Longitudinal Mode

Name	Formulation Symbol	Description
ALT	H	Altitude (ft). Not required except if INDGST = 1 and for printout display

Name	Formulation Symbol	Description
AIY7F	I_y	Aircraft pitch inertia (slug ft^2).
AREFF	S	Reference wing area (ft^2).
CHRDF	$\bar{c}$	Reference chord (ft).
SPANF	b	Span (ft) (For gust calculations).
WTR7P	W	Airplane gross weight (lb).
RHØ	ρ	Flight air density (slug/ft^3).
V	V	Flight velocity (ft/sec).
GAMA	Γ_o	Initial flight path angle (deg).
AA	α_A	Angle of attack of axis system about which aerodynamic data are given (deg).
INDBAF	–	Basic airframe handling qualities. (MIL-F-8785B type). 0 - yes, compute; 1 - no, do not compute.
CXA	C_{x_α}	
CXAD	$C_{x_{\dot{\alpha}}}$	
CXQ	C_{x_q}	
CXU	C_{x_u}	
CXH	C_{x_h}	
CXD	$C_{x_{\delta e}}$	
CZA	C_{z_α}	Aerodynamic characteristic data (non-dimensional/radian).
CZAD	$C_{z_{\dot{\alpha}}}$	
CZQ	C_{z_q}	
CZU	C_{z_u}	
CZH	C_{z_h}	
CZD	$C_{z_{\delta e}}$	
CMA	C_{m_α}	
CMAD	$C_{m_{\dot{\alpha}}}$	
CMQ	C_{m_q}	
CMU	C_{m_u}	
CMH	C_{m_h}	
CMD	$C_{m_{\delta e}}$	

2.1.2.4 GROUP #3 Input for Aerodynamic Data for Lateral Mode

Name	Formulation Symbol	Description
ALT	H	Altitude (ft). Not required except if INDGST = 1 and for printout display.
AREFF	S	Reference wing area (ft^2).
SPANF	b	Span (ft).
WTR7P	W	Airplane gross weight (lb).
RHØ	ρ	Flight Air Density ($slug/ft^3$).
V	V	Flight Velocity (ft/sec).
GAMA	Γ_o	Initial flight path angle (deg).
AA	α_A	Angle of attack of axis system about which aerodynamic data are given (deg).
ALFAI	α_I	Angle of attack of axis system about which inertia data are given (deg).
ZIXB	I_x	Moment of inertia about the x-axis (slug ft^2).
ZIZB	I_z	Moment of inertia about the z-axis (slug ft^2).
ZIXZB	I_{x_z}	Moment of inertia about the x_z-axes (slug ft^2).
CYB	C_{y_β}	
CYBD	$C_{y_{\dot\beta}}$	
CYP	C_{y_p}	
CYR	C_{y_r}	
CYDR	$C_{y_{\delta r}}$	
CYD1	$C_{y_{\delta 1}}$	
CLB	C_{ℓ_β}	Aerodynamic characteristic data (non-dimensional/radian).
CLBD	$C_{\ell_{\dot\beta}}$	
CLP	C_{ℓ_p}	
CLR	C_{ℓ_r}	
CLDR	$C_{\ell_{\delta r}}$	
CLD1	$C_{\ell_{\delta 1}}$	
CNB	C_{n_β}	
CNBD	$C_{n_{\dot\beta}}$	
CNP	C_{n_p}	

Name	Formulation Symbol	Description
CNR	C_{n_r}	Aerodynamics characteristic data (non-dimensional/radian).
CNDR	$C_{n_{\delta r}}$	
CND1	$C_{n_{\delta 1}}$	
CYD2	$C_{y_{\delta 2}}$	
CYD3	$C_{y_{\delta 3}}$	
CLD2	$C_{\ell_{\delta 2}}$	
CLD3	$C_{\ell_{\delta 3}}$	
CND2	$C_{n_{\delta 2}}$	
CND3	$C_{n_{\delta 3}}$	
DA	δ_a	Lateral control deflection, positive when producing right wing down rolling moment, must be nonzero (deg).
D1	δ_1	Control deflection (designated as δ_1, δ_2, or δ_3 for aileron and other integral surfaces in lateral control system).
D2	δ_2	
D3	δ_3	
INDBAF	-	Basic airframe handling qualities (MIL-8785 type). 0 - yes, compute; 1 - no, do not compute.

2.1.2.5 GROUP #4 Equation and Optimization Input

Name	Nominal Value	Description
PLTPI	0.	Initial value of the Ith variable parameter (P's). The search routine must start at a stable point (P's can be stabilized by MINI).
CNSTMN	0.	Minimum permissible values of P's.
CNSTMX	0.	Maximum permissible values of P's.
CUTØFF	0.	The tolerances to be placed on P's to end search. (This card not required if ISETØL = -1).
DEL	0.	Discrete increment of P's to be used in determining the gradient.
ISETRM	0	The number of cost terms to be used in the cost function (maximum of 8).

Name	Nominal Value	Description
KPARAM	0	The number of system parameters (P's) that will be optimized (maximum of 5).
ISETØL	0	To have program stop for tolerance on cost use -1; on P's use 0; on both cost and P's use 1.
TØLISE	0.	The tolerance to be placed on cost to end search. (This card not required if ISETØL = 0).
JUMP	0.	Initial value of step size used in method to find minimum of a gradient line. Use 1. if unknown. (decimal point required).
NIN	0	The element numbers in $\underline{u}$ which defines the system transfer function input for each ISE term.
NMØD	0	The number of the models to be used in each ISE term (maximum of 8).
NØUT	0	The element numbers in $\underline{x}$ which defines the system transfer function output for each ISE term.
NTYPE	0	The orders of S in the input denominators (u_{NIN} = (1/S**NTYPE)); NTYPE can equal 0, $\pm$1, $\pm$2, ...
WEIGHT	0.	The weighting factors for each cost term.
MØDNØR	-1	Model numerator orders of up to 8 models.
MØDDØR	0	Model denominator orders of up to 8 models.
PØLYN	0.	Model numerator polynomial coefficients (Maximum number of coefficients is 40).
PØLYD	1.	Model denominator polynomial coefficients (Maximum number of coefficients is 60).

(See Section 2.1.3 for clarification of how the model data is inserted.)

Name	Nominal Value	Description
ICØST	0	Indicator to denote method to be used to determine cost function. Maximum of 8 terms. 0 - user supplied cost equations; 1 - Mean Square or integral squared error; 2 - root mean square or square root of ISE.
AØRDER	-1	Order of polynomials to be input into the A matrix (no decimal).
BØRDER	-1	Order of polynomials to be input into the B matrix (no decimal).
AINPUT	0.	Coefficients of polynomials to be input into the A matrix.
BINPUT	0.	Coefficients of polynomials to be input into B matrix (maximum number of coefficients is 290).

(See Section 2.1.4 for clarification of the order and matrix polynomials input.)

Name	Nominal Value	Description
MSKIP	0	Nominally the initial control parameters will be stabilized; ≠ 0 - skip stabilization.
NUMEQ	0	Number of equations (Maximum of 12).

2.1.2.6 GROUP #5 Input for Time Response & Frequency Response Auxiliary Computations

Name	Nominal Value	Description
ISEAUX	0	The ISE term(s) (up to 8 max.) for which you desire these auxiliary calculations.
NFREQ	21	The number of omegas to use for frequency response calculations (maximum of 21).
ØMEGAS	→	Calculation frequencies. The 21 nominal values are .01, .05, .1, .15, .2, .3, .5, .7, 1., 1.5, 2., 2.5, 3., 3.5, 4., 4.5, 5., 5.5, 6., 8., 10.
INDTRØ	0	Time response option for input. 0 - unit step input; 1 - unit impulse input.

Name	Nominal Value	Description
MØI	0	Method of integration. 0 - predictor corrector; 1 - Runge-Kutta.
HMAX	.1	Delta time for print interval.
NTIME	2	The number of input time values (maximum of 60).
RTIMES	0., 3.	Input time values.
RINPUT	1.,1.	Corresponding input values at RTIMES listed.

2.1.2.7 GROUP #6 Input for Root Locus Auxiliary Computations

Name	Nominal Value	Description
NCASE	1	The number of root locii to be run (maximum of 4).
KGAIN	2	Gain mode. 2 180° locus; - 2 0° locus.
DELT	.125	Step size between points.
TØL	.0001	Error tolerance.
BNDY	-12.,4., -2., 8.	Computation area boundary for root locus plot. 4 values must be input (if nominal values aren't satisfactory) for each case, and in the following order - left, right, bottom and top respectively.

2.1.3 Model Transfer Function Input Clarification - Entering the desired model(s) numberator and denominator order is done by entering an integer subscript anywhere in the field from card column 67 to 72. For example, for model number 4, use a subscript of 4 on the cards MØDNØR and MØDDØR. Enter the coefficients using PØLYN and PØLYD starting with the constant term. Each succeeding term is the coefficient of the next highest power of S. If more than one model is entered, start with lowest numbered model. For a zero model, no input is required for this section. It should be noted that a subscript value entered on a card starting with PØLYN or PØLYD is needed only if more than one card is required (or desired) to input the coefficients. The following example should explain the above more clearly.

Example: Transfer function desired on model 2.

Model 2 numerator order is 2, denominator order is 4.

Card Column	1	8	12	67
	MØDNØR	INT	2	2
	PØLYN		331., 183., -21.6	
	MØDDØR	INT	4	2
	PØLYD		331., 473., 104.6, 19.72, 1.	

Note: INT was not required above since only 1 integer value was entered per card. See 2.1.1 for more information on use of INT and the subscript field, columns 67-72.

2.1.4 <u>Matrix Equations Input</u> - To input equations into the matrix it is necessary to input the order and the coefficients into their respective arrays correctly. This can be done by entering to integer subscripts anywhere in the field from card column 67 to 72. The matrix subscript guide is shown in Figure 2. This is for a 12 x 12 matrix, and the number in each square is the subscript to be used for the array AØRDER (or BØRDER). Enter the order for all equations to be input in this fashion. Then enter the coefficients using AINPUT (or BINPUT) starting with the constant term. Each succeeding term is the coefficient of the next higher power of S. Start with the polynomial whose order has the lowest subscript value, then proceed to the next polynomial, etc. It should be noted that a subscript value entered on a card starting with AINPUT (or BINPUT) is needed only if more than one card is required (or desired) to input the coefficients. The following example should explain the above more clearly. (Note - when BLANK sub-routine is used to insert matrix array data, the row, column designation is used which is discussed in Section 2.2.2.1.

Example: A second order term having ascending order coefficients of -1.6, -.08, and 0.024 is to be entered into the 5th column, 6th row. Also a first order term with ascending order coefficients of -1.0 and -0.05 is to be entered into the 8th column, 8th row. Both forms shown in the following example are acceptable. The proper subscripts from Figure 2, would be 54 and 92 respectively.

Row \ Col	1	2	3	4	5	6	7	8	9	10	11	12
1	1	13	25	37	49	61	73	85	97	109	121	133
2	2	14	26	38	50	62	74	86	98	110	122	134
3	3	15	27	39	51	63	75	87	99	111	123	135
4	4	16	28	40	52	64	76	88	100	112	124	136
5	5	17	29	41	53	65	77	89	101	113	125	137
6	6	18	30	42	54	66	78	90	102	114	126	138
7	7	19	31	43	55	67	79	91	103	115	127	139
8	8	20	32	44	56	68	80	92	104	116	128	140
9	9	21	33	45	57	69	81	93	105	117	129	141
10	10	22	34	46	58	70	82	94	106	118	130	142
11	11	23	35	47	59	71	83	95	107	119	131	143
12	12	24	36	48	60	72	84	96	108	120	132	144

GP73-0733-58

FIGURE 2
MATRIX ARRAY SUBSCRIPT GUIDE

Card Column	1	8	12	67
	AØRDER	INT	2	54
	AINPUT		-1.6, -.08,.024	
	AORDER	INT	1	92
	AINPUT		-1.,-.05	4
or				
	AØRDER		2	54
	AØRDER		1	92
	AINPUT		-1.6, -.08, .024, -1., -.05	

2.2 Fortran Coded Input

Fortran coded input is also used for data input. It is of the form of a user written subroutine called BLANK, and is divided into three sections called BLANKI, COSTI and PRATEI.

BLANKI is used to insert the equations that contain the optimization parameters which are varied to minimize the cost function (P's) into the A & B matrices. BLANKI can also be used to insert other A & B matrix array terms in place of the symbolic data input cards AINPUT & BINPUT. This feature can be used to great advantage in analyses of the effects of control system gains by generalizing the equations and letting BLANKI perform the computations. In this manner the effects of control variations on pilot ratings can be easily made by changing a card or two in BLANKI.

COSTI allows the user to insert non-ISE cost function terms or equations into the program. COSTI Fortran input is required for programming cost functions that combine ISE terms in a non linear manner and/or use optimization parameters. If a user input cost function is programmed, the corresponding symbolic data input card ICOST must have a zero for that term.

PRATEI allows the user to enter the pilot rating equation. This equation can consist of optimization parameters and cost functions in any programable arithmetical forms.

2.2.1 Card Format (Basic Information)

Card Columns	1	2-5	6	7-72	73-80
Field	I	II	III	IV	V

Card Field I — Comment information is designated by a c, *, or $ in column 1. Comments may appear anywhere in columns 2-80. A comment statement has no effect upon the program but does appear on the listing.

Card Field II — Used for statement label (unsigned integer) which can be referred to from other sections of the program. The integer may appear anywhere in the field and must be a unique statement label.

Card Field III — A Fortran statement may be continued for 19 lines. Continuation of a Fortran statement is denoted by any character in column 6 other than zero. A blank or a zero denotes the first line of a statement.

Card Field IV — Used for the Fortran statement. Any type of Fortran statement is allowed as long as the computation remains within the BLANK subroutine and does not change the optimization parameters.

Card Field V — Positions beyond 72 may be used for identification codes or sequencing but are not processed by the compiler.

2.2.2 BLANK Input - BLANK is a user written subroutine, and consists of three sections; BLANKI, COSTI, and PRATEI.

2.2.2.1 BLANKI - The BLANKI section is provided to input those system A and B matrix elements that are functions of the parameters P(N) which are varied to minimize the cost function. In addition, BLANKI can be used to input other A and B matrix elements if desired. The user coded Fortran statements are inserted after the card 'ENTRY BLANKI'. The variable names available and description follow along with an example.

AØRDER (I,J) is used to input the order, an integer value, for the Ith row and Jth column entry in the A matrix.

BØRDER (I,J) similarily used for B matrix.

A (I,J,K) is the name for the Ith row, Jth column and Kth coeffi- where K = 1 is used for the constant term coefficient, K = 2 for the first order term coefficient, etc. of the entering polynomial.

B (I,J,K) similarily used for B matrix.

P(N) is the array name of the parameters which are varied to minimize the cost function and N = 1, 2, 3, 4 or 5.

KNTISE is a counter which is zero only once -- the first time BLANK is called. It can be used to save computation time on inputs that do not vary (see below).

```
Card Column   2    7
                   ENTRY BLANKI
                   IF(KNTISE.NE.0)GØ TØ 30
                   DO 20 I = 1, 12
                   AØRDER (1,I) = -1
                   AØRDER (I,1) = -1
                   AØRDER (4,I) = -1
              20   AØRDER (I,4) = -1
                   AØRDER (1,1) = 0
                   AØRDER (4,4) = 0
                   A(1,1,1) = 1.
                   A(4,4,1) = 1.
                   AØRDER (6,8) = 2
                   AØRDER (6,6) = 2
                   TAU = .2
              30   A(6,8,1) = -2. *P(1)
                   A(6,8,2) = P(1)*(TAU-2.*P(2))
                   A(6,8,3) = P(1)*TAU*P(2)
                   A(6,6,1) = 2.
                   A(6,6,2) = TAU+2.*P(3)
                   A(6,6,3) = TAU*P(3)
                   RETURN
```

In the above example, the first time BLANK is called, the polynomial orders of rows 1 and 4 and columns 1 and 4 are set to a -1, except for array locations 1,1 & 4,4. The order of these locations are set to zero, and 1.0 inserted in the element. The array locations having an order of -1 are ignored by the computer. This technique is used to eliminate the automatically programmed u & h equations in the longitudinal mode (eliminating the phugoid). The values themselves are still in the matrix locations but are not used. The other Fortran statement then insert the following expressions in array locations 6,8 & 6,6.

$P(1)[\tau P(2)S^2 + (\tau - 2P(2))S-2]$ in array location 6,8

and $[\tau P(3)S^2 + (\tau + 2P(3))S + 2]$ in array location 6,6

2.2.2.2 COSTI - In the BLANK subroutine after card 'ENTRY CØSTI' the user supplied cost function terms are entered. This is required whenever a cost function is not just a weighted linear combination of ISE (or $\sqrt{\text{ISE}}$) terms. If a user supplied cost term is entered, the symbolic input data card ICOST must have a zero entered for the corresponding cost function term (see example below). The variable names and description follow along with the previous example.

ACØST(I) is the variable name that must be set equal to the supplied cost function. The subscript (I) equals 1 to 8 and depends on the cost function term being supplied.

P(N) is the array name of the parameters which are varied to minimize the cost function (N = 1, 2, 3, 4, or 5).

In this example the user is supplying the second cost term of a total cost function defined as:

$J = 75.\ \sigma_\phi + 3.25\ (1 - e^{-0.77T_1})$ where

$\sigma_\phi = \sqrt{\text{ISE}_\phi}$, numerically equal to the r.m.s. roll angle. σ_ϕ is computed directly and no user supplied cost term is required.

The user supplied portion of the second term $(1 - e^{-0.77T_1})$ uses T_1 (the pilot lead) and for this example T_1 is designated as P(2). To input this cost function term requires user supplied Fortran statements as follows:

Card Column 7

```
ENTRY CØSTI
E2 = -P*.77
E1 = EXP(E2)
ACØST(2) = 1. - E1
RETURN
```

The other symbolic data required to complete the cost function input would be:

ICOST 2, 0

The 2 indicates a standard ISE_ϕ (σ_ϕ) to be computed for the first cost term, and the 0 indicates that a user written cost function term has been input for the second term.

WEIGHT 75.,3.25

This defines the coefficient weight on the two cost function terms to provide the complete cost function.

ISETRM 2

This indicates that two cost function terms are being computed.

NIN I, J

NOUT K, L

I and K are the matrix element number in $\underline{u}$ and $\underline{x}$ respectively which define the system transfer function input and output for the first (σ_ϕ) cost term. These data would depend on how the equations are input in the A and B matrices. Usually I = 1 and K = 2 for the ϕ/ϕ_c transfer function. J and L are the input and output element numbers for the second cost term. The transfer function for this second term is not used in the cost term calculation. J and L are dummy numbers which are always required whenever a user cost term is specified. Any dummy system transfer function can be specified. This dummy transfer function is computed and printed but not used in any other computation.

2.2.2.3 PRATEI - In the BLANK subroutine after card ENTRY PRATEI the equation to calculate the pilot rating is entered if desired. The variable names and description follow along with an example.

PR is the variable name that must be set equal to the supplied pilot rating equation.

ACOST(I) - cost function for Ith ISE term.

P(N) - pilot parameter, Nth value.

In this example the pilot rating is:

$PR = 1 + 71\ \sigma_\phi + 3.1\ (1 - e^{-.77T_1})$

where σ_ϕ is ACØST(1) (the first cost term)

and $(1 - e^{-.77T_1})$ is ACOST(2) (see example in previous section 2.2.2.2)

```
Card Column   7
              ENTRY PRATEI
              PR = 1. + 71.*ACØST(1) + 3.1*ACOST(2)
              RETURN
```

2.2.2.4 Subroutine BLANK Required Cards - The listing below shows the required cards of the BLANK subroutine, except for comment cards (C in column 1) which are optional.

```
1     7      card column
      SUBROUTINE BLANK
C
C     THIS SUBROUTINE IS FILLED IN BY USER, WHO MAKES THE A AND/OR B MATRICES
C                    FUNCTIONS OF THE P(*)*S
C     SIMULTANEOUSLY THE USER MAKES THE AORDER AND BORDER MATRICES COMPATIBLE
C                    WITH THE A AND B MATRICES RESPECTIVELY
C    BLANK COMPUTES THOSE ELEMENTS OF THE A MATRIX THAT
C    DEPEND ON THE PILOT PARAMETERS, P(I).  SKBJI COMPUTES
C    THE DETERMINANT OF THE A MATRIX (AS A HUMONGOUS POLYNOMIAL
C    IN S).  POLYRT FINDS ROOTS OF POLYNOMIALS.
      COMMON COMON(4400)
      COMMON / 5 / A(12,12,11),DUM(15),B(12,12,11),DUM1(16)
      INTEGER AORDER(12,12),BORDER(12,12)
      DIMENSION P(5),ACOST(8)
      EQUIVALENCE (AORDER,COMON(   1))
      EQUIVALENCE (BORDER,COMON( 145))
      EQUIVALENCE (KNTISE,COMON( 514))
      EQUIVALENCE (P     ,COMON( 780)
      EQUIVALENCE (PR    ,COMON(4345))
      EQUIVALENCE (ACOST ,COMON( 695))
      ENTRY BLANKI
C     USER SUPPLIED BLANKI INPUT FOLLOWS THIS CARD
      RETURN
      ENTRY COSTI
C     USER SUPPLIED COST FUNCTION EQUATIONS COSTI FOLLOW THIS CARD, IF DESIRED
      RETURN
      ENTRY PRATEI
C     USER SUPPLIED PILOT RATING EQUATION PRATEI FOLLOWS THIS CARD, IF DESIRED
      RETURN
      END
```

2.3 McPilot Deck Structure

Figure 3 shows the make-up of the deck to run a case on the McPilot program. The control cards, which vary from installation to installation are not detailed. The End-Of-Record and End-Of-Information control cards, consist of the digits 7, 8, 9 and 6, 7, 8, 9 respectively multi-punched in card column 1. All cards shown are required to run a case.

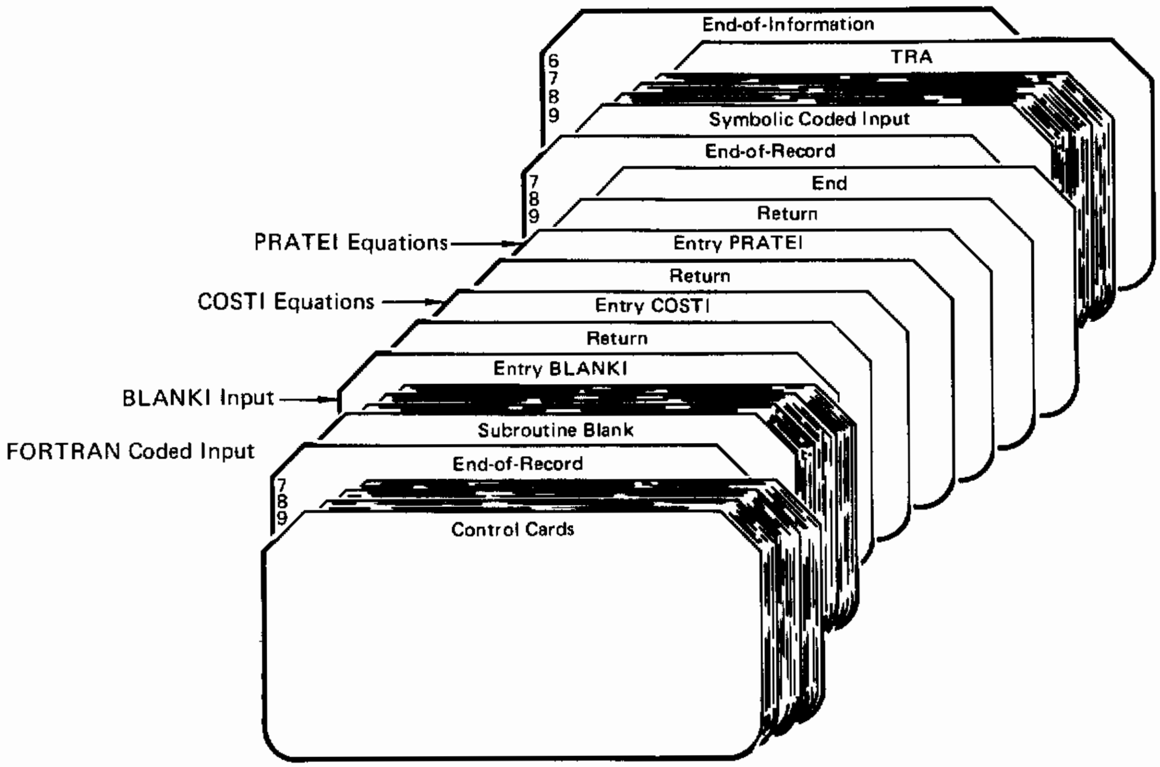


FIGURE 3
MC PILOT DECK STRUCTURE

3. PROGRAM OUTPUT

This section contains information regarding the output generated by the McPilot program. Depending on the control cards used, a Fortran listing of the blank subroutine is usually listed first along with compilation information where applicable. This is followed by output as per example program output in Section 3.2 and detailed in the following sections. Portions of the output are optional per user input indicator cards.

3.1 Output Sequence

Output may be divided into the eleven groups listed below and explained further in the following subsections.

(a) Listing of symbolic input cards.

(b) Basic airframe characteristics data about the longitudinal or lateral axis (INDAIR = 0).

(c) Dryden relationship gust calculations.

(d) Initial check of input data with the A and B matrices evaluated at the initial parameter values.

(e) Transfer function initial stabilization (MSKIP = 0).

(f) Output of the optimization data at the initial parameter values.

(g) Output of the optimization data at the minimum cost function values along each gradient line of the search.

(h) Calculated pilot rating.

(i) Frequency response calculations. } (NISEAP > 0)

(j) Time response calculations. } (NISEAP > 0)

(k) Root locus calculations (JLOCUS > 0).

3.1.1 Symbolic Input Listing - All of the symbolic input inserted by the user is listed right after the compilation print. This makes it convenient for the user to quickly scan the data input for errors and to find data to be changed for additional runs.

3.1.2 Airframe Characteristics - If INDAIR = 0, either the longitudinal (INDMØD = 0) or lateral (INDMØD = 1) aerodynamic characteristics are printed. This consists of:

(a) Aerodynamic characteristic input data.

(b) Dimensional stability derivatives.

If in addition the user wants the airframe characteristics (INDBAF = 0) the following is computed and printed:

(a) Real and imaginary roots.

(b) Damping ratio and frequency computations.

(c) Characteristic equation coefficients.

3.1.3 Gust Calculations - If INDGST = 1, the following results of the Dryden relationship gust calculations are listed:

(a) Scale length L_w, L_u, L_v.

(b) Root mean square of the gust intensity σ_w, σ_u, σ_v.

3.1.4 Initial Optimization Data - This is a printout of input data, but arranged in a more understandable manner. It consists of:

(a) Expanded A and B matrices printed with 4 rows and 12 columns per page.

(b) Expanded model numerator and denominator transfer functions for each model.

(c) Number of ISE terms in the cost function.

(d) Data values for each ISE term.

(e) Number of variable parameters.

(f) Data values for each parameter including their initial input values.

(g) Number of equations.

(h) Method of ending optimization (tolerances on parameters, ISE, or both).

3.1.5 Initial Stabilization - When MSKIP = 0, McPilot program goes to the MINI subroutine. This subroutine checks on the stability of the transfer function with the initial values of the optimization parameters. If the system is not stable, MINI changes the optimization parameters in such a manner as to drive any positive real roots, or positive real parts of a complex root, negative. The printout consists of the values of the most positive root and the optimization parameters for each iteration until stable starting values are obtained, 200 iterations have been completed, or creep step (change in optimization parameters) is less than the original value/100. This last condition implies that the predominate unstable root is unchanged for a local change in the values of the optimization parameters. This is usually indicative of an error in the system set up.

3.1.6 Initial Optimization Output - This is output of the MISER subprogram of McPilot. The quantities printed are:

(a) KOUNT: (i-j) where i is the number of times the gradient of the cost function has been determined, and j is the total number of cost function terms which have been calculated. Initial point is 0-1.

(b) JUMP: The initial value of JUMP.

(c) The value of each cost term and the scale factor used to make model and system steady states equal. Scale factors are internally calculated.

(d) TOTISE: The total cost function value.

(e) AJP: The interval size along the gradient line at the last step in the optimization procedure. Initial value is 0.

(f) P: The initial values of the optimization parameters.

(g) DIRCOS: The direction cosines of the gradient line. Initial values 0.

(h) The system transfer function for each ISE cost term and the roots of each system transfer function.

3.1.7 Optimization Output - The output of this section is identical to the previous section, except that it is the values found at the minimum point along the gradient search. This portion can be suppressed by use of KTIME (see Section 2.1.2.2).

3.1.8 Pilot Rating - The pilot rating, calculated using the equation(s) supplied by the user in blank subroutine (PRATEI), is printed.

3.1.9 Frequency Response - If NISEAP $\geq$ 1, the frequency response is calculated and printed. The output consists of:

(a) ISE term, requested per input card ISEAUX (see 2.1.2.6), and the numerator and denominator order and coefficients.

(b) For each input frequency (rad/sec) the program prints real part, imaginary part, magnitude, magnitude (dB), phase angle.

3.1.10 Time Response Calculations - If NISEAP $\geq$ 1, the time response is calculated using unit step input (INDTRØ = 0), or unit impulse input (INDTRØ = 1), and the following is printed out:

(a) ISE term requested, type of input (unit step or unit impulse), numerator and denominator orders and coefficients.

(b) For each time (sec) the program prints the input variable and output variable.

(c) Printer-plot of input versus time.

(d) Printer-plot of output versus time.

3.1.11 Root Locus Calculations - If JLOCUS $\geq$ 1, the auxiliary root locus calculations are performed and the results printed as follows:

(a) Contents of the A matrix for the open loop poles polynomial-- PLOCUS set to zero. PLOCUS is the root locus parameter.

(b) Contents of the A matrix for the characteristic equation -- PLOCUS set to one.

(c) Open loop pole and zero polynomials, and characteristic equation coefficients.

(d) Heading which lists the system order, number of open loop poles and zeros, location of open poles and zeros, and root locus computation data that was input.

(e) Root locus points and corresponding gains, listed 3 branches per page. Each branch begins on an open loop pole within the computation area or at the boundary, and ends on an open loop zero or at the boundary. If there are more open loop zeros than poles, branches begin on zeros and end on poles.

(f) Plot of the root locus over the computation area. Symbols used on the plot correspond to branch numbers as follows: A - Branch No. 1, B - Branch No. 2, ..., J - Branch No. 10. For systems with more than 10 branches, A is used again for Branch No. 11, etc.

3.2 Example Problem

An example problem, exercising the still air random tracking longitudinal mode with airframe characteristics computed, is detailed in this section. The block diagram, equations, and the computer printout are all shown. This example is for the F-4 flying at M = 1.5 at an altitude of 35,000 ft. The block diagram of the system is presented in Figure 4. The pilot parameters K_p, T_1 and T_2 were assigned the computer optimization symbols P(1), P(2), and P(3) respectively. The equations and subscript matrix array location for the control system and acceleration are detailed below. The airframe equations of motion, including the elevator deflection terms, are automatically inserted into the first 4 rows and 5 columns. The other equations starting in the fifth row along with their numerical index location in the A matrix (in parentheses under the equations) are:

(5) $\Theta_e + \Theta = \Theta_c$
(89) (17) (5–B Matrix)

(6) $P(1)\ [0.2P(2)\ S^2 + (0.2 - 2P(2))S - 2]\ \Theta_e$
(90 – Inserted by BLANKI)

$+\ [0.2P(3)\ S^2 + (0.2 + 2P(3))\ S + 2]\ A = 0$
(66 – Inserted by BLANKI)

(7) $0.15S^2\ \Theta - (S + 1)\ A - (S + 1)\ B = 0$
(19) (67) (79)

(8) $B\ -\ (0.05\ S + 1)\ \delta e = 0$
(80) (56)

(9) $-\frac{V}{g}\dot{\Theta} + \frac{V}{g}\dot{\alpha} + (10^{-4}\ S + 1)\ N_Z = 0$
(21) (33) (105)

There are nine equations with the variables being, in column order, U, Θ, α, h, δ_e, A, B, Θ_e, and N_Z. A and B are arbitrarily assi ed variables to keep the equations simple. Equation 7 represents SAS on. The $10^{-4}S$ term is added to the N_Z term of equation 9 to increase the order of the transfer function denominator (see 4.0 Running Hints). A computer printout of this example follows.

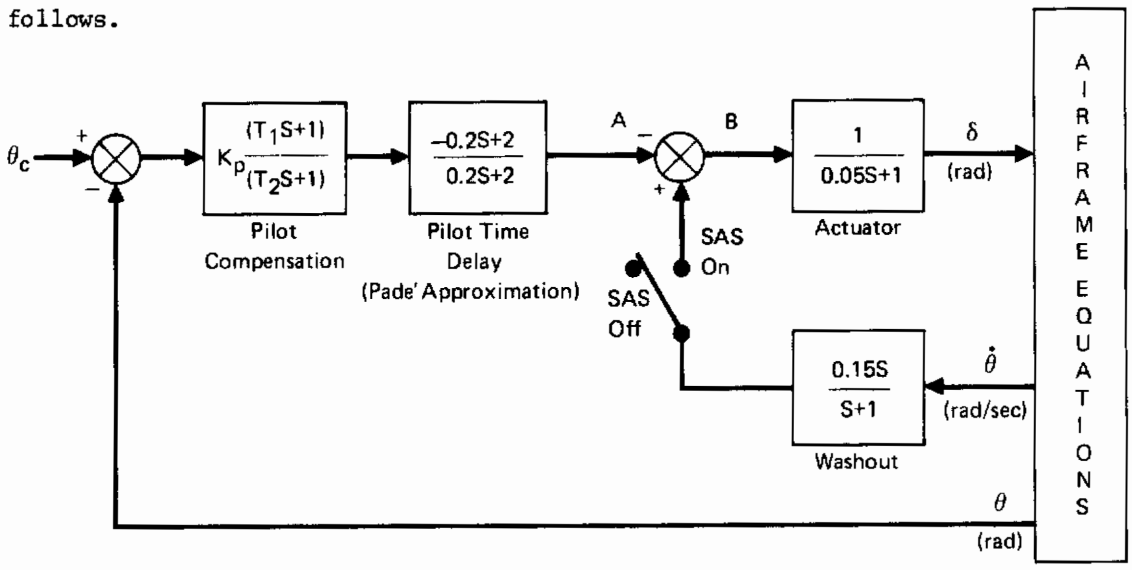


FIGURE 4
F-4 LONGITUDINAL AXIS RANDOM TRACKING MODEL

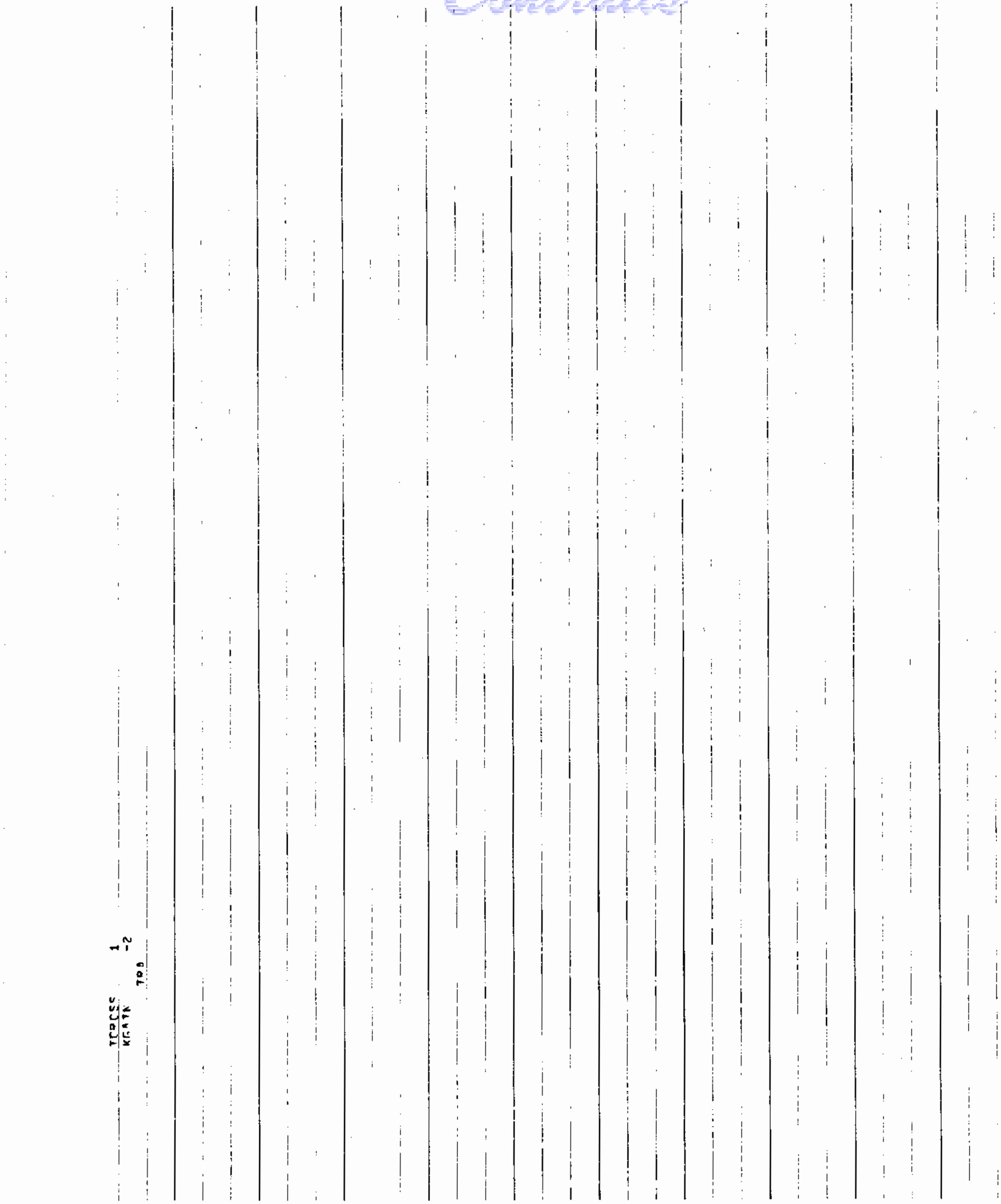
```
      SUBROUTINE  BLANK      TRACE                    CDC 6600 FTN V3.0-P336 OPT=1  73/08/30. 13.15.37.        PAGE    1

            SUBROUTINE BLANK
C
C           THIS SUBROUTINE IS FILLED IN BY USER. WHO MAKES THE A AND/OR B MATRICES
C                    FUNCTIONS OF THE P(*)*S
C           SIMULTANEOUSLY THE USER MAKES THE AORDER AND BORDER MATRICES COMPATIBLE
C                    WITH THE A AND B MATRICES RESPECTIVELY
C
C
C          BLANK COMPUTES THOSE ELEMENTS OF THE A MATRIX THAT
C          DEPEND ON THE PILOT PARAMETERS, P(I).  SKBJI COMPUTES
C          THE DETERMINANT OF THE A MATRIX (AS A HOMOGEOUS POLYNOMIAL
C          IN S).  POLYRT FINDS ROOTS OF POLYNOMIALS.
            COMMON COMON(4400)
            COMMON / 5 / A(12,12,11),DUM(15),B(12,12,11),DUM1(16)
            INTEGER AORDER(12,12),BORDER(12,12)
            DIMENSION P(5),ACOST(8)
            EQUIVALENCE (AORDER,COMON(   1))
            EQUIVALENCE (BORDER,COMON( 145))
            EQUIVALENCE (KNTISE,COMON( 514))
            EQUIVALENCE (P      ,COMON( 780))
            EQUIVALENCE (PR     ,COMON(4345))
            EQUIVALENCE (ACOST ,COMON( 695))
            ENTRY BLANKI
C           USER SUPPLIED BLANK INPUT FOLLOWS THIS CARD
            IF(KNTISE.NE.0)GO TO 30
            DO 20 I=1,12
            AORDER(1,I)=-1
            AORDER(I,1)=-1
            AORDER(4,I)=-1
   20       AORDER(I,4)=-1
            AORDER(1,1)=0
            AORDER(4,4)=0
            A(1,1,1)=1.
            A(4,4,1)=1.
            AORDER(6,8)=2
            AORDER(6,6)=2
            TAU=.2
   30       A(6,8,1)=-2.*P(1)
            A(6,8,2)=P(1)*(TAU-2.*P(2))
            A(6,8,3)=P(1)*TAU*P(2)
            A(6,6,1)=2.
            A(6,6,2)=TAU+2.*P(3)
            A(6,6,3)=TAU*P(3)
            RETURN
            ENTRY COSTI
C           USER SUPPLIED COST FUNCTION EQUATIONS FOLLOW THIS CARD, IF DESIRED
            RETURN
            ENTRY PRATEI
C           USER SUPPLIED PILOT RATING EQUATION FOLLOWS THIS CARD, IF DESIRED
            SIGT=ACOST(1)**.5
            SIGN2=ACOST(2)**.5
            PHI1=ATAN(P(2))-ATAN(P(3))
            PHI1=ABS(PHI1)
            IF(P(1).EQ.0.)PHI1=0.
            PR=3.9*SIGT+.03*SIGN2+8.6/SIGN2+1.6*PHI1
```

```
      SUBROUTINE BLANK     TRACE                          CDC 6600 FTN V3.0-P336 OPT=1  73/08/30. 13.15.37.      PAGE    2
                 RETURN
                 END
```

```
         CATALOG OF ?               FILE     4                                      73/08/30. 13.17.19.        PAGE     5
REC      NAME       TYPE          LENGTH    CKSUM      DATE      COMMENTS

  1    * ECF *          SUM =             0
                   TITLE   RCD 9F-4 AT M=1.5 H=35K WITH SAS
                   RHO              .00073?
                   V                1460.
                   SREFE            530.
                   CHORD            16.04
                   ALT              35000.
                   WTSTR            40424.
                   AIYZE            135008.
                   CXA              -.058
                   CXU              -.0135
                   CXH              -.00016
                   CZA              -3.33          Aerodynamic and
                   CZD              -1.7           physical data
                   CZU              .015
                   CZH              .0062
                   CZD              -.2234
                   CMA              -.728
                   CMAD             -.6767
                   CMQ              -2.57
                   CMU              .0355
                   CMH              .0005
                   CMD              -.3278
                   ACRDER  INT  0,-1,2,-1,1                                          17
                   AORDER  INT  1                                                    33
                   ACRDER  INT  1                                                    56
                   ACRDER  INT  1                                                    67
                   ACRDER  INT  1,0                                                  79
                   AORDER  INT  0                                                    89   Other equation input
                   ACRDER  INT  1                                                   105   except those being
                   AINFUT       1.,0.,0.,.15                                              varied in optimization
                   AINFUT       0.,-45.4,0.,45.4                                      5
                   AINFUT       -1.,-.05,-1.,-1.,-1.,-1.,1.,1.,1.,.0001               9
                   BORDER  INT  0                                                     5
                   BINFUT       1.
                   MCDNCR  INT  2
                   MODDCR  INT  4                     Model data
                   FCLYA        331.,183.,-21.6
                   ECLYD        331.,473.,104.6,15.72,1.
                   CNSTMN       0.,0.,0.
                   CNSTMX       1000.,5.,10.
                   CUTCFF       .05,.005,.005
                   DEL          .05,.005,.005
                   ISETRM       2
                   KEARAM       3
                   ISETOL       1
                   ICLISE       .01                  Optimization data
                   JUMF         .2
                   AIN     INT  1,1
                   NMOT    INT  1,4
                   NCUT    INT  2,9
                   NTYFE   INT  0,0
                   WEIGHT       1.,.0001
                   NUMEQ        0
                   FLTFI        .9,.15,.26  Initial parameters
                   ICOST   INT  1,1  Use standard cost function
                   NISEAF       1  ]Get time and frequency response
                   ISEEUX       1
                   IACIRC       1  ]  Impulse for ISE 1
                   JLOCUS       1  Get root locus on P(1)
```



```
                    F-4 AT M=1.5 H=35K WITH SAS
INPUT DATA (NON-DIMENSIONAL) PER RADIAN    LONGITUDINAL MODE
---------------------------------------    -----------------
CXA =-5.8000E-02   CXAD = 0.             CXQ = 0.             CXD =    0.          CXU =-1.3500E-02   CXH =-1.6000E-04
CZA =-3.3300E+00   CZAD = 0.             CZQ =-1.7000E+00     CZD =   -2.2340E-01  CZU = 1.5000E-02   CZH = 6.2000E-03
CMA =-7.2800E-01   CMAD =-6.7670E-01     CMQ =-2.5700E+00     CMD =   -3.2780E-01  CMU = 3.5500E-02   CMH = 5.0000E-04
VEL.= 1.4600E+03   S    = 5.3000E+02     IY  = 1.3801E+05     ALFAA = 0.           ALT = 3.5000E+04   GAMA= 0.
RHO = 7.3700E-04   CHI  = 4.0424E+04     MAC = 1.6040E+01

DIMENSIONAL STABILITY DERIVATIVES
---------------------------------
XA =-1.9219E+01    XAD = 0.              XQ = 0.              XD = 0.              XU =-6.1277E-03    XH =-3.6312E-05    XW =-1.3163E-02
ZA =-1.1034E+03    ZAD = 0.              ZQ =-3.0942E+00      ZD =-7.4023E+01      ZU = 5.8085E-03    ZH = 1.4071E-03    ZW =-7.5575E-01
MA =-3.5229E+01    MAD =-1.7986E-01      MQ =-6.8308E-01      MD =-1.5861E+01      MU = 2.3530E-03    MH = 1.6571E-05    MW =-2.4127E-02

                DELTA/NZ =-7.7050E-02                             NZ/A = 2.9183E+01                     C = 7.8549E+02

        ROOTS (COMPLEX FORM)
    -.5358E-02
    -.2043E-02          .4803E-01
    -.2043E-02         -.4803E-01
    -.8075E+00          .5917E+01
    -.8075E+00         -.5917E+01

        SHORT PERIOD MODE
   ZETA =  .13522E+00                              WN =  .59716E+01 RAD/SEC
                                                      =  .95041E+00 CYCLES/SEC                        (WN)SQUARED/NZA =  .12220E+01

   UNDAMPED PERIOD =  .10522E+01              TIME TO HALF AMP. =  .55939E+00             TIME TO ONE TENTH AMP. =  .28515E+01
     DAMPED PERIOD =  .10619E+01            CYCLES TO HALF AMP. =  .80834E+00           CYCLES TO ONE TENTH AMP. =  .26852E+01
                                   ONE OVER CYCLES TO HALF AMP. =  .12371E+01  ONE OVER CYCLES TO ONE TENTH AMP. =  .37241E+00

        LONG PERIOD MODE
   ZETA =  .42594E-01                              WN =  .48075E-01 RAD/SEC
                                                      =  .76513E-02 CYCLES/SEC

   UNDAMPED PERIOD =  .13070E+03              TIME TO HALF AMP. =  .33922E+03             TIME TO ONE TENTH AMP. =  .11269E+04
     DAMPED PERIOD =  .13082E+03            CYCLES TO HALF AMP. =  .25931E+01           CYCLES TO ONE TENTH AMP. =  .86142E+01
                                   ONE OVER CYCLES TO HALF AMP. =  .38563E+00  ONE OVER CYCLES TO ONE TENTH AMP. =  .11609E+00

   CHARACTERISTIC EQUATION COEFFICIENTS
   ------------------------------------
   A =-1.45000E+03 S**5 B =-2.37169E+03 S**4 C =-5.20898E+04 S**3 D =-4.97261E+02 S**2 E =-1.21499E+02 S**1 F =-6.44745E-01 S**0
```

F-4 AT M=1.5 H=35K WITH SAS

BEGIN CHECK OF DATA

DATA IS FOR FIRST ISF

F-4 AT M=1.5 H=35K WITH SAS

A MATRIX

U	θ	α	H	δ_e	A	B	θ_e	N_Z			
0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
1.0000E+03	3.2174E+01	1.9219E+01	3.6312E-05	-0.	0.	0.	0.	0.	0.	0.	0.
1.0000E+00	0.	-0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	1	1	-1	0	-1	-1	-1	-1	-1	-1	-1
-6.[illegible]E-03	0.	1.1374E+03	-1.4071E-03	7.4023E+01	0.	0.	0.	0.	0.	0.	0.
0.	-1.4559E+02	1.4600E+03	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	2	1	-1	0	-1	-1	-1	-1	-1	-1	-1
-2.2[illegible]E-03	0.	3.5225E+01	-1.6671E-05	1.5861E+01	0.	0.	0.	0.	0.	0.	0.
0.	6.8308E-01	1.7986E-01	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	1.3100E+00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1
-0.	-1.4500E+03	1.4600E+03	1.0000E+00	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	1.0000E+00	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

-1	0	-1	-1	-1	-1	-1	0	-1	-1	-1	-1
0.	1.0000E+00	0.	0.	0.	0.	0.	1.0000E+00	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	2	-1	2	-1	-1	-1	-1
0.	0.	0.	0.	0.	2.0000E+00	0.	-2.0000E+00	0.	0.	0.	0.
0.	0.	0.	0.	0.	8.5516E-01	0.	-1.5590E-01	0.	0.	0.	0.
0.	0.	0.	0.	0.	6.5616E-02	0.	3.5590E-02	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	2	-1	-1	-1	1	1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	-1.0000E+00	-1.0000E+00	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	-1.0000E+00	-1.0000E+00	0.	0.	0.	0.	0.
0.	1.5000E-01	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	1	-1	0	-1	-1	-1	-1	-1
0.	0.	0.	0.	-1.0000E+00	0.	1.0000E+00	0.	0.	0.	0.	0.
0.	0.	0.	0.	-5.0000E-02	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

-1	1	1	-1	-1	-1	-1	-1	1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	1.0000E+00	0.	0.	0.
0.	-4.5490E+01	4.5490E+01	0.	0.	0.	0.	0.	1.0000E-04	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 BT M=1.5 H=35K WITH SAS Q MATRIX

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
1.000(E+00)	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

	MODEL 1 NUMERATOR ORDER = 2	MODEL 2 NUMERATOR ORDER = -1	MODEL 3 NUMERATOR ORDER = -1	MODEL 4 NUMERATOR ORDER = -1	MODEL 5 NUMERATOR ORDER = -1	MODEL 6 NUMERATOR ORDER = -1	MODEL 7 NUMERATOR ORDER = -1	MODEL 8 NUMERATOR ORDER = -1
0	3.3150E+02	0.	0.	0.	0.	0.	0.	0.
1	1.8300E+02	0.	0.	0.	0.	0.	0.	0.
2	-2.1600E+01	0.	0.	0.	0.	0.	0.	0.
3	0.	0.	0.	0.	0.	0.	0.	0.
4	0.	0.	0.	0.	0.	0.	0.	0.
5	0.	0.	0.	0.	0.	0.	0.	0.
6	0.	0.	0.	0.	0.	0.	0.	0.
7	0.	0.	0.	0.	0.	0.	0.	0.
8	0.	0.	0.	0.	0.	0.	0.	0.
9	0.	0.	0.	0.	0.	0.	0.	0.
10	0.	0.	0.	0.	0.	0.	0.	0.

	DENOMINATOR ORDER = 4	DENOMINATOR ORDER = 0	DENOMINATOR ORDER = 0	DENOMINATOR ORDER = 0	DENOMINATOR ORDER = 0	DENOMINATOR ORDER = 0	DENOMINATOR ORDER = 0	DENOMINATOR ORDER = 0
0	3.3100E+02	1.0000E+00	1.0000E+00	1.0000E+00	1.0000E+00	1.0000E+00	1.0000E+00	1.0000E+00
1	4.7300E+02	0.	0.	0.	0.	0.	0.	0.
2	1.0460E+02	0.	0.	0.	0.	0.	0.	0.
3	1.9720E+01	0.	0.	0.	0.	0.	0.	0.
4	1.0000E+00	0.	0.	0.	0.	0.	0.	0.
5	0.	0.	0.	0.	0.	0.	0.	0.
6	0.	0.	0.	0.	0.	0.	0.	0.
7	0.	0.	0.	0.	0.	0.	0.	0.
8	0.	0.	0.	0.	0.	0.	0.	0.
9	0.	0.	0.	0.	0.	0.	0.	0.
10	0.	0.	0.	0.	0.	0.	0.	0.

2 ISE TERMS WILL BE CALCULATED FOR EACH ISE

NIN	NMOD	NOUT	NSYST	NTYPE	WEIGHT
1	1	2	1	0	1.000E+00
1	4	0	1	0	1.000E-04

3 PARAMETERS WILL BE VARIED TO OPTIMIZE ISE

CNSTMN	CNSTMX	CUTOFF	DEL	P
0.	1.000E+03	5.000E-02	5.000E-02	4.000E-01
0.	5.000E+03	5.000E-03	5.000E-03	1.500E-01
0.	1.000E+01	5.000E-03	5.000E-03	2.600E-01

SYSTEM 1 IS A 9 X 9 MATRIX

ISETCL = 1 THEREFORE THE PROGRAM WILL STOP WHEN EACH P IS WITHIN ITS RESPECTIVE CUTOFF AND ISE IS WITHIN TCLISE = 1.000E-02

F-4 AT M=1.5 H=35K WITH SAS

END CHECK OF DATA

```
INNER MINI
INITIAL POINT IS
   -.179E+00    .800E+00    .150E+00    .260E+00
STAPLE STARTING PARAMETERS
   -.187E+00    .850E+00    .150E+00    .260E+00
```

F-4 AT M=1.5 H=35K WITH SAS

KOUNT	JUMP	ISE TERMS				SCALE			
0- 1	2.000E-01	1	3.359350E-01	2	6.327216E+02	1	0.	2	0.
		3	0.	4	0.	3	0.	4	0.
TOTISE	NJF	5	0.	6	0.	5	0.	6	0.
3.99207181E-01	0.	7	0.	8	0.	7	0.	8	0.

C						DIRCOS					
1	8.500000E-01	2	1.500000E-01	3	2.600000E-01	1	0.	2	0.	3	0.

FOR ISE TERM 1

SYSTEM NUMERATOR		SYSTEM DENOMINATOR		NUMERATOR ROOTS		DENOMINATOR ROOTS	
ORDER = 5		ORDER = 8					
2.531862E+04	S** 0	2.531862E+04	S** 0				
5.593123E+04	S** 1	1.700776E+05	S** 1	1.0000E+01	0.	-1.8727E-01	0.
4.220410E+04	S** 2	1.982543E+05	S** 2	-6.4352E-01	0.	-1.0229E+00	0.
1.001486E+03	S** 3	6.643399E+04	S** 3	-1.0000E+00	0.	-1.0679E+00	5.7807E+00
-5.900625E+02	S** 4	1.301255E+04	S** 4	-6.6667E+00	0.	-1.0679E+00	-5.7807E+00
-5.981522E-02	S** 5	1.856955E+03	S** 5	-1.0000E+04	0.	-5.0852E+00	0.
		1.386047E+02	S** 6			-1.4017E+01	1.2945E+00
		3.809842E+00	S** 7			-1.4017E+01	-1.2945E+00
		3.796000E-04	S** 8			-1.0000E+04	0.

FOR ISE TERM 2

SYSTEM NUMERATOR		SYSTEM DENOMINATOR		NUMERATOR ROOTS		DENOMINATOR ROOTS	
ORDER = 6		ORDER = 8					
0.	S** 0	2.531862E+04	S** 0				
1.149465E+06	S** 1	1.700776E+05	S** 1	1.4085E+01	0.	-1.8727E-01	0.
1.205796E+06	S** 2	1.982543E+05	S** 2	1.0000E+01	0.	-1.0229E+00	0.
3.331877E+04	S** 3	6.643399E+04	S** 3	0.	0.	-1.0679E+00	5.7807E+00
-2.328073E+04	S** 4	1.301255E+04	S** 4	-1.0000E+00	0.	-1.0679E+00	-5.7807E+00
-1.828246E+02	S** 5	1.856955E+03	S** 5	-6.6667E+00	0.	-5.0852E+00	0.
8.569669E+01	S** 6	1.386047E+02	S** 6	-1.4285E+01	0.	-1.4017E+01	1.2945E+00
		3.809842E+00	S** 7			-1.4017E+01	-1.2945E+00
		3.796000E-04	S** 8			-1.0000E+04	0.

F-4 AT M=1.5 H=35K WITH SAS

KOUNT	JUMP	ISE TERMS				SCALE			
5- 58	5.000E-02	1	3.252383E-01	2	5.854296E+02	1	0.	2	0.
		3	0.	4	0.	3	0.	4	0.
TOTISE	AJP	5	0.	6	0.	5	0.	6	0.
3.837812975E-01	5.573E-03	7	0.	8	0.	7	0.	8	0.

F						DIPOCS					
1	8.924881E-01	2	1.779478E-01	3	3.250823E-01	1	-3.475125E-01	2	8.469645E-01	3	-4.023508E-01

FOR ISE TERM 1

SYSTEM NUMERATOR		SYSTEM DENOMINATOR		NUMERATOR ROOTS		DENOMINATOR ROOTS	
ORDER = 5		ORDER = 8					
2.658420E+04	S** 0	2.658420E+04	S** 0				
6.956984E+04	S** 1	1.741162E+05	S** 1	1.0000E+01	0.	-1.9545E-01	0.
4.613701E+04	S** 2	2.092777E+05	S** 2	-6.4352E-01	0.	-1.0224E+00	0.
2.016522E+03	S** 3	7.622974E+04	S** 3	-1.0000E+00	0.	-1.1186E+00	5.7658E+00
-7.349155E+02	S** 4	1.503950E+04	S** 4	-5.6196E+00	0.	-1.1186E+00	-5.7658E+00
-7.351166E-02	S** 5	2.218367E+03	S** 5	-1.0000E+04	0.	-4.0013E+00	0.
		1.710634E+02	S** 6			-1.4105E+01	1.5075E+00
		4.807086E+00	S** 7			-1.4105E+01	-1.5075E+00
		4.790002E-04	S** 8			-1.0000E+04	0.

FOR ISE TERM 2

SYSTEM NUMERATOR		SYSTEM DENOMINATOR		NUMERATOR ROOTS		DENOMINATOR ROOTS	
ORDER = 6		ORDER = 8					
0.	S** 0	2.658420E+04	S** 0				
1.206923E+06	S** 1	1.741162E+05	S** 1	1.4085E+01	0.	-1.9545E-01	0.
1.259800E+06	S** 2	2.092777E+05	S** 2	1.0000E+01	0.	-1.0224E+00	0.
6.530846E+04	S** 3	7.622974E+04	S** 3	0.	0.	-1.1186E+00	5.7658E+00
-2.801534E+04	S** 4	1.503950E+04	S** 4	-1.0000E+00	0.	-1.1186E+00	-5.7658E+00
-3.394963E+02	S** 5	2.218367E+03	S** 5	-5.6196E+00	0.	-4.0013E+00	0.
1.067453E+02	S** 6	1.710634E+02	S** 6	-1.4285E+01	0.	-1.4105E+01	1.5075E+00
		4.807086E+00	S** 7			-1.4105E+01	-1.5075E+00
		4.790002E-04	S** 8			-1.0000E+04	0.

```
F-4 AT M=1.5 H=35K WITH SAS

AMEN BROTHER, YOU HAVE ARRIVED
----YOU HAVE HIT A NEW LOW----

++++++++++++++++++++++++++++++++++++++++++++++++++
+                                                +
+   THE CALCULATED PILOT RATING IS  3.531        +
+                                                +
++++++++++++++++++++++++++++++++++++++++++++++++++
```

F-4 AT M=1.5 H=35K WITH SAS
FREQUENCY RESPONSE FOR ISE NO. 1

THE INPUT NUMERATOR POLYNOMIAL IS OF ORDER 5
ITS COEFFICIENTS IN DESCENDING ORDER ARE-
-7.3511665E-02 -7.3431546E+02 2.0165223E+03 4.6137008E+04 6.9969842E+04 2.6584199E+04

THE INPUT DENOMINATOR POLYNOMIAL IS OF ORDER 8
ITS COEFFICIENTS IN DESCENDING ORDER ARE-
4.7900016E-04 4.2070857E+00 1.7106344E+02 2.2183669E+03 1.5039502E+04 7.5229743E+04
2.0927771E+05 1.7411616E+05 2.6584199E+04

OMEGA	REAL PART	IMAG PART	MAGN	MAGNDB	PHASE-DEG
.0100	9.9805E-01	-3.9076E-02	9.9882E-01	-1.0277E-02	-2.2421E+00
.0500	9.5419E-01	-1.8417E-01	9.7180E-01	-2.4845E-01	-1.0924E+01
.1000	8.4525E-01	-3.1266E-01	9.0123E-01	-9.0333E-01	-2.0299E+01
.1500	7.2346E-01	-3.7564E-01	8.1517E-01	-1.7750E+00	-2.7440E+01
.2000	6.1827E-01	-3.9347E-01	7.3285E-01	-2.6997E+00	-3.2473E+01
.3000	4.7560E-01	-3.7229E-01	6.0399E-01	-4.3794E+00	-3.8053E+01
.5000	3.5066E-01	-3.0522E-01	4.6444E-01	-6.6615E+00	-4.1086E+01
.7000	3.0162E-01	-2.6667E-01	4.0260E-01	-7.9025E+00	-4.1481E+01
1.0000	2.6730E-01	-2.4547E-01	3.6291E-01	-8.8039E+00	-4.2563E+01
1.5000	2.3279E-01	-2.5220E-01	3.4321E-01	-9.2888E+00	-4.7292E+01
2.0000	2.0031E-01	-2.7910E-01	3.4354E-01	-9.2804E+00	-5.4332E+01
2.5000	1.6281E-01	-3.1490E-01	3.5450E-01	-9.0078E+00	-6.2660E+01
3.0000	1.1616E-01	-3.5689E-01	3.7532E-01	-8.5120E+00	-7.1971E+01
3.5000	5.3889E-02	-4.0455E-01	4.0812E-01	-7.7842E+00	-8.2412E+01
4.0000	-3.6311E-02	-4.5542E-01	4.5686E-01	-6.8043E+00	-9.4559E+01
4.5000	-1.7624E-01	-4.9563E-01	5.2603E-01	-5.5798E+00	-1.0957E+02
5.0000	-3.8904E-01	-4.7313E-01	6.1254E-01	-4.2574E+00	-1.2943E+02
5.5000	-6.1886E-01	-2.7568E-01	6.7748E-01	-3.3821E+00	-1.5599E+02
6.0000	-6.3734E-01	7.2882E-02	6.4150E-01	-3.8561E+00	1.7348E+02
8.0000	-5.0446E-02	2.3084E-01	2.3629E-01	-1.2531E+01	1.0233E+02
10.0000	3.1447E-02	1.0708E-01	1.1160E-01	-1.9047E+01	7.3634E+01

F-4 AT M=1.5 H=35K WITH SAS
TIME RESPONSE FOR ISE NO. 1
USING UNIT IMPULSE INPUT

THE INPUT NUMERATOR POLYNOMIAL IS OF ORDER 6

ITS COEFFICIENTS IN DESCENDING ORDER ARE-

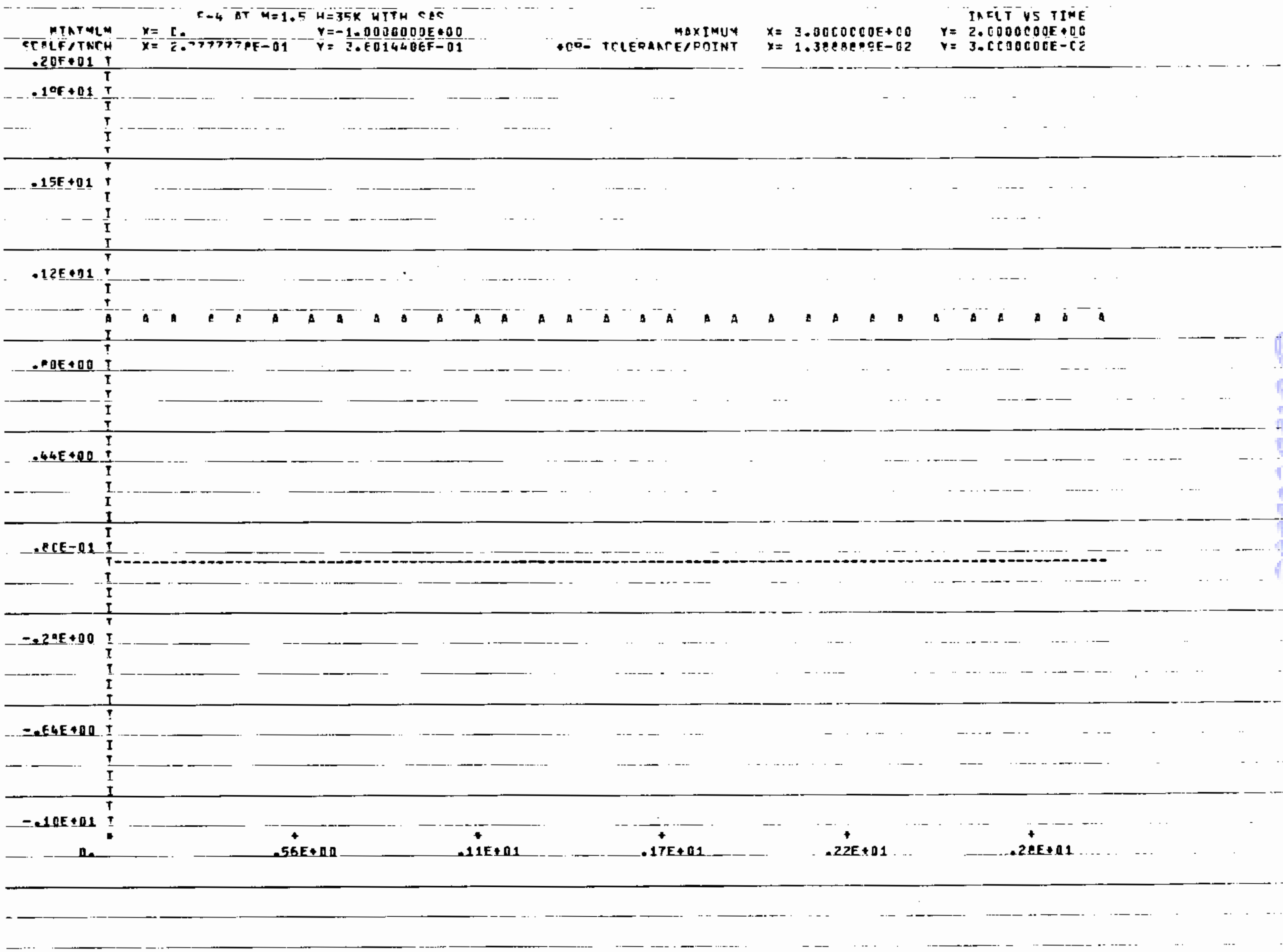
-7.3511665E-02   -7.3691546E+02   2.0165223E+03   4.6137008E+04   6.9969842E+04   2.6584199E+04
0.

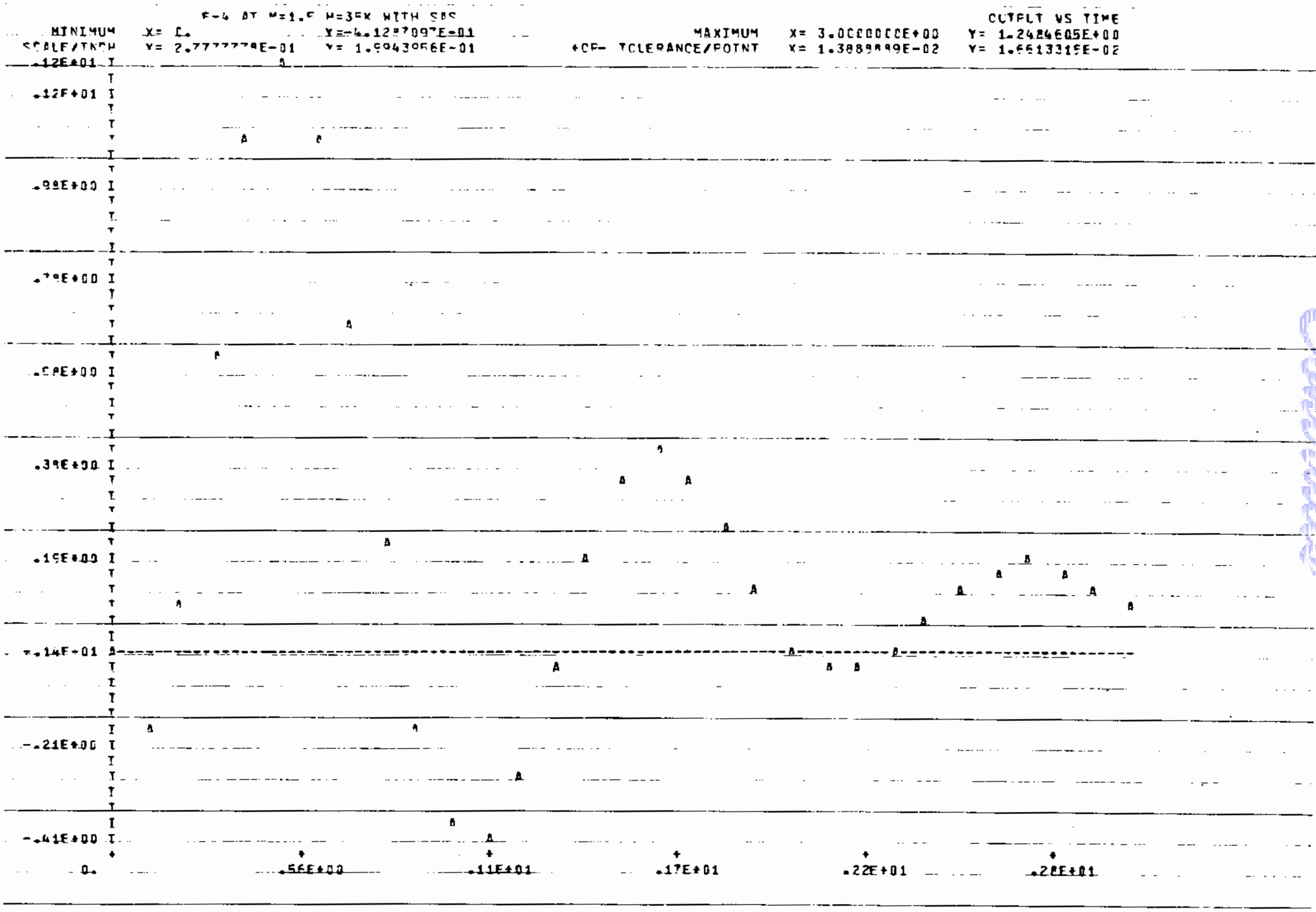
THE INPUT DENOMINATOR POLYNOMIAL IS OF ORDER 8

ITS COEFFICIENTS IN DESCENDING ORDER ARE-

4.7900016E-04   4.8070857E+00   1.7106344E+02   2.2183669E+03   1.5039302E+04   7.6229743E+04
2.0927771E+05   1.7411616E+05   2.6584199E+04

T=	R=	C=
0.000	1.00000	0.0000000
.100	1.00006	-.1718249
.200	1.00000	.0799272
.300	1.00000	.6235021
.400	1.00000	1.0824304
.500	1.00000	1.2484605
.600	1.00000	1.0899712
.700	1.00000	.7001233
.800	1.00000	.2317936
.900	1.00000	-.1651071
1.000	1.00000	-.3900075
1.100	1.00000	-.4128710
1.200	1.00000	-.2595492
1.300	1.00000	-.0380802
1.400	1.00000	.1930239
1.500	1.00000	.3533461
1.600	1.00000	.4080063
1.700	1.00000	.3609619
1.800	1.00000	.2460009
1.900	1.00003	.1105894
2.000	1.00000	-.0011369
2.100	1.00000	-.0609746
2.200	1.00000	-.0619313
2.300	1.00000	-.0160827
2.400	1.00000	.0528771
2.500	1.00000	.1190749
2.600	1.00000	.1626660
2.700	1.00000	.1745844
2.800	1.00000	.1571023
2.900	1.00000	.1208772
3.000	1.00000	.0800667


F-4 AT M=1.5 H=35K WITH SAS
INPUT VS TIME
MINIMUM X= 0. Y=-1.0000000E+00
SCALE/INCH X= 2.7777778E-01 Y= 2.6014406E-01
MAXIMUM X= 3.0000000E+00 Y= 2.0000000E+00
+00- TOLERANCE/POINT X= 1.3888889E-02 Y= 3.0000000E-02
.20E+01
.19E+01
.15E+01
.12E+01
.80E+00
.44E+00
.80E-01
-.28E+00
-.64E+00
-.10E+01
0.
.56E+00
.11E+01
.17E+01
.22E+01
.28E+01


F-4 AT M=1.5 H=35K WITH SAS
MINIMUM X= 0.
SCALE/INCH Y= 2.7777778E-01
X=-4.1287097E-01
Y= 1.9943956E-01
MAXIMUM
+CP- TOLERANCE/POINT
X= 3.0000000E+00
X= 1.3888889E-02
OUTPUT VS TIME
Y= 1.2424605E+00
Y= 1.6613315E-02
.12E+01
.12E+01
.99E+00
.79E+00
.59E+00
.39E+00
.19E+00
-.14E-01
-.21E+00
-.41E+00
0.
.56E+00
.11E+01
.17E+01
.22E+01
.28E+01

F-4 AT M=1.5 H=35K WITH SAS

A MATRIX FOR OPEN LOOP POLES POLYNOMIAL

0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
1.0000E+00	3.2174E+01	1.9218E+01	3.6312E-05	-0.	0.	0.	0.	0.	0.	0.	0.
1.0000E+00	0.	-0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	1	1	-1	0	-1	-1	-1	-1	-1	-1	-1
-6.6085E-03	0.	1.1034E+03	-1.4071E-03	7.4023E+01	0.	0.	0.	0.	0.	0.	0.
0.	-1.4569E+03	1.4600E+03	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	2	1	-1	0	-1	-1	-1	-1	-1	-1	-1
-2.3570E-03	0.	3.5225E+01	-1.6571E-05	1.5861E+01	0.	0.	0.	0.	0.	0.	0.
0.	6.8308E-01	1.7986E-01	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	1.0000E+00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1
-0.	-1.4500E+03	1.4500E+03	1.0000E+00	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	1.0000E+00	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

-1	0	-1	-1	-1	-1	-1	0	-1	-1	-1	-1
0.	1.0000E+00	0.	0.	0.	0.	0.	1.0000E+00	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	2	-1	2	-1	-1	-1	-1
0.	0.	0.	0.	0.	2.0000E+00	0.	-0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	8.5616E-01	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	8.5616E-02	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	2	-1	-1	-1	1	1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	-1.0000E+00	-1.0000E+00	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	-1.0000E+00	-1.0000E+00	0.	0.	0.	0.	0.
0.	1.5100E-01	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	1	-1	0	-1	-1	-1	-1	-1
0.	0.	0.	0.	-1.0000E+00	0.	1.0000E+00	0.	0.	0.	0.	0.
0.	0.	0.	0.	-5.0000E-02	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

-1	1	1	-1	-1	-1	-1	-1	1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	1.0000E+00	0.	0.	0.
0.	-4.5400E+01	4.6400E+01	0.	0.	0.	0.	0.	1.0000E-04	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

A MATRIX FOR CHARACTERISTIC EQUATION--PLOCUS = 1

0	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
1.0000E+00	3.2174E+01	1.9219E+01	3.6312E-05	-0.	0.	0.	0.	0.	0.	0.	0.
1.0000E+00	0.	-0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	1	1	-1	0	-1	-1	-1	-1	-1	-1	-1
-6.8088E-03	0.	1.1334E+03	-1.4071E-03	7.4023E+01	0.	0.	0.	0.	0.	0.	0.
0.	-1.4559E+03	1.4600E+03	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	2	1	-1	0	-1	-1	-1	-1	-1	-1	-1
-2.3530E-03	0.	3.5225E+01	-1.6571E-05	1.5961E+01	0.	0.	0.	0.	0.	0.	0.
0.	6.9308E-01	1.7986E-01	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	1.0190E+00	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	0	-1	-1	-1	-1	-1	-1	-1	-1
-0.	-1.4560E+03	1.4600E+03	1.0000E+00	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	1.0000E+00	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

-1	0	-1	-1	-1	-1	-1	0	-1	-1	-1	-1
0.	1.0000E+00	0.	0.	0.	0.	0.	1.0000E+00	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	-1	2	-1	2	-1	-1	-1	-1
0.	0.	0.	0.	0.	2.0000E+00	0.	-1.5000E+00	0.	0.	0.	0.
0.	0.	0.	0.	0.	7.2000E-01	0.	-8.0000E-02	0.	0.	0.	0.
0.	0.	0.	0.	0.	5.2000E-02	0.	2.4000E-02	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	2	-1	-1	-1	1	1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	-1.0000E+00	-1.0000E+00	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	-1.0000E+00	-1.0000E+00	0.	0.	0.	0.	0.
0.	1.5000E-01	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

-1	-1	-1	-1	1	-1	0	-1	-1	-1	-1	-1
0.	0.	0.	0.	-1.0000E+00	0.	1.0000E+00	0.	0.	0.	0.	0.
0.	0.	0.	0.	-5.0000E-02	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SES

-1	1	1	-1	-1	-1	-1	-1	1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	1.0000E+00	0.	0.	0.
0.	-4.5400E+01	4.5400E+01	0.	0.	0.	0.	0.	1.0000E-04	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.
0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.	0.

F-4 AT M=1.5 H=35K WITH SAS

OPEN LOOP POLES POLYNOMIAL IN ASCENDING ORDER

0	0.
1	1.0414632E+05
2	1.6314070E+05
3	7.4213220E+04
4	1.5774417E+04
5	2.2184404E+03
6	1.7105344E+02
7	4.8070857E+00
8	4.7900016E-04

CHARACTERISTIC EQUATION IN ASCENDING ORDER--FLOCUS = 1

0	2.9786613E+04
1	1.8254494E+05
2	2.1483552E+05
3	7.6472659E+04
4	1.4950972E+04
5	2.2183581E+03
6	1.7105344E+02
7	4.8070857E+00
8	4.7900016E-04

OPEN LOOP ZEROES POLYNOMIAL IN ASCENDING ORDER

0	2.9786613E+04
1	7.8399624E+04
2	5.1694913E+04
3	2.2594308E+03
4	-8.2344563E+02
5	-8.2367106E-02

ROOT LOCUS CALCULATION PROGRAM
RUN NUMBER 1

F-4 AT M=1.5 H=35K WITH SAS

SYSTEM ORDER - 8
NO. OF OPEN LOOP POLES - 8
NO. OF OPEN LOOP ZEROES - 5
TIME DELAY - 0.

OPEN LOOP POLE LOCATIONS REAL	IMAGINARY	MULTIPLICITY	OPEN LOOP ZERO LOCATIONS REAL	IMAGINARY	MULTIPLICITY
0.00000	0.00000	1	10.00000	0.00000	1
-1.02829	0.00000	1	-.64352	0.00000	1
-2.16756	5.95862	1	-1.00000	0.00000	1
-2.16756	-5.95862	1	-5.61963	0.00000	1
-3.04802	0.00000	1	-10000.00000	0.00000	1
-10.00000	0.00000	1			
-17.25490	0.00000	1			
-10000.00000	0.00000	1			

STEP SIZE - .12500
ACCURACY - 1.0E-04
GAIN MODE - NEGATIVE
GAIN CONSTANT - 1.00000
PLANT GAIN - -1.71956E+02

PRINT INTERVAL - 1
GAIN INTERVAL - 1
PLOT INTERVAL - 1
COMPARISON PLOT INTERVAL - 0

REGION OF CALCULATION - REAL - -12.0000 TO 4.0000
IMAG - -2.0000 TO 8.0000

CRITICAL POINT INDICATORS - P - OPEN LOOP POLE
Z - OPEN LOOP ZERO
E - BOUNDARY POINT
B - BREAKAWAY POINT
S - PROGRAM STUCK - BRANCH TERMINATED

F-4 AT M=1.5 H=35K WITH SAS

ROOT LOCUS - RUN NUMBER 1

	BRANCH NUMBER 1				BRANCH NUMBER 2				BRANCH NUMBER 3		
	LOCUS-REAL	LOCUS-IMAG	GAIN		LOCUS-REAL	LOCUS-IMAG	GAIN		LOCUS-REAL	LOCUS-IMAG	GAIN
P	0.00000	0.00000	0.00000	P	-1.02820	0.00000	0.00000	P	-2.16756	5.95862	0.00000
	-.12500	0.00000	.51036		-1.02820	0.00000	0.00000		-2.05048	5.91486	.05285
	-.25000	0.00000	1.25616	Z	-1.00000	0.00000			-1.93133	5.87706	.10774
	-.37500	0.00000	2.62333						-1.81047	5.84515	.28505
	-.50000	0.00000	6.18342						-1.68824	5.81897	.38522
	-.50000	0.00000	6.18342						-1.56496	5.79833	.48865
Z	-.64352	0.00000							-1.44090	5.78299	.59576
									-1.31533	5.77266	.70699
									-1.19146	5.76704	.82274
									-1.06646	5.76582	.94343
									-.94149	5.76867	1.06947
									-.81667	5.77527	1.20127
									-.69207	5.78531	1.33924
									-.56777	5.79850	1.48377
									-.44380	5.81454	1.63527
									-.32020	5.83317	1.79414
									-.19697	5.85413	1.96077
									-.07411	5.87718	2.13559
									.04837	5.90212	2.31855
									.17051	5.92872	2.51129
									.29231	5.95682	2.71382
									.41380	5.98622	2.92454
									.53501	6.01677	3.14626
									.65597	6.04833	3.37860
									.77669	6.08074	3.62199
									.89721	6.11390	3.87685
									1.01756	6.14767	4.14363
									1.13777	6.18195	4.42275
									1.25786	6.21664	4.71465
									1.37786	6.25164	5.01980
									1.49779	6.28687	5.33864
									1.61770	6.32218	5.67164
									1.73758	6.35760	6.01926
									1.85745	6.39305	6.38198
									1.97732	6.42847	6.76027
									2.09723	6.46379	7.15464
									2.21723	6.49879	7.56555
									2.33728	6.53360	7.99352
									2.45743	6.56811	8.43909
									2.57767	6.60228	8.90264
									2.69802	6.63605	9.38482
									2.81849	6.66939	9.88609
									2.93909	6.70225	10.40700
									3.05984	6.73460	10.94806
									3.18073	6.76638	11.50981
									3.30177	6.79758	12.09279
									3.42273	6.82814	12.69755
									3.54435	6.85805	13.32464
									3.66588	6.88726	13.97460

F-4 AT M=1.5 H=35K WITH SAS   ROOT LOCUS - RUN NUMBER 1

PAGE 2

BRANCH NUMBER 1			BRANCH NUMBER 2				BRANCH NUMBER 3		
LOCUS-REAL	LOCUS-IMAG	GAIN	LOCUS-REAL	LOCUS-IMAG	GAIN		LOCUS-REAL	LOCUS-IMAG	GAIN
							3.78759	6.91575	14.64799
							3.90948	6.94349	15.34536
						E	4.03154	6.97044	16.06728

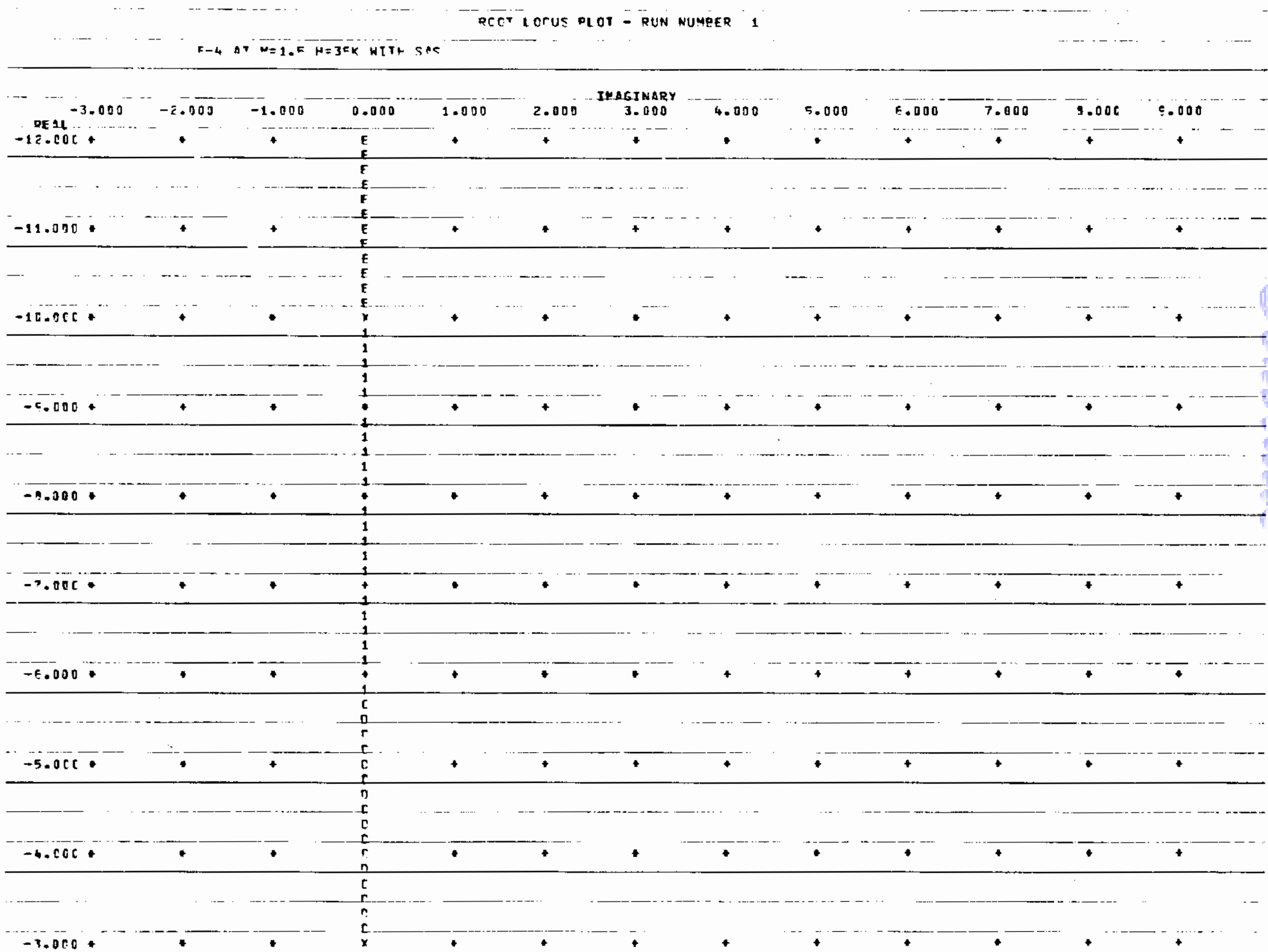
F-4 AT M=1.5 H=35K WITH SAS

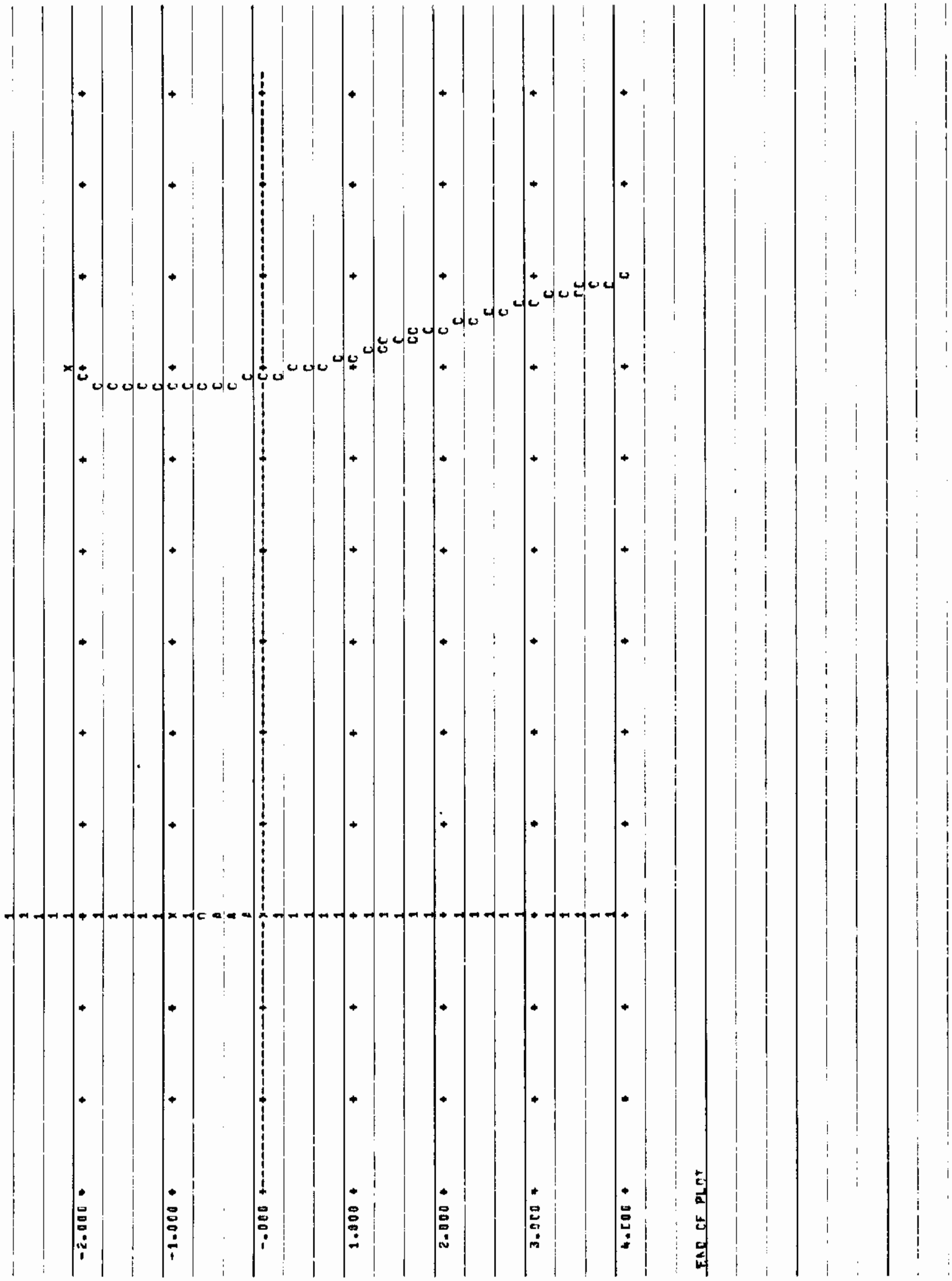
ROOT LOCUS - RUN NUMBER 1

	BRANCH NUMBER 4				BRANCH NUMBER 5		
	LOCUS-REAL	LOCUS-IMAG	GAIN		LOCUS-REAL	LOCUS-IMAG	GAIN
F	-3.04802	0.00000	0.00000	P	-10.00000	0.00000	0.00000
	-3.17302	0.00000	.09803		-10.12500	0.00000	.04256
	-3.29802	0.00000	.19883		-10.25000	0.00000	.08390
	-3.42302	0.00000	.30365		-10.37500	0.00000	.12405
	-3.54802	0.00000	.41406		-10.50000	0.00000	.16305
	-3.67302	0.00000	.53171		-10.62500	0.00000	.20091
	-3.79802	0.00000	.65873		-10.75000	0.00000	.23765
	-3.92302	0.00000	.79778		-10.87500	0.00000	.27328
	-4.04802	0.00000	.95224		-11.00000	0.00000	.30779
	-4.17302	0.00000	1.12662		-11.12500	0.00000	.34119
	-4.29802	0.00000	1.32701		-11.25000	0.00000	.37348
	-4.42302	0.00000	1.56203		-11.37500	0.00000	.40464
	-4.54802	0.00000	1.84419		-11.50000	0.00000	.43466
	-4.67302	0.00000	2.19257		-11.62500	0.00000	.46353
	-4.79802	0.00000	2.63780		-11.75000	0.00000	.49123
	-4.92302	0.00000	3.23237		-11.87500	0.00000	.51773
	-5.04802	0.00000	4.07465		-12.00000	0.00000	.54302
	-5.17302	0.00000	5.37304	E	-12.12500	0.00000	.56707
	-5.29802	0.00000	7.65978				
	-5.42302	0.00000	12.82053				
	-5.42302	0.00000	12.82053				
Z	-5.61963	0.00000					

IMAGINARY AXIS CROSSING POINTS - RUN NUMBER 1

IMAG. VALUE	GAIN
5.892049	2.245414


ROOT LOCUS PLOT - RUN NUMBER 1
F-4 AT M=1.5 H=35K WITH SAS
IMAGINARY
REAL
-3.000 -2.000 -1.000 0.000 1.000 2.000 3.000 4.000 5.000 6.000 7.000 8.000 9.000
-12.000
-11.000
-10.000
-9.000
-8.000
-7.000
-6.000
-5.000
-4.000
-3.000


-2.000
-1.000
-.000
1.000
2.000
3.000
4.000
END OF PLOT

4. RUNNING HINTS

In the course of running McPilot to get the pilot rating correlations, several program characteristics were discovered that affect the program operation. These are outlined here to aid the McPilot user and caution him on the applicability of some answers obtained.

A slowly changing or shallow minimum cost function, or local minimum can be encountered which makes it very difficult to find the actual minimum. It is sometimes necessary to experiment with different starting values of the pilot parameters to assure arriving at the true cost function minimum. Usual starting values were a small pilot gain (10^{-4}) and zero lead and lag. If the system optimized quickly in the region of the initial pilot parameter values, it was usually advisable to start at another point, say with a gain of 1 or 10, to determine if the program optimized to the same minimum value. If it does not, there may be local minimums in the cost function, or a shallow cost region exists. It sometimes helped to tighten the tolerances on P(N) and ISE. A rule of thumb is that the tolerance should be about 10^{-3} of the parameter value. Tolerance values nominally used for the still air random tracking task were 0.05 for gain, 0.005 for lead and lag, and 0.01 for the cost function. Time response, frequency response, and root locus analyses were used to assure being in the minimum region. If a cost function describes a trough shaped surface, it can take a long time to reach the minimum using the steepest descent technique, and very close tolerances must be specified.

For a given system, it is better to have a small matrix array with high order terms, than a large matrix array with low order terms. This reduces the running time for a case.

When continuing a case that ran out of time before optimizing, reduce DEL to about 0.1 (usually start a case with DEL = 1.0).

High frequency terms in the control system did not affect the results of McPilot. Eliminating them did, however, reduce the running time. Arbitrarily, all terms having a frequency of 20 rad/sec or greater and first order time constant terms of 0.025 seconds or less were eliminated.

It is advisable, when analyzing a complex system, to start with a simple version and progress to the complex system. In this way, the region where the complex system optimum occurs can be predicted, and the complex system may be checked against the simplified one.

Occasionally one of the optimization parameters is driven across its boundary value. If at the same time the system is driven unstable, it could take a long time for McPilot to drive the parameter back into the desired region. The most common occurence of this phenomenon is when a gain boundary is set at zero, and negative gains are unstable. The situation may be alleviated by inserting a statement in BLANKI that if P(N) exceeds a boundary, set the parameter equal to the boundary. This statement must occur in the initial segment of BLANKI prior to the point where the parameter is first used to compute a matrix array value. This disrupts the slope computation in the optimization loop; however, the overall effect is often to reduce the run time for a case.

The method of Krasovskii requires that the numerator of the Laplace transform of the response must be of less order than the denominator. If this condition is not met, McPilot will exit on an error and print a message to that effect. This situation occurred frequently for the N_Z terms of the cost functions used in the study. To overcome this problem, a high frequency filter $\left(\frac{1}{10^{-4}\,S + 1}\right)$ term was introduced in the N_Z equation. This term introduces an extraneous root which is of no consequence in the pilot rating correlation because the root magnitude is so large. The example problem (Section 3.2) used this technique.

APPENDIX A

CORRELATION EQUATIONS

The following sets of equations were developed in this study by correlating conventional aircraft handling qualities data based on system models for specified flight tasks. For a complete description of the correlations, the tasks performed, and the data used see Volume I. Unless otherwise specified, quasi-linear pilot models are required for these correlations.

Task: Still Air Random Tracking - Longitudinal Axis

Input Transient Analog (Laplace transform):

$\theta_c = 1$ (an impulse)

Cost Function:

$$J = \sigma_{\theta_{error}}^2 + 10^{-4}\sigma_{N_Z}^2$$

Pilot Rating Formula:

$$PR = 3.9\ \sigma_{\theta_{error}} + .03\sigma_{N_Z} + 8.6\ \frac{1}{\sigma_{N_Z}} + 1.6|\phi_1|$$

Note: $\sigma_{\theta_{error}}$ = r.m.s. pitch error between system and model where

$$\frac{\theta_{model}}{\theta_c} = \frac{-21.6S^2 + 183S + 331}{S^4 + 19.72S^3 + 104.6S^2 + 473S + 331}$$

Task: Rough Air Tracking - Longitudinal Axis

Input Transient Analog (Laplace transform):

$$Wg = \frac{\frac{12.25V}{1000}\ \frac{563}{V-50}\ S}{S^2 + \frac{V+50}{1000}S + \frac{.05V}{1000}}$$

Cost Function:

$$J = \sigma_\theta^2 + \sigma_{N_Z}^2$$

Pilot Rating Formula:

$$PR = 1. + 2.5|\phi_1| + 7.8\ \frac{\sigma\theta}{\sigma_{N_Z}} + 11.5\sigma_{NZ}$$

Task: Rough Air Tracking - Lateral Axis

Input Transient Analog (Laplace transform):

$$Vg = \frac{\frac{\sigma_g}{(0.752)}(S + .342)}{S^2 + 1.18S + .348}$$

Cost Function:

$$J = 75\ \sigma_\phi + 3.25\ (1-e^{-.77T_1})$$

Pilot Rating Formula:

$$PR = 1 + 71\ \sigma_\phi + 3.1\ (1-e^{-.77T_1})$$

Task: Still Air Random Tracking - Lateral Axis

Input Transient Analog (Laplace transform):

$$\phi_c = \frac{1.86}{S + 1}$$

Cost Function:

$$J = 2\sigma_\phi + (1 - e^{-.77T_1})$$

Pilot Rating Formula:

$$PR = 1 + 20\,X + 4.68\,(1 - e^{-.77T_1})$$

$$X = \sigma_\phi - 0.6, \quad \sigma_\phi > 0.6$$

$$= 0, \quad \sigma_\phi \leq 0.6$$

Task: Step Input - Longitudinal Tracking

Note: The Dual Mode Pilot developed in Volume I was used as the pilot model for this correlation. Care should be exercised using this pilot model as it is not substantiated for this task.

Input (Laplace transform):

$$\theta_c = \frac{1}{s} \text{ (a step)}$$

Cost Function:

$$J = ISE_\theta + 10^{-4}\, ISE_{N_Z}$$

Pilot Rating Formula:

$$PR = 3.2\sqrt{ISE_\theta} + 0.076\sqrt{ISE_{N_Z}} + 4.5\frac{1}{\sqrt{ISE_{N_Z}}} + 6.8|\phi_1|$$

Note: ISE_θ is the ISE of the difference between the system pitch response and a model response where

$$\frac{\theta_{model}}{\theta_c} =$$

$$\frac{1.23S^7 + 18.1S^6 - 17.0S^5 - 1526S^4 - 1.02\times10^4 S^3 - 2.8\times10^4 S^2 - 3.43\times10^4 S - 1.54\times10^4}{-.032S^9 - 1.42S^8 - 25.4S^7 - 261S^6 - 1811S^5 - 8759S^4 - 2.8\times10^4 S^3 - 5.21\times10^4 S^2 - 4.79\times10^4 S - 1.54\times10^4}$$

APPENDIX B

PROGRAM LISTING

The McPilot program is divided into five primary level overlays to reduce the required memory. During execution, the overlays are called into memory and executed as requested. Each overlay is numbered with an ordered pair of numbers (I,J). I denotes a primary level overlay and J the secondary level. The initial or main overlay which always remains in memory has levels (0,0). Synopsis of subprograms in the main overlay is shown, and is followed by a complete listing of the McPilot program.

MCPILOT	Initializes labeled common blocks to zero.
OVRLAY	Calls in the data reading overlay or the various execution overlays.
CLASS	Writes classifications on output pages.
BLOCK DATA	Is the directory of input symbolic names.
BLOCK DATA DATAB	Is an analogue of the input directory to permit filling the A and B matrices directly.
FUNCTION NEW	Used to place data in four-dimensional array.
BLANK	Is a subroutine, partially written by the user, that makes any A or B element of the system equation $A\underline{x} = B\underline{u}$ a function of the parameters to be optimized.

Synopsis of subprograms in the primary level overlay (1,0):

INPUT	Drives the reading routine; gives non-zero nominal values to selected constants; relocates data from input arrays to the A and B matrices.
DIPLAC	Right justifies a number so that displacement (col. 67-72) on an input card may be punched anywhere in the field.
READA	Provides a general method of reading a variable field data card. Data may be read into symbolic locations in memory.
PACBCD	Packs BCD words into six character words
PACKR	Packs BCD characters into words.
READ31	Converts from BCD to octal, floating point, and integer numbers.
DSERCH	Provides a method of searching the directory to find the common subscript corresponding to a BCD argument.
SVI	Provides a method of placing a number of variables and other arrays into one composite array.

Synopsis of subprograms in the primary level overlay (2,0)

AERO	Is the main program of the AERO overlay; it calls the subroutines demanded by the user-input control indicators.
IVXIV	Calculates the dimensional stability derivatives for longitudinal motions.
STFIX	Drives the analysis of the stick-fixed behavior of the subject airplane/control system.
FILLIT	Initializes a 4 x 4 matrix with polynomial terms and fills in the variables. Longitudinal mode.
DETVAL	Evaluates the determinant of a 4 x 4 matrix with polynomial terms.
DMULR	Determines the roots of a polynomial by an iterative method.
IIIX3	Calculates the dimensional stability derivatives for lateral motions and performs an analysis of the behavior of the subject airplane/control system.
GUST	Calculates and inserts into the matrix the equations for gust for either lateral or longitudinal mode.
FIL3X3	Initializes a 3 x 3 matrix with polynomial terms and fills in the variables.

Synopsis of subprograms in the primary level overlay (3,0)

MISER	Is the controlling program of the MISER overlay.
WRITE1	Prints all output except diagnostic messages.
MINIMI	Finds by golden section the point at which ISE is a minimum along the gradient.
GRADI	Calculates the gradient at a point P.
ISE	Calculates the total ISE at a point P.
SKBJI	Calculates the determinant of a matrix whose elements are polynomials in S.
POLYRT	Calculates the real and complex roots of a real polynomial.
PMPY	Multiplies two polynomials together.
PSUB	Subtracts one polynomial from another.
POLPRO	Multiplies any number of polynomials together.
SIMQ	Solves a set of simultaneous equations.
MINI	Used to find a set of pilot parameters for which the system is stable.
COST	Determines the value of the largest positive part of the system Eigenvalues.

Synopsis of subprograms in primary level overlay (4,0)

AUXIL	Calculates the steady-state frequency response of a transfer function input as a numerator/denominator polynomial ratio.
TIMRP	Calculates time response for a transfer function input.
PCCUT AND INITUP	Determines numerically the solution of an arbitrary set of simultaneous ordinary differential equations.
PLOTR	A point plot routine which plots a graph of one or more curves from given sets of rectangular coordinates.
PACK	Used to exchange bits or bytes from one word to another.
DISCOT2	Provides for linear interpolation of functions of one or two independent variables.

Synopsis of subprograms in primary level overlay (5,0)

MATLOC	Used in conjunction with SKBJI with logic to set specified elements to zero and one thus finding the open loop poles and zeroes which are passed to RTLCSV.
SKBJI	Calculates the determinant of a matrix whose elements are polynomials in S.
PSUB	Subtracts one polynomial from another.
FLIPPO	Flips over an array of N values such that ARRAY(N) = ARRAY(1), etc.
RLPLOT	Generates a root locus printer - plot.
RTLOC	Does bulk of computation work to generate root locus, finds gain, etc.
BLOCK DATA	Initializes variable names to given constants.
ARCTAN	Used to calculate arctangents.
POLYRT	Calculates the real and complex roots of a real polynomial.
RTLCSV	Control subroutine for generating the plots, lists of points, etc.

Note: Many variables are passed from one subroutine to another by blank common and to save storage many variables are equivalenced on top of others that are no longer needed.

```
      OVERLAY(MCPILOT,0,0)
      PROGRAM MCPILOT (INPUT,OUTPUT,TAPE5=INPUT,TAPE6=OUTPUT)
      COMMON COMON(4400)
      COMMON / 4 / LCOUNT,NAME(200),LOC(200)
      COMMON / 5 / TABLE(3200)
      COMMON / 6 / DUMMY(310)
      DO 14 I=1,4400
   14 COMON(I) = 0
      DO 15 I=1,3200
   15 TABLE(I) = 0
      DO 16 I=1,310
16    DUMMY(I)=0.
      CALL OVERLAY (7LMCPILOT,1,0,6HRECALL)
      END
```

```
      SUBROUTINE OVRLAY
      COMMON DUM(747),ILINK
      GO TO (1,2,3,4,5),ILINK
    1 CALL OVERLAY (7LMCPILOT,1,0,6HRECALL)
    2 CALL OVERLAY (7LMCPILOT,2,0,6HRECALL)
    3 CALL OVERLAY (7LMCPILOT,3,0,6HRECALL)
    4 CALL OVERLAY (7LMCPILOT,4,0,6HRECALL)
    5 CALL OVERLAY (7LMCPILOT,5,0,6HRECALL)
      END
```

```
      SUBROUTINE CLASS(ICL)
C     CLASSIFICATION AT BOTTOM OF PAGE AND AT TOP OF NEXT PAGE
 8279 FORMAT(1H2,3(15X,*GARBAGE      *,15X)/3(15X,*GARBAGE      *,15X)/)
 8280 FORMAT('1')
 8281 FORMAT(1H2,3(15X,*CONFIDENTIAL*,15X)/3(15X,*CONFIDENTIAL*,15X)/)
 8282 FORMAT(1H2,3(15X,*SECRET       *,15X)/3(15X,*SECRET       *,15X)/)
 8283 FORMAT(1H2,3(15X,*TOP SECRET   *,15X)/3(15X,*TOP SECRET   *,15X)/)
 8284 FORMAT(1H2,3(15X,*PRIVATE      *,15X)/3(15X,*PRIVATE      *,15X)/)
 8285 FORMAT(1H2,3(15X,*PROPRIETARY *,15X)/3(15X,*PROPRIETARY *,15X)/)
      IF(ICL.EQ.(-1))WRITE(6,8279)
      IF(ICL.EQ.0)WRITE(6,8280)
      IF(ICL.EQ.1)WRITE(6,8281)
      IF(ICL.EQ.2)WRITE(6,8282)
      IF(ICL.EQ.3)WRITE(6,8283)
      IF(ICL.EQ.4)WRITE(6,8284)
      IF(ICL.EQ.5)WRITE(6,8285)
      RETURN
      ENTRY CLASST
C     CLASSIFICATION AT TOP OF PAGE ONLY
 7279 FORMAT('1',3(15X,'GARBAGE',15X)//)
 7281 FORMAT('1',3(15X,'CONFIDENTIAL',15X)//)
 7282 FORMAT('1',3(15X,'SECRET',15X)//)
 7283 FORMAT('1',3(15X,'TOP SECRET',15X)//)
 7284 FORMAT('1',3(15X,'PRIVATE',15X)//)
 7285 FORMAT('1',3(15X,'PROPRIETARY',15X)//)
      IF(ICL.EQ.(-1))WRITE(6,7279)
      IF(ICL.EQ.0)WRITE(6,8280)
      IF(ICL.EQ.1)WRITE(6,7281)
      IF(ICL.EQ.2)WRITE(6,7282)
      IF(ICL.EQ.3)WRITE(6,7283)
      IF(ICL.EQ.4)WRITE(6,7284)
      IF(ICL.EQ.5)WRITE(6,7285)
      RETURN
      ENTRY CLASSB
C     CLASSIFICATION AT BOTTOM OF PAGE ONLY
 9279 FORMAT(1H2,3(15X,*GARBAGE      *,15X))
 9281 FORMAT(1H2,3(15X,*CONFIDENTIAL*,15X))
 9282 FORMAT(1H2,3(15X,*SECRET       *,15X))
 9283 FORMAT(1H2,3(15X,*TOP SECRET   *,15X))
 9284 FORMAT(1H2,3(15X,*PRIVATE      *,15X))
 9285 FORMAT(1H2,3(15X,*PROPRIETARY *,15X))
      IF(ICL.EQ.(-1))WRITE(6,9279)
      IF(ICL.EQ.0)RETURN
      IF(ICL.EQ.1)WRITE(6,9281)
      IF(ICL.EQ.2)WRITE(6,9282)
      IF(ICL.EQ.3)WRITE(6,9283)
      IF(ICL.EQ.4)WRITE(6,9284)
      IF(ICL.EQ.5)WRITE(6,9285)
      RETURN
      END
```

```
 BLOCK DATA
 COMMON / 4 / LCOUNT,NAME(200),LOC(200)
 DATA LCOUNT / 116 /
 DATA (NAME(I),I=1,46) /
*6HAORDER,6HPORDER,6HMODDOR,6HMCDNOR,6HBNDY  ,6HNIN   ,6HNMOD  ,
*6HNOUT  ,6HNTYPE ,6HNLMEQ ,6HINDTRO,6HHMAX  ,6HISETOL,6HISETRM,
*6HKPARAM,6HCNSTMN,6HDIRECT,6HLOCATE,6HCNSTMX,6HINDCLS,6HINDGST,
*6HINDMOD,6HAREFF ,6HCHRDF ,6HWTR7P ,6HAIY7F ,6HSFANF ,6HCUTOFF,
*6HMSKIP ,6HINDAIR,6HINDBAF,6HINDTST,6HDEL   ,6HRHC   ,6HV     ,
*6HALT   ,6HGAMA  ,6HAA    ,6HJUMP  ,6HPLTPI ,6HTITLE ,6HTOLISE,
*6HWEIGHT,6HBINPUT,6HPOLYN ,6HPOLYD /
 DATA (LOC(I) ,I=1,46) /
*1       ,145     ,289     ,297     ,449     ,465     ,473     ,
*481     ,497     ,505     ,506     ,507     ,509     ,510     ,
*515     ,704     ,709     ,714     ,719     ,725     ,726     ,
*727     ,728     ,729     ,730     ,731     ,732     ,734     ,
*740     ,743     ,744     ,746     ,749     ,754     ,755     ,
*756     ,757     ,758     ,779     ,780     ,795     ,805     ,
*807     ,1010    ,1300    ,1340    /
 DATA (NAME(I),I=47,64)  /
*6HCXA   ,6HCXAD  ,6HCXQ   ,6HCXU   ,6HCXH   ,6HCXD   ,
*6HCZA   ,6HCZAD  ,6HCZQ   ,6HCZU   ,6HCZH   ,6HCZD   ,
*6HCMA   ,6HCMAD  ,6HCMQ   ,6HCMU   ,6HCMH   ,6HCMD   /
 DATA (LOC( I) ,I=47,64)  /
*1400    ,1401    ,1402    ,1403    ,1404    ,1405    ,
*1406    ,1407    ,1408    ,1409    ,1410    ,1411    ,
*1412    ,1413    ,1414    ,1415    ,1416    ,1417    /
 DATA (NAME(I),I=65,99)  /
*6HCYB   ,6HCYBD  ,6HCYP   ,6HCYR   ,6HCYDR  ,6HCYD1  ,
*6HCLB   ,6HCLBD  ,6HCLP   ,6HCLR   ,6HCLDR  ,6HCLD1  ,
*6HCNB   ,6HCNBD  ,6HCNP   ,6HCNR   ,6HCNDR  ,6HCND1  ,
*6HCYD2  ,6HCYD3  ,6HDA    ,6HD1    ,6HD2    ,6HD3    ,
*6HCLD2  ,6HCLD3  ,6HTA    ,6HTB    ,6HTC    ,6HCND2  ,
*6HCND3  ,6HZIXB  ,6HZIZB  ,6HZIXZB ,6HALFAI /
 DATA (LOC( I) ,I=65,99)  /
*1425    ,1426    ,1427    ,1428    ,1429    ,1430    ,
*1431    ,1432    ,1433    ,1434    ,1435    ,1436    ,
*1437    ,1438    ,1439    ,1440    ,1441    ,1442    ,
*1443    ,1444    ,1445    ,1446    ,1447    ,1448    ,
*1449    ,1450    ,1451    ,1452    ,1453    ,1454    ,
*1455    ,1456    ,1457    ,1458    ,1459    /
 DATA (NAME(I),I=100,116)/
*6HAINPUT,6HRTIMES,6HRINPUT,6HOMEGAS,6HKTIME ,6HNISEAP,
*6HISEAUX,6HNTIME ,6HNFREQ ,6HICOST ,6HNCASE ,6HKGAIN ,
*6HJLOCUS,6HDELT  ,6HTOL   ,6HICRCSS,6HMOI   /
 DATA (LOC( I) ,I=100,116)/
*1711    ,4176    ,4236    ,4296    ,4318    ,4320    ,
*4321    ,4329    ,4330    ,4331    ,4344    ,4350    ,
*4354    ,4258    ,4362    ,4366    ,4379    /
 END
```

```
      BLOCK DATA CATAB
      COMMON / 3 / MAXT,STABLE(2),LOCS(2)
      DATA MAXT / 2 /
      DATA (STABLE(I),I=1,2) /
     *6HAMATX ,6HBMATX /
      DATA (LOCS(I),I=1,2) /
     *1,1600 /
      END
```

```
FUNCTION NEW(I,J,K,L)
DATA IDIM,JDIM,KDIM / 12,12,11 /
NEW = I+IDIM*(J-1+JDIM*(K-1+KDIM*(L-1)))
RETURN
END
```

```
      SUBROUTINE BLANK
C
C      THIS SUBROUTINE IS FILLED IN BY USER, WHO MAKES THE A AND/OR B
C           MATRICES FUNCTIONS OF THE P(*)'S
C      SIMULTANEOUSLY THE USER MAKES THE AORDER AND BORDER MATRICES
C         COMPATIBLE WITH THE A AND B MATRICES RESPECTIVELY
C
C     BLANK COMPUTES THOSE ELEMENTS OF THE A MATRIX THAT
C     DEPEND ON THE PILOT PARAMETERS, P(I).   SKBJI COMPUTES
C     THE DETERMINANT OF THE A MATRIX (AS A HUMONGOUS POLYNOMIAL
C     IN S).   POLYRT FINDS ROOTS OF POLYNOMIALS.
C
      COMMON COMON(4400)
      COMMON / 3 / A(12,12,11),DUM(15),B(12,12,11),DUM1(16)
      INTEGER AORDER(12,12),BORDER(12,12)
      DIMENSION P(5),ACOST(8)
      EQUIVALENCE (AORDER,COMON(    1))
      EQUIVALENCE (BORDER,COMON( 145))
      EQUIVALENCE (KNTISE,COMON( 514))
      EQUIVALENCE (P     ,COMON( 780))
      EQUIVALENCE (PR    ,COMON(4345))
      EQUIVALENCE (ACOST ,COMON( 695))
      ENTRY BLANKI
C     USER SUPPLIED BLANK INPUT FOLLOWS THIS CARD
      IF(KNTISE.NE.0)GO TO 30
      DO 20 I=1,12
      AORDER(1,I)=-1
      AORDER(I,1)=-1
      AORDER(4,I)=-1
 20   AORDER(I,4)=-1
      AORDER(1,1)=0
      AORDER(4,4)=0
      A(1,1,1)=1.
      A(4,4,1)=1.
      AORDER(6,8)=2
      AORDER(6,6)=2
      TAU=.2
 30   A(6,8,1)=-2.*P(1)
      A(6,8,2)=P(1)*(TAU-2.*P(2))
      A(6,8,3)=P(1)*TAU*P(2)
      A(6,6,1)=2.
      A(6,6,2)=TAU+2.*P(3)
      A(6,6,3)=TAU*P(3)
      RETURN
      ENTRY COSTI
C     USER SUPPLIED COST FUNCTION EQUATIONS FOLLOW THIS CARD, IF DESIRED
      RETURN
      ENTRY PRATEI
C     USER SUPPLIED PILOT RATING EQUATION FOLLOWS THIS CARD, IF DESIRED
      SIGT=ACOST(1)**.5
      SIGNZ=ACOST(2)**.5
      PHI1=ATAN(P(2))-ATAN(P(3))
```

```
PHI1=ABS(PHI1)
IF(P(1).EQ.0.)PHI1=0.
PR=3.9*SIGT+.03*SIGNZ+8.6/SIGNZ+1.6*PHI1
RETURN
END
```

```
      OVERLAY(MCPILOT,1,0)
      PROGRAM INPUT
      COMMON COMON(4400)
      COMMON / 5 / AA(1599),BB(1601)
      EQUIVALENCE (AORDER,COMON(   1))
      EQUIVALENCE (BORDER,COMON( 145))
      EQUIVALENCE (MODDOR,COMON( 289))
      EQUIVALENCE (MODNOR,COMON( 297))
      EQUIVALENCE (BNDY  ,COMON( 449))
      EQUIVALENCE (NSYST ,COMON( 489))
      EQUIVALENCE (HMAX  ,COMON( 507))
      EQUIVALENCE (MODELD,COMON( 519))
      EQUIVALENCE (MODELN,COMON( 607))
      EQUIVALENCE (INCAIR,COMON( 743))
      EQUIVALENCE (ILINK ,COMON( 748))
      EQUIVALENCE (BINPUT,COMON(1010))
      EQUIVALENCE (AINPUT,COMON(1711))
      EQUIVALENCE (RTIMES,COMON(4176))
      EQUIVALENCE (RINPUT,COMON(4236))
      EQUIVALENCE (OMEGAS,COMON(4296))
      EQUIVALENCE (NTIME ,COMON(4329))
      EQUIVALENCE (NFREQ ,COMON(4330))
      EQUIVALENCE (NCASE ,COMON(4344))
      EQUIVALENCE (KGAIN ,COMON(4350))
      EQUIVALENCE (DELT  ,COMON(4358))
      EQUIVALENCE (TOL   ,COMON(4362))
      EQUIVALENCE (ICROSS,COMON(4366))
      EQUIVALENCE (ACOST ,COMON(4371))
      EQUIVALENCE (B,BB),(A,AA)
      INTEGER BORDER
      DIMENSION B(12,12,11),BORDER(12,12),BINPUT(1)
      DIMENSION MODNOR(8),MODDOR(8),MODELN(8,11),MODELD(8,11)
      DIMENSION A(12,12,11),AORDER(12,12),AINPUT(1)
      DIMENSION OMEGAS(21),RTIMES(60),RINPUT(60)
      DIMENSION BNDY(16),ICROSS(4),TOL(4),DELT(4),KGAIN(4)
      DIMENSION NSYST(8)
      DIMENSION ACOST(8)
      REAL MODELN,MODELD
      INTEGER AORDER
      DO 14 I=1,12
      DO 14 J=1,12
      AORDER(I,J) = -1
   14 BORDER(I,J) = -1
      DO 9 N=1,8
      MODNOR(N)=-1
      ACOST(N)=0.
      NSYST(N)=1
    9 MODDOR(N)=-1
      INPERR = 0
      OMEGAS( 1)=.01
      OMEGAS( 2)=.05
      OMEGAS( 3)=.1
```

```
      OMEGAS( 4)=.15
      OMEGAS( 5)=.2
      OMEGAS( 6)=.3
      OMEGAS( 7)=.5
      OMEGAS( 8)=.7
      OMEGAS( 9)=1.
      OMEGAS(10)=1.5
      OMEGAS(11)=2.
      OMEGAS(12)=2.5
      OMEGAS(13)=3.
      OMEGAS(14)=3.5
      OMEGAS(15)=4.
      OMEGAS(16)=4.5
      OMEGAS(17)=5.
      OMEGAS(18)=5.5
      OMEGAS(19)=6.
      OMEGAS(20)=8.
      OMEGAS(21)=10.
      NFREQ=21
      HMAX=.1
      NTIME=2
      RTIMES(1)=0.
      RTIMES(2)=3.
      RTIMES(3)=5.
      RINPUT(1)=1.
      RINPUT(2)=1.
      RINPUT(3)=1.
      DO 10 N=1,4
      KGAIN(N)=2
      TOL(N)=.0001
      DELT(N)=.125
      BNDY(N*4-3)=-12.
      BNDY(N*4-2)=4.
      BNDY(N*4-1)=-2.
      BNDY(N*4)=8.
10    CONTINUE
      NCASE=1
      CALL READA(INPERR)
      IF (INPERR .EQ. 0) GO TO 15
      WRITE(6,16) INPERR
   16 FORMAT(1H0,10HTHERE WERE ,I4,26H ERRORS IN THE INPUT DATA.)
      STOP
   15 CONTINUE
      IIII = 1
      COMON(4317)=COMON(725)
      DO 5 J=1,12
      DO 5 I=1,12
      IF (BORDER(I,J).EQ.-1) GO TO 5
      MM = BORDER(I,J) + 1
      DO 6 K=1,MM
      B(I,J,K) = BINPUT(IIII)
    6 IIII = IIII + 1
```

```
    5 CONTINUE
    7 CONTINUE
      IIII = 1
      DO 2 J=1,12
      DO 2 I=1,12
      IF( AORDER(I,J) .EQ. -1 ) GO TO 2
      MM = AORDER(I,J) + 1
      DO 1 K=1,MM
      A(I,J,K) = AINPUT(IIII)
    1 IIII = IIII + 1
    2 CONTINUE
      II=1300
      DO 100 I=1,8
      IF(MODNOR(I).EQ.-1)GO TO 100
      NC=MODNOR(I)+1
      DO 101 J=1,NC
      MODELN(I,J)=COMON(II)
  101 II=II+1
  100 CONTINUE
      II=1340
      DO 200 I=1,8
      IF(MODDOR(I).EQ.-1)GO TO 202
      NC=MODDOR(I)+1
      DO 201 J=1,NC
      MODELD(I,J)=COMON(II)
  201 II=II+1
  200 CONTINUE
      GO TO 3
  202 MODDOR(I)=0
      MODELD(I,1)=1.
      GO TO 200
    3 CONTINUE
      IF( INDAIR .EQ. 0) GO TO 4
C     CALL MISER PROGRAM
      ILINK = 3
      GO TO 999
    4 ILINK = 2
C     CALL AERO PROGRAM
  999 CALL OVRLAY
C     NUMEQ IS THE NUMBER OF EQUATIONS IN THE SYSTEM--MAXIMUM 12
C     SYSTEMS ARE OF THE FORM A*X = B*U
C     A(I,J,K) IS A SYSTEM MATRIX WHERE I=ROW--MAXIMUM 12
C                                       J=COLUMN--MAXIMUM 12
C                                       K=COEFFICIENT OF S**(K-1)--
C                                                MAXIMUM 11
C     AORDER(I,J) TELLS ORDER OF A(I,J,*) POLYNOMIAL ELEMENT
C          LET AORDER(I,J)=(-1) IF A(I,J,K)=0 FOR ALL K
C     B(I,J,K) IS B SYSTEM MATRIX WHERE I,J,K ARE DEFINED AS ABOVE
C     BORDER(I,J) TELLS ORDER OF B(I,J,*) (ANALOGOUS TO AORDER)
C     MODELN(N,K) IS MODEL NUMERATOR WHERE K IS DEFINED AS ABOVE AND N
C                 IS THE MODEL NUMBER--MAXIMUM=8
C     MODELD(N,K) IS MODEL DENOMINATOR WHERE K AND N ARE DEFINED AS ABOV
```

```
C      MODNOR(N) TELLS ORDER OF MODELN(N,*)
C          LET MODNOR(N)=(K-1) FOR MODELN(N,K) UNLESS MODELN(N,K) IS ZERO
C                  LET MODNOR(N)=-1     IF  MODELN(N,K) IS ZERO
C      MODDOR(N) TELLS ORDER OF MODELD(N,*)
C          LET MODDOR(N)=(K-1) FOR MODELD(N,K) UNLESS MODELD(N,K) IS ZERO
C                  LET MODDOR(N)=-1     IF  MODELD(N,K) IS ZERO
C      NNN IS NUMBER OF FACTORS FOR THE I,J,*,L ELEMENT--MAXIMUM=10
C      NN(N) IS THE ORDER OF THE NTH FACTOR--MAXIMUM ORDER=4
C      TEMP(N,M) IS THE COEFFICIENT OF S**M OF THE NTH FACTOR
C      ISETRM IS THE TOTAL NUMBER OF ISE TERMS--MAXIMUM=8
C      KPARAM IS THE NUMBER OF PARAMETERS TO BE VARIED--MAXIMUM=15
C      NIN(*),NMOD(*),NOUT(*),NSYST(*),NTYPE(*), AND WEIGHT(*) MUST BE
C            SPECIFIED FOR EACH ISE TERM USED
C      NIN(N)    TELLS WHICH INPUT WILL BE USED IN THE NTH ISE TERM TO
C            DETERMINE THE SYSTEM TRANSFER FUNCTION
C      NMOD(N)   TELLS WHICH MODEL  WILL BE USED IN THE NTH ISE TERM
C      NOUT(N)  TELLS WHICH OUTPUT WILL BE USED IN THE NTH ISE TERM TO
C            DETERMINE THE SYSTEM TRANSFER FUNCTION
C      NSYST(N)  TELLS WHICH SYSTEM WILL BE USED IN THE NTH ISE TERM
C      NTYPE(N)  TELLS WHAT TYPE OF INPUT NIN(N) IS FOR THE NTH ISE TERM
C      WEIGHT(N) TELLS WHAT WEIGHT THE NTH ISE TERM WILL HAVE WITH
C               RESPECT TO THE OTHER ISE TERMS
C      ISETOL DECIDES WHAT STOPS THE PROGRAM
C      LET ISETOL = -1 IF AND ONLY IF PROGRAM IS TO STOP WHEN LATEST ISE
C              IS WITHIN TOLISE TOLERANCE OF PREVIOUS ISE
C      LET ISETOL =  0 IF AND ONLY IF PROGRAM IS TO STOP WHEN LATEST P(I)
C              IS WITHIN CUTOFF(I) TOLERANCE OF PREVIOUS P(I) FOR ALL I
C      LET ISETOL = +1 IF AND ONLY IF PROGRAM IS TO STOP WHEN BOTH LATEST
C           ISE IS WITHIN TOLISE TOLERANCE OF PREVIOUS ISE AND LATEST P(I)
C           IS WITHIN CUTOFF(I) TOLERANCE OF PREVIOUS P(I) FOR ALL I
C      TOLISE IS THE TOLERANCE ON ISE--NEEDED WHEN ISETOL = -1 OR +1
C      JUMP IS THE FIRST STEP SIZE USED IN DETERMINING THE END POINTS FOR
C         THE INTERVAL IN MINIMI. JUMP IS UPDATED BY MINIMI EACH TIME
C         MINIMI IS CALLED. IT IS PRINTED SO THAT IN FUTURE RUNS OF
C                   THE SAME CASE ONE CAN BETTER ESTIMATE JUMP.
C      CNSTMN(I) IS THE MINIMUM CONSTRAINT ON P(I)
C      CNSTMX(I) IS THE MAXIMUM CONSTRAINT ON P(I)
C      CUTOFF(I) IS THE TOLERANCE ON P(I)--NEEDED WHEN ISETOL = 0 OR +1
C      DEL(I)   IS THE DISTANCE P WILL BE INCREMENTED IN THE ITH DIMENSION
C                     TO CALCULATE DIRCOS(I) IN THE ITH DIMENSION IN GRADI
       END
```

```
      SUBROUTINE DIPLAC(RA1,INC,BLANK)
      DIMENSION RA1(6)
      DIMENSION RC(2)
C     ****************************************************************
CPICK UP COLUMN 67 TO 72 AND
CRIGHT JUSTIFY FOR USE IN THE INPUT ROUTINE
C
C
C
C     ****************************************************************
      DO 7 I=1,6
      IF (RA1(6) .NE. BLANK) GO TO 2
     IF (I.EQ.6) GO TO 7
     DO 1 N=1,5
      N1=7-N
      N2=6-N
 1    RA1(N1)=RA1(N2)
 7    RA1(1)=BLANK
      INC=0
C
C
C    IF EXIT THROUGH GO STATEMENT 2
C    THEN A VALID NUMBER EXIST.
C
      GO TO 6
 2    CALL PACKR (RA1,RC    ,6)
      CALL READ31(3,RC,INC,1)
 6    RETURN
     END
```

```
      SUBROUTINE READA(INPERR)
      DIMENSION FI(56)
      DIMENSION MSG(58) , RA(55) ,FO(2,28)
       DIMENSION RAI(6)
      EQUIVALENCE (MSG(1),SYM) , (MSG( 2),OP )
      EQUIVALENCE 'MSG(3),RA ) , (MSG(58),INC)
C
      DATA SLTSYM / 6H        /
      DATA BCD,AINT,OCT,TRA,COMMA,BLANK,POINT,EE /
     1     3HBCD,3HINT,3HOCT,3HTRA,1H,,1H ,1H.,1HE /
C     CARD NO. 251 DELETED
      DATA STP,BADSYM /6HSTOP  ,6H      /
 4     FORMAT(A6,1X,A3,1X,55A1,6A1)
 10   FORMAT (12X,A6,1X,A3,1X,55A1,I6)
 14   FORMAT(18X,A6,1X,A3,1X,55A1,I6)
 100   READ(5,4)SYM,OP,RA,RAI
      IF(SYM.NE.STP) GO TO 103
      STOP
  103 CONTINUE
       CALL DIPLAC(RAI,INC,BLANK)
      J = 58
      IF (MSG(58) .LE. 0) J=57
      WRITE(6,14)  (MSG(I),I=1,J)
      IF(OP .NE. TRA) GO TO 212
      IID=-1
      RETURN
 212  IF (RA(1) .EQ. BLANK) GO TO 803
      IF (SYM.EQ.BADSYM) GO TO 100
      IF (SYM    .EQ. SLTSYM) GO TO 215
      CALL DSERCH (SYM,IDD,ITABLE)
      IF(ITABLE .EQ. 1) GO TO 900
C
C     DECODE CARD SECTION
C
 215  IF (OP .NE. BCD) GO TO 300
CBCD DATA
      CALL       PACBCD (RA,FI,JJ)
      GO TO 500
CNUMERIC TYPE
 300  IFI = 0
COCTAL TYPE
      IF (OP .EQ. OCT) IFI=1
CINTEGER TYPE
      IF (OP .EQ. AINT) IFI=4
      IQ=0
      LQ=0
      JJ=1
 310  KQ=LQ+1
 313  LQ=LQ+1
      IF (LQ .GT. 55) GO TO 318
      IF (RA(LQ) .EQ. COMMA) GO TO 320
      IF (RA(LQ) .NE. BLANK) GO TO 315
```

```
318   IQ=1
320   MQ=LQ-1
      IF (IFI .NE. 0) GO TO 400
325   JQ=KQ-1
330   JQ=JQ+1
331   IF (RA(JQ) .EQ. POINT) GO TO 335
      IF (RA(JQ) .NE. EE    ) GO TO 340
335   IFI=2
      GO TO 400
340   IF (JQ-MQ) 330,345,345
345   IFI=3
400   CALL PACKR (RA(KQ) , FO(1,JJ) , MQ-KQ+1 )
      IF (IQ .NE. 0) GO TO 425
420   JJ=JJ+1
      GO TO 310
425   CALL READ31 (IFI, FO , FI , JJ )
500   MIORO=0
      IF(INC .GT. 0) MIORC=-1
      IIC=IOD+INC+MIORC
      CALL  SVI (ITABLE,JJ,IID,FI)
 501  CONTINUE
      SLTSYM=SYM
      GO TO 100
 803  CONTINUE
      WRITE (6,83)
83    FORMAT (26H ERROR. COLUMN 12 IS BLANK)
 900  INPERR = INPERR+1
      BACSYM = SYM
      GO TO 100
      END
```

```
      SUBROUTINE PACBCD (RA,FI,JJ)
      DIMENSION RA(55),FI(56)
      DECODE (1,5,RA) JJ
5     FORMAT(I1)
       K1=2
       KK=7
       DO 10 I=1,JJ
       ENCODE (10,1,FI(I)) (RA(L), L=K1,KK)
       K1=K1+6
10     KK=KK+6
1     FORMAT(6A1)
      RETURN
      END
```

```
      SUBROUTINE PACKR (I1 , I2 , N )
      DIMENSION I1(1)
1     FORMAT (1H(,I2,2HX,,I2,3HA1))
      K = 20-N
      ENCODE (10,1,XMAT) K , N
      ENCODE (20,XMAT,I2) (I1(I),I=1,N)
      RETURN
      END
```

```
      SUBROUTINE READ31 ( IFI , FJ , FI , JJ )
      DIMENSION FI(56) , FJ(56)
      GO TO (100,200,300,400) , IFI
100   II=1
      DO 101 I=1,JJ
      DECODE (20,1,FJ(II)) FI(I)
      II=II+2
101   CONTINUE
1     FORMAT (   O20)
      RETURN
200   II=1
      DO 201 I=1,JJ
      DECODE(20,2,FJ(II)) FI(I)
      II=II+2
201    CONTINUE
2     FORMAT (   E20.0)
      RETURN
300   II=1
       J1=JJ
       JJ=1
      DECODE (20, 5 , FJ ) (FI(I),I=1 ,JJ)
       JJ=J1
      IF( JJ .EQ. 1) RETURN
      II=II+2
      DO 301 I=2,JJ
      DECODE(20,2,FJ(II)) FI(I)
       II=II+2
301   CONTINUE
5     FORMAT (10X,I10/ (E20.0) )
      RETURN
400   II=1
      DO 401 I=1,JJ
      DECODE (20,4,FJ(II)) FI(I)
      II=II+2
401   CONTINUE
4     FORMAT (   I20)
      RETURN
      END
```

```
      SUBROUTINE DSERCH(SYM,LOC,ICOM)
      COMMON COMON(4400)
      EQUIVALENCE (DIRECT,COMON( 709))
      EQUIVALENCE (LOCATE,COMON( 714))
      COMMON / 3 / MAXT,STABLE(2),LOCS(2)
      COMMON / 4 / LCOUNT,ANAME(200),LOCF(200)
      DIMENSION DIRECT(5),LOCATE(5)
      DO 50 I=1,LCOUNT
      IF (SYM .NE. ANAME(I) ) GO TO 50
      ICOM=0
      LOC=LOCF(I)
      RETURN
50    CONTINUE
      DO 100 I=1,MAXT
      IF(SYM .NE. STABLE(I)) GO TO 100
      ICOM=-1
      LOC=LOCS(I)
      RETURN
100   CONTINUE
      DO 200 I=1,5
      IF(SYM.NE. DIRECT(I)) GO TO 200
      ICOM = 0
      LOC=LOCATE(I)
      RETURN
 200  CONTINUE
      ICOM=1
      LOC = 0
      WRITE (6,6) SYM
      RETURN
 6    FORMAT( 2HC-, A10, 19H-  NOT IN DIRECTORY)
      END
```

```
      SUBROUTINE SVI(ICOM,N,ISTART,A)
      COMMON DATA(44CC)
      COMMON / 5 / TABLE(1)
      DIMENSION A(1)
      J=ISTART
      K = IABS(ICO 1)+1
      GO TO (1C ,20 ,30 ),K
10    DO 100 I=1,N
      DATA(J)=A(I)
100   J =J+1
      RETURN
20    DO 200 I =1,N
      TABLE(J)=A(I)
200   J =J+1
      RETURN
   30 CONTINUE
      RETURN
      END
```

```
      OVERLAY(MCPILOT,2,0)
      PROGRAM AERO
C     KC7S - ROOTS OF A/C LONGITUDINAL TRANSFER FUNCTIONS
C
      COMMON / 5 / A(3200)
      COMMON COMON(4400)
      EQUIVALENCE (INDCLS,COMON( 725))
      EQUIVALENCE (INDAIR,COMON( 743))
      EQUIVALENCE (INDGST,COMON( 726))
      EQUIVALENCE (INDMOD,COMON( 727))
      EQUIVALENCE (ILINK ,COMON( 748))
      EQUIVALENCE (TITLE ,COMON( 795))
      DIMENSION TITLE(10)
C
C
      IF(INDAIR.NE.0)GO TO 999
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
 6000 FORMAT(1H ,20X,10A6)
      IF(INDMOD.NE.0)GO TO 100
      CALL IVXIV
 50   CONTINUE
      IF(INDGST.GT.0)CALL GUST
      GO TO 999
  100 CONTINUE
      CALL IIIX3
      GO TO 50
  999 CONTINUE
      ILINK = 3
      CALL OVRLAY
      END
```

OB-OUTENE IVXIV

```
C
      COMMON COMON(4400)
      COMMON / 5 / A(12,12,11)
      EQUIVALENCE (AORDER,COMON(   1))
      EQUIVALENCE (RHO   ,COMON( 754))
      EQUIVALENCE (V     ,COMON( 755))
      EQUIVALENCE (S     ,COMON( 728))
      EQUIVALENCE (W     ,COMON( 730))
      EQUIVALENCE (CBAR  ,COMON( 729))
      EQUIVALENCE (IY    ,COMON( 731))
      EQUIVALENCE (CXA   ,COMON(1400))
      EQUIVALENCE (CXAD  ,COMON(1401))
      EQUIVALENCE (CXQ   ,COMON(1402))
      EQUIVALENCE (CXU   ,COMON(1403))
      EQUIVALENCE (CXH   ,COMON(1404))
      EQUIVALENCE (CXD   ,COMON(1405))
      EQUIVALENCE (CZA   ,COMON(1406))
      EQUIVALENCE (CZAD  ,COMON(1407))
      EQUIVALENCE (CZQ   ,COMON(1408))
      EQUIVALENCE (CZU   ,COMON(1409))
      EQUIVALENCE (CZH   ,COMON(1410))
      EQUIVALENCE (CZD   ,COMON(1411))
      EQUIVALENCE (CMA   ,COMON(1412))
      EQUIVALENCE (CMAD  ,COMON(1413))
      EQUIVALENCE (CMQ   ,COMON(1414))
      EQUIVALENCE (CMU   ,COMON(1415))
      EQUIVALENCE (CMH   ,COMON(1416))
      EQUIVALENCE (CMD   ,COMON(1417))
      EQUIVALENCE (ALT   ,COMON( 756))
      EQUIVALENCE (GAMA  ,COMON( 757))
      EQUIVALENCE (AA    ,COMON( 758))
      EQUIVALENCE (AX    ,COMON( 759))
      EQUIVALENCE (XH    ,COMON(1462))
      EQUIVALENCE (XA    ,COMON(1463))
      EQUIVALENCE (XAD   ,COMON(1464))
      EQUIVALENCE (XQ    ,COMON(1465))
      EQUIVALENCE (XD    ,COMON(1466))
      EQUIVALENCE (XU    ,COMON(1467))
      EQUIVALENCE (ZH    ,COMON(1468))
      EQUIVALENCE (ZA    ,COMON(1469))
      EQUIVALENCE (ZAD   ,COMON(1470))
      EQUIVALENCE (ZQ    ,COMON(1471))
      EQUIVALENCE (ZD    ,COMON(1472))
      EQUIVALENCE (ZU    ,COMON(1473))
      EQUIVALENCE (MH    ,COMON(1474))
      EQUIVALENCE (MA    ,COMON(1475))
      EQUIVALENCE (MAD   ,COMON(1476))
      EQUIVALENCE (MQ    ,COMON(1477))
      EQUIVALENCE (MD    ,COMON(1478))
      EQUIVALENCE (MU    ,COMON(1479))
      EQUIVALENCE (SINA  ,COMON(1460))
```

```
      EQUIVALENCE (COSA  ,COMON(1461))
      EQUIVALENCE (INDCLS,COMON( 725))
      EQUIVALENCE (INDRAF,COMON( 744))
      EQUIVALENCE (INDGST,COMON( 726))
      EQUIVALENCE (INDMOD,COMON( 727))
      INTEGER AORDER(12,12)
      REAL MH,MA,MAD,MQ,MD,MU,MW,LX,IY,MASS,
     1 MUS,NZA
      DATA DTR/57.29578/,G/32.174/
C
C     THE CORE LOCATIONS USED FOR THE AERODYNAMIC
C     COEFFICIENTS ARE ALSO IN THE MIDDLE OF ASKBJI,
C     A WORKING ARRAY IN THE MAIN MISER PROGRAM.  THIS
C     IS ALSO THE TEMP ARRAY OF SUBROUTINE SKBJI.
 15   WRITE(6,4100)
 20   WRITE(6,4500)CXA,CXAD,CXQ,CXD,CXU,CXH,CZA,CZAD,CZQ,CZD,
     1 CZU,CZH,CMA,CMAD,CMQ,CMD,CMU,CMH,V,S,IY,AA,ALT,GAMA,
     2 RHO,W,CBAR
 25   CZERO=CXU+CXH+CZU+CZH+CMU+CMH
      IF(CZERO.EQ.0.)GO TO 57
      MASS=W/G
      RVS=RHO*V*S
      RVSC=RVS*CBAR
      Q=.5*RHO*V**2
C
C      CALCULATE DIMENSIONAL DERIVATIVES UNPRIMED
C
      MU=RVSC/IY*CMU
      MH=RVSC/(2.*IY)*CMH
      MA=RVSC*V/(2.*IY)*CMA
      MAD=RVSC*CBAR/(4.*IY)*CMAD
      MQ=RVSC*CBAR/(4.*IY)*CMQ
      MD=RVSC*V/(2.*IY)*CMD
      XU=RVS/MASS*CXU
      XH=RVS/(2.*MASS)*CXH
      XA=RVS*V/(2.*MASS)*CXA
      XAD=RVSC/(4.*MASS)*CXAD
      XQ=RVSC/(4.*MASS)*CXQ
      XD=RVS*V/(2.*MASS)*CXD
      ZU=RVS/MASS*CZU
      ZH=RVS/(2.*MASS)*CZH
      ZA=RVS*V/(2.*MASS)*CZA
      ZAD=RVSC/(4.*MASS)*CZAD
      ZQ=RVSC/(4.*MASS)*CZQ
      ZD=RVS*V/(2.*MASS)*CZD
C
      AO=AX-AA
      SINA=SIN(AC/DTR)
      COSA=COS(AC/DTR)
C
      IF(AO.NE.0.)GO TO 40
C
```

```
      XW=XA/V
      ZW=ZA/V
      MW=MA/V
C
C     WRITE DIMENSIONAL DERIVATIVES UNPRIMED
C
      WRITE(6,5500)XA,XAD,XQ,XD,XU,XH,XW,ZA,ZAD,ZQ,ZD,ZU,ZH,ZW,
     1 MA,MAD,MQ,MD,MU,MH,MW
      GO TO 50
C
C     CALCULATE DIMENSIONAL DERIVATIVES PRIMED
C
 40   MUS=MU
      MU=MU*COSA-MA/V*SINA
      MH=MH*MU/MUS
      MA=MA*COSA+V*MUS*SINA
      MAD=MAD*COSA
      XUS=XU
      XAS=XA
      XADS=XAD
      XQS=XQ
      XDS=XD
      XU=XU*COSA**2+ZA/V*SINA**2-(XA/V+ZU)*SINA*COSA
      XH=XH*XU/XUS
      XA=XA*COSA**2-V*ZU*SINA**2+(V*XUS-ZA)*SINA*COSA
      XAD=XAD*COSA-ZAD*SINA
      XQ=XQ*COSA-ZQ*SINA
      XD=XD*COSA-ZD*SINA
      ZUS=ZU
      ZU=ZU*COSA**2-XAS/V*SINA**2+(XUS-ZA/V)*SINA*COSA
      ZH=ZH*ZU/ZUS
      ZA=ZA*COSA**2+V*XUS*SINA**2+(XAS+V*ZUS)*SINA*COSA
      ZAD=ZAD*COSA+XADS*SINA
      ZQ=ZQ*COSA+XQS*SINA
      ZD=ZD*COSA+XDS*SINA
C
C     WRITE DIMENSIONAL DERIVATIVES PRIMED
C
      WRITE(6,5600)
      WRITE(6,6000)XA,XAD,XQ,XD,XU,XH,ZA,ZAD,ZQ,ZD,ZU,ZH,
     1 MA,MAD,MQ,MD,MU,MH
C
C     CALCULATE AND PRINT LA,DELTA/NZ, AND NZ/A
C
 50   CONTINUE
      IF(INDBAF.NE.0) GO TO 55
      DZERO=XD+ZD+MD
      IF(DZERO.EQ.0.0)GO TO 60
      TERM4=V/G*(MD*ZA-MA*ZD)
      TERM5=V*COSA+ZQ
      DNZ=(MA*TERM5-MQ*ZA)/TERM4
      NZA=TERM4/(-MD*TERM5+MQ*ZD)
```

```
      WRITE(6,7000)DNZ,NZA,Q
      GO TO 75
   55 CALL FILLIT(1)
      GO TO 999
 57   WRITE(6,1000)
 1000 FORMAT(6(/),10X,*SINCE CXL,CXH,CZU,CZH,CMU,CMH EQUAL ZERO, THIS IN
     1 TURN CAUSES A ZERO ROW OR COLUMN IN MATRIX - EXIT CALLED*)
      STOP
 60   ICHAR=1
      WRITE(6,7500)
 75   CONTINUE
C     STICK FIXED ANALYSIS
C
      CALL STFIX(NZA,DZERO)
C
 4100 FORMAT(T2,*INPUT DATA (NON-DIMENSIONAL) PER RADIAN    LONGITUDINAL
     1 MODE*,/T2,13(*---*),4X,17(*-*))
 4500 FORMAT(T2,*CXA =*,1PE11.4,4X,*CXAD =*,1PE11.4,4X,*CXQ =*,1PE11.4,
     14X,*CXD =*,1PE13.4,4X,*CXL =*,1PE11.4,4X,*CXH =*,1PE11.4/T2,*CZA =
     2*,1PE11.4,4X,*CZAD =*,1PE11.4,4X,*CZQ =*,1PE11.4,4X,*CZD =*,1PE13.
     34,4X,*CZU =*,1PE11.4,4X,*CZH =*,1PE11.4/T2,*CMA =*,1PE11.4,4X,
     4*CMAD =*,1PE11.4,4X,*CMQ =*,1PE11.4,4X,*CMD =*,1PE13.4,4X,*CMU =*,
     51PE11.4,4X,*CMH =*,1PE11.4/T2,*VEL.=*,1PE11.4,4X,*S     =*,1PE11.4,
     64X,*IY   =*,1PE11.4,4X,*ALFAA =*,1PE11.4,4X,*ALT =*,1PE11.4,4X,
     7*GAMA=*,1PE11.4/T2,*RHO =*,1PE11.4,4X,*GWT   =*,1PE11.4,4X,*MAC =*,
     81PE11.4)
 5500 FORMAT(/T2,*DIMENSIONAL STABILITY DERIVATIVES *,
     1 /T2,33(*-*),/T2,*XA =*,1PE11.4,4X,*XAD =*,1PE11.4,4X,*XQ =*,
     2 1PE11.4,4X,*XD =*,1PE11.4,4X,*XU =*,1PE11.4,4X,*XH =*,1PE11.4,4X,
     3 *XW =*,1PE11.4/T2,*ZA =*,1PE11.4,4X,*ZAD =*,1PE11.4,4X,*ZQ =*,
     4 1PE11.4,4X,*ZD =*,1PE11.4,4X,*ZU =*,1PE11.4,4X,*ZH =*,1PE11.4,4X,
     5 *ZW =*,1PE11.4/T2,*MA =*,1PE11.4,4X,*MAD =*,1PE11.4,4X,*MQ =*,
     6 1PE11.4,4X,*MD =*,1PE11.4,4X,*MU =*,1PE11.4,4X,*MH =*,1PE11.4,4X,
     7 *MW =*,1PE11.4)
 5600 FORMAT(/T2,*DIMENSIONAL STABILITY DERIVATIVES *,
     1 /T2,33(*-*))
 6000 FORMAT(T2,*XA =*,1PE12.5,4X,*XAD =*,1PE12.5,4X,*XQ =*,1PE12.5,4X
     1 *XD =*,1PE12.5,4X,*XL =*,1PE12.5,4X,*XH =*,1PE12.5/T2,*ZA =*,
     2 1PE12.5,4X,*ZAD =*,1PE12.5,4X,*ZQ =*,1PE12.5,4X,*ZD =*,1PE12.5,
     3 4X,*ZU =*,1PE12.5,4X,*ZH =*,1PE12.5/T2,*MA =*,1PE12.5,4X,*MAD =*,
     4 1PE12.5,4X,*MQ =*,1PE12.5,4X,*MD =*,1PE12.5,4X,*MU =*,1PE12.5,
     5 4X,*MH =*,1PE12.5)
 7000 FORMAT(*0*,T25,*DELTA/NZ =*,1PE11.4,T73,*NZ/A =*,1PE11.4,T112,
     1 *Q =*,1PE11.4/)
 7500 FORMAT(* *,T35,*DELTA/NZ AND NZ/A ARE NOT DEFINED FOR ZERO CONTROL
     1 DERIVATIVES.*)
 999  CONTINUE
      AORDER(1,5)=0
      AORDER(2,5)=0
      AORDER(3,5)=0
      A(1,5,1)=-XD
      A(2,5,1)=-ZD
```

```
A(3,5,1)=-MC
RETURN
END
```

```
      SUBROUTINE STFIX(NZA,DZERO)
C
C     STICK FIXED CALCULATION
C
C
      COMMON COMON(4400)
      EQUIVALENCE (ZA    ,COMON(1469))
      EQUIVALENCE (ZU    ,COMON(1473))
      EQUIVALENCE (MA    ,COMON(1475))
      EQUIVALENCE (MQ    ,COMON(1477))
      EQUIVALENCE (MU    ,COMON(1479))
      EQUIVALENCE (V     ,COMON( 755))
      EQUIVALENCE (INDCLS,COMON( 725))
C
      REAL MA,MQ,MU,LA,NZA,LAWN
      DIMENSION ROOTR(10),ROOTI(10)
C
C
C
      DIMENSION ROOTRD(10),ROOTID(10),DEN(81)
      DATA G/32.174/
      DATA TWOPI/6.2832/
C
C
C     COMPLETE EQUATIONS
C
C
C     EVALUATE DENOMINATOR
C
      CALL FILLIT(1)
      CALL DETVAL(N,DEN)
      LA=-ZA/V
      M=N+1
C
C     CALCULATE ROOTS
C
      CALL DMULR(DEN,N,ROOTRD,ROOTID)
      WRITE(6,1500)
      DO 10 I=1,N
      IF(ABS(ROOTID(I)).LT.1.E-4)GO TO 8
      WRITE(6,2000)ROOTRD(I),ROOTID(I)
      GO TO 9
 8    WRITE(6,2000)ROOTRD(I)
 9    ROOTR(I)=ROOTRD(I)
 10   ROOTI(I)=ROOTID(I)
      NCOM=0
      IF(ABS(ROOTI(1)).LT.0.0001)GO TO 12
      NCOM=NCOM+1
      J=1
      K=1
      GO TO 13
 12   IF(ABS(ROOTI(2)).LT.0.0001)GO TO 13
```

```
      NCOM=NCOM+1
      J=2
      K=2
      GO TO 14
 13   IF(ABS(ROOTI(3)).LT.C.0001)GO TO 14
      NCOM=NCOM+1
      J=3
      L=3
      GO TO 15
 14   IF(ABS(ROOTI(4)).LT.C.0001)GO TO 15
      NCOM=NCOM+1
      J=4
      L=4
 15   IF(NCOM-1)40,16,26
C
C     1 PAIR OF COMPLEX ROOTS
C
 16   WN=SQRT(ROOTR(J)**2+ROOTI(J)**2)
      WNC=WN/TWOPI
      Z=-ROOTR(J)/WN
      PD=TWOPI/ABS(ROOTI(J))
      P=TWOPI/WN
      T2=-.69315/ROOTR(J)
      T10=-2.30259/ROOTR(J)
      C2=T2/PD
      C10=T10/PD
      OC2=1./C2
      OC10=1./C10
      CON=ZA/V*MQ-MA
      WNSPB=SQRT(CON)
      WNPB=G/V*(MA*ZU-MU*ZA)/CON
      DSP=ABS(WNSPB-WN)
      DP=ABS(WNPB-WN)
      IF(DSP.GT.DP)GO TO 18
C
C     WRITE SHORT PERIOD
C
      LAWN=LA/WN
      WRITE(6,2500)Z,WN,WNC
      IF(DZERO.EQ.C.)GO TO 20
      WNZA=WN**2/NZA
      WRITE(6,3000)WNZA
      GO TO 20
C
C     WRITE LONG PERIOD
C
 18   WRITE(6,3500)Z,WN,WNC
 20   IF(T2.LT.0.0)GO TO 22
      WRITE(6,4000)P,T2,T10,PD,C2,C10,OC2,OC10
      GO TO 40
 22   WRITE(6,4500)P,T2,T10,PD,C2,C10,OC2,OC10
      GO TO 40
```

```
C
C      2 COMPLEX PAIRS
C
 26    IF(ABS(ROOTI(K)).GT.ABS(ROOTI(L)))GO TO 28
       I=L
       L=K
       K=I
C
C      CALCULATE SHORT PERIOD
C
 28    WNSP=SQRT(ROOTR(K)**2+ROOTI(K)**2)
       WNSPC=WNSP/TWOPI
       ZSP=-ROOTR(K)/WNSP
       PDSP=TWOPI/ABS(ROOTI(K))
       PSP=TWOPI/WNSP
       T2SP=-.69315/ROOTR(K)
       T10SP=-2.30259/ROOTR(K)
       C2SP=T2SP/PDSP
       C10SP=T10SP/PDSP
       OC2SP=1./C2SP
       OC10SP=1./C10SP
       LAWN=LA/WNSP
       WRITE(6,2500)ZSP,WNSP,WNSPC
       IF(DZERO.EQ.0.)GO TO 30
       WNZA=WNSP**2/NZA
       WRITE(6,3000)WNZA
 30    IF(T2SP.LT.0.0)GO TO 32
       WRITE(6,4000)PSP,T2SP,T10SP,PDSP,C2SP,C10SP,OC2SP,OC10SP
       GO TO 34
 32    WRITE(6,4500)PSP,T2SP,T10SP,PDSP,C2SP,C10SP,OC2SP,OC10SP
C
C      CALCULATE LONG PERIOD
C
 34    WNP=SQRT(ROOTR(L)**2+ROOTI(L)**2)
       WNPC=WNP/TWOPI
       ZP=-ROOTR(L)/WNP
       PDP=TWOPI/ABS(ROOTI(L))
       PP=TWOPI/WNP
       T2P=-.69315/ROOTR(L)
       T10P=-2.30259/ROOTR(L)
       C2P=T2P/PDP
       C10P=T10P/PDP
       OC2P=1./C2P
       OC10P=1./C10P
       WRITE(6,3500)ZP,WNP,WNPC
       IF(T2P.LT.0.)GO TO 36
       WRITE(6,4000)PP,T2P,T10P,PDP,C2P,C10P,OC2P,OC10P
       GO TO 40
 36    WRITE(6,4500)PP,T2P,T10P,PDP,C2P,C10P,OC2P,OC10P
C
C      WRITE COEFFICIENTS
C
```

```
40    WRITE(6,5000)(DEN(I),I=1,M)
1500  FORMAT(T11,*ROOTS (COMPLEX FORM)*)
2000  FORMAT(T5,E11.4,10X,E11.4)
2500  FORMAT(/T11,*SHORT PERIOD MODE*,/T5,*ZETA =*,E12.5,T54,*WN =*,
     1 E12.5,1X,*RAD/SEC*/T57,*=*,E12.5,1X,*CYCLES/SEC*)
3000  FORMAT(*+*,T102,*(WN)SQUARED/NZA =*,E12.5)
3500  FORMAT(/T11,*LONG PERIOD MODE*,/T5,*ZETA =*,E12.5,T54,*WN =*,
     1 E12.5,1X,*RAD/SEC*,/T57,*=*,E12.5,1X,*CYCLES/SEC*)
4000  FORMAT(/T5,*UNDAMPED PERIOD =*,E12.5,16X,*TIME TO HALF AMP. =*,
     1 E12.5,14X,*TIME TO ONE TENTH AMP. =*,E12.5,/T7,*DAMPED PERIOD =*,
     2 E12.5,14X,*CYCLES TO HALF AMP. =*,E12.5,12X,*CYCLES TO ONE *,
     3 *TENTH AMP. =*,E12.5,/T39,*ONE OVER CYCLES TO HALF AMP. =*,E12.5,
     4 3X,*ONE OVER CYCLES TO ONE TENTH AMP. =*,E12.5)
4500  FORMAT(/T5,*UNDAMPED PERIOD =*,E12.5,14X,*TIME TO DOUBLE AMP. =*,
     1 E12.5,14X,*TIME TO TEN TIMES AMP. =*,E12.5,/T7,*DAMPED PERIOD =*,
     2 E12.5,12X,*CYCLES TO DOUBLE AMP. =*,E12.5,12X,*CYCLES TO TEN *,
     3 *TIMES AMP. =*,E12.5,/T37,*ONE OVER CYCLES TO DOUBLE AMP. =*,
     4 E12.5,3X,*ONE OVER CYCLES TO TEN TIMES AMP. =*,E12.5)
5000  FORMAT(/T5,*CHARACTERISTIC EQUATION COEFFICIENTS*/T5,36(*-*)/T5,*A
     1 =*,1PE12.5,9H S**5 B =,1PE12.5,9H S**4 C =,1PE12.5,9H S**3 D =,
     2 1PE12.5,9H S**2 E =,1PE12.5,9H S**1 F =,1PE12.5,5H S**0)
999   RETURN
      END
```

```
      SUBROUTINE FILLIT(IRC)
C
C     TO INITIALIZE A 4X4 DETERMINANT WITH POLYNOMIAL TERMS
C          IRC - POSITION OF FIRST TERM ON DIAGONAL
C          NORD - ARRAY OF ORDERS OF POLYNOMIAL TERMS
C          D - ARRAY OF COEFFICIENTS OF POLYNOMIAL TERMS
C
      COMMON COMON(4400)
      COMMON / 5 / D(12,12,11)
      EQUIVALENCE (NORD  ,COMON(    1))
      EQUIVALENCE (XH    ,COMON(1462))
      EQUIVALENCE (XA    ,COMON(1463))
      EQUIVALENCE (XAD   ,COMON(1464))
      EQUIVALENCE (XQ    ,COMON(1465))
      EQUIVALENCE (XU    ,COMON(1467))
      EQUIVALENCE (ZH    ,COMON(1468))
      EQUIVALENCE (ZA    ,COMON(1469))
      EQUIVALENCE (ZAD   ,COMON(1470))
      EQUIVALENCE (ZQ    ,COMON(1471))
      EQUIVALENCE (ZU    ,COMON(1473))
      EQUIVALENCE (MH    ,COMON(1474))
      EQUIVALENCE (MA    ,COMON(1475))
      EQUIVALENCE (MAD   ,COMON(1476))
      EQUIVALENCE (MQ    ,COMON(1477))
      EQUIVALENCE (MU    ,COMON(1479))
      EQUIVALENCE (V     ,COMON( 755))
      EQUIVALENCE (GAMA  ,COMON( 757))
      EQUIVALENCE (SINA  ,COMON(1460))
      EQUIVALENCE (COSA  ,COMON(1461))
      EQUIVALENCE (INDCLS,COMON( 725))
      EQUIVALENCE (INDGST,COMON( 726))
      EQUIVALENCE (INDMOD,COMON( 727))
C
      REAL MH,MA,MAD,MQ,MD,MU
      INTEGER NORD(12,12),ORD(4,4)
C
      DATA DTR/57.29578/,G/32.174/
      DATA ORD/1,3*0,2*1,2,0,3*1,4*0,1/
C
C
C     ZERO DETERMINANT
C
      DO 10 I=1,4
      DO 10 J=1,4
      NORD(I,J)=-1
      DO 10 K=1,11
 10   D(I,J,K)=0.0
C
C     SET DIAGONAL TO 1.0
C
      DO 20 I=1,4
      NORD(I,I)=0
```

```
 20     D(1,1,1)=1.0
C
C       FILL ARRAY WITH ORDERS OF POLYNOMIALS OF 4X4 DETERMINANT
C
        IR=IRC-1
        DO 30 I=1,4
        IR=IR+1
        IC=IRC-1
        DO 30 J=1,4
        IC=IC+1
 30     NORD(IR,IC)=ORD(I,J)
C
C       FILL ARRAY WITH COEFFICIENTS OF POLYNOMIALS OF 4X4 DETERMINANT
C
        SING=SIN(GAMA/DTR)
        COSG=COS(GAMA/DTR)
        IR=IRC
        IC=IRC
        D(IR,IC,1)=-XU
        D(IR,IC,2)=1.0
        IC=IC+1
        D(IR,IC,1)=G*COSG
        D(IR,IC,2)=V*SINA-XQ
        IC=IC+1
        D(IR,IC,1)=-XA
        D(IR,IC,2)=-XAD
        D(IR,IC+1,1)=-XH
        IR=IR+1
        IC=IRC
        D(IR,IC,1)=-ZU
        IC=IC+1
        D(IR,IC,1)=G*SING
        D(IR,IC,2)=-V*COSA-ZQ
        IC=IC+1
        D(IR,IC,1)=-ZA
        D(IR,IC,2)=V-ZAD
        D(IR,IC+1,1)=-ZH
        IR=IR+1
        IC=IRC
        D(IR,IC,1)=-MU
        IC=IC+1
        D(IR,IC,1)=0.
        D(IR,IC,2)=-MQ
        D(IR,IC,3)=1.0
        IC=IC+1
        D(IR,IC,1)=-MA
        D(IR,IC,2)=-MAD
        D(IR,IC+1,1)=-MH
        IR=IR+1
        IC=IRC
        D(IR,IC,1)=-SING
        D(IR,IC+1,1)=-V*COSG
```

```
D(IR,IC+2,1)=V*COSG
IC=IC+3
D(IR,IC,1)=C.
D(IR,IC,2)=1.C
RETURN
END
```

```
      SUBROUTINE DETVAL(NA,DANS)
      COMMON / 5 / A(12,12,11)
      EQUIVALENCE (NORD  ,COMCN(    1))
      COMMCN COMCN(44CC)
C
C     ******************************************************************
C     EVALUATION OF DETERMINANTS WITH POLYNOMIAL  TERMS
C
C          A(I,J,K) = 8 X 8 DETERMINANT
C          NORD(I,J) = ORDER OF EACH TERM -- MAX. = 10TH ORDER
C     ******************************************************************
C
      DIMENSION RESULT(81),ANS(81),R2(21),R3(31)
      DIMENSION CANS(81)
      DIMENSION NORD(12,12)
C
C     ******************************************************************
C     ZERO THE ORDER OF THE ANSWER AND THE ANSWER
C     ******************************************************************
C
      NA=0
       DO 10 I=1,81
 10   ANS(I)=0.0
C
C     ******************************************************************
C     THE FOLLOWING FINDS THE PROPER TERMS OF THE DETERMINANT TO
C         MULTIPLY AND KEEPS A RECORD OF THE SIGN USING ISGN1,ISGN2,
C         ISGN3,ISGN4,ISGN5,ISGN6,ISGN7,AND KSIGN.
C     ******************************************************************
C
      DO 900 I=1,4
      KSIGN=0
      IF(NORD(1,I).EQ.-1)GO TO 900
      DO 800 J=1,4
      ISGN1=0
      IF(NORD(2,J).EQ.-1)GO TO 800
      IF(J-I)12,800,14
 12   ISGN1=ISGN1+1
 14   IR=NORD(1,I)+NORD(2,J)+1
      DO 16 ID=1,IR
 16   R2(ID)=0.0
      NO1=NORD(1,I)+1
      NO2=NORD(2,J)+1
      DO 18 IX=1,NO1
      DO 18 JX=1,NO2
      IND=IX+JX-1
 18   R2(IND)=A(1,I,IX)*A(2,J,JX)+R2(IND)
      DO 700 K=1,4
      IF(NORD(3,K).EQ.-1)GO TO 700
      ISGN2=0
      IF(K-I)20,700,22
 20   ISGN2=ISGN2+1
```

```
 22     IF(K-J)24,700,26
 24     ISGN2=ISGN2+1
 26     IS=IR+NORD(3,K)
        DO 28 ID=1,IS
 28     R3(ID)=0.0
        NO3=NORD(3,K)+1
        NO4=NO1+NO2-1
        DO 30 IX=1,NO4
        DO 30 JX=1,NO3
        IND=IX+JX-1
 30     R3(IND)=R2(IX)*A(3,K,JX)+R3(IND)
        DO 600 L=1,4
        IF(NORD(4,L).EQ.-1)GO TO 600
        ISGN3=0
        IF(L-I)32,600,34
 32     ISGN3=ISGN3+1
 34     IF(L-J)36,600,38
 36     ISGN3=ISGN3+1
 38     IF(L-K)40,600,42
 40     ISGN3=ISGN3+1
 42     IT=IS+NORD(4,L)
        DO 44 ID=1,IT
 44     RESULT(ID) = 0
        NO5=NORD(4,L)+1
        NO6=NO3+NO4-1
        DO 46 IX=1,NO6
        DO 46 JX=1,NO5
        IND=IX+JX-1
   46   RESULT(IND)=R3(IX)*A(4,L,JX)+RESULT(IND)
        NR=IT-1
C
C       ****************************************************************
C       CALCULATE THE SIGN OF THE TERM USING MOD 2.
C       ****************************************************************
C
        KSIGN=ISGN1+ISGN2+ISGN3
        IF(KSIGN-(KSIGN/2)*2)151,154,151
 151    DO 152 IM=1,IT
 152    RESULT(IM)=-RESULT(IM)
 154    DO 156 IQ=1,IT
 156    ANS(IQ)=ANS(IQ)+RESULT(IQ)
        NA=MAX0(NA,NR)
C    1111 TO 1114 DELETED
 600    CONTINUE
 700    CONTINUE
 800    CONTINUE
 900    CONTINUE
C
C       REVERSE POLYNOMIAL SO THAT IT GOES FROM HIGHEST DEGREE
C       TO LOWEST DEGREE.
C
        NC=NA+1
```

```
      DO 910 I=1,ND
      J=ND+1-I
  910 DANS(I)=ANS(J)
      RETURN
      END
```

```
      SUBROUTINE DMULR(COEF,N1,ROCTR,ROOTI)
C
C***********************************************************************
C
C     POLYNOMIAL ROOT FINDER SUBRCUTINE....
C
C     ITERATIVE METHOD FOR POLYNOMIAL EQUATIONS ....
C
C     THIS METHOD APPROXIMATES THE FUNCTION F(Z) BY A QUADRATIC
C     WHICH MAY, IN GENERAL, HAVE COMPLES COEFFICIENTS AND DOES NOT
C     REQUIRE A KNOWLEDGE OF THE DERIVATIVE OF F(Z) THOUGH
C     THE FUNCTION F(Z) MUST BE EVALUATED ONCE PER ITERATION ....
C
C     THIS SUBROUTINE FINDS REAL AND COMPLEX ROOTS OF A POLYNOMIAL
C     WITH REAL COEFFICIENTS ....
C
C     USE OF MULLER SUBROUTINE ....
C     1. CALL CMULR  (COEF,N1,ROOTR,ROOTI) ....
C     2.  COEF IS THE TAG OF THE ARRAY OF COEFFICIENTS.
C     THE COEFFICIENTS MUST BE ORDERED FROM HIGHEST DEGREE
C     TO LOWEST DEGREE.
C     3.  N1 IS DEGREE OF THE POLYNOMIAL .
C     4.  ROOTR IS THE TAG OF THE ARRAY WHERE THE REAL PARTS
C     OF THE COMPLEX ROOTS ARE STORED.
C     5.  ROOTI IS THE TAG OF THE ARRAY WHERE THE IMAGINARY
C     PARTS OF THE COMPLEX ROOTS ARE STORED ....
C
C     ALL ARITHMETIC IS IN THE COMPLEX MODE ....
C     THEREFORE UNDER-FLOW IS NORMAL FOR REAL ROOTS ....
C
C     MULTIPLE ROOTS DECREASES ACCURACY OF THIS SUBROUTINE .
C     WHEN MULTIPLICITY IS FOUR THE ACCURACY DECREASES TO
C     ABOUT TWO PLACES ....
C
      DIMENSION COEF(1),ROOTR(1),ROOTI(1),COE(121)
      N4=0
      I=N1+1
 1    IF(COEF(I))3,2,3
 2    N4=N4+1
      ROOTR(N4)=0.0
      ROOTI(N4)=0.0
      I=I-1
      IF(N4-N1)1,37,1
 3    I=1
 4    IF(COEF(I))6,5,6
 5    N1=N1-1
      I=I+1
      GO TO 4
 6    N2=N1+1
      DO 7 N=1,N2
 7    COE(N)=COEF(I-1+N)
 10   AXR=0.8
```

```
      AXI=0.0
      L=1
      N3=1
      ALP1R=AXR
      ALP1I=AXI
      M=1
      GO TO 99
C
 11   BET1R=TEMR
      BET1I=TEMI
      AXR=0.85
      ALP2R=AXR
      ALP2I=AXI
      M=2
      GO TO 99
C
 12   BET2R=TEMR
      BET2I=TEMI
      AXR=0.9
      ALP3R=AXR
      ALP3I=AXI
      M=3
      GO TO 99
C
 13   BET3R=TEMR
      BET3I=TEMI
 14   TE1=ALP1R-ALP3R
      TE2=ALP1I-ALP3I
      TE5=ALP3R-ALP2R
      TE6=ALP3I-ALP2I
      TEM=TE5*TE5+TE6*TE6
      TE3=(TE1*TE5+TE2*TE6)/TEM
      TE4=(TE2*TE5-TE1*TE6)/TEM
      TE7=TE3+1.0
      TE9=TE3*TE3-TE4*TE4
      TE10=2.0   *TE3*TE4
      DE15=TE7*BET3R-TE4*BET3I
      DE16=TE7*BET3I+TE4*BET3R
      TE11=TE3*BET2R-TE4*BET2I+BET1R-DE15
      TE12=TE3*BET2I+TE4*BET2R+BET1I-DE16
      TE7=TE9-1.0
      TE1=TE9*BET2R-TE10*BET2I
      TE2=TE9*BET2I+TE10*BET2R
      TE13=TE1-BET1R-TE7*BET3R+TE10*BET3I
      TE14=TE2-BET1I-TE7*BET3I-TE10*BET3R
      TE15=DE15*TE3-DE16*TE4
      TE16=DE15*TE4+DE16*TE3
      TE1=TE13*TE13-TE14*TE14-4.0   *(TE11*TE15-TE12*TE16)
      TE2=2.0   *TE13*TE14-4.0   *(TE12*TE15+TE11*TE16)
      TEM= SQRT(TE1*TE1+TE2*TE2)
      IF(TE1)113,113,112
 113  TE4= SQRT(0.5   *(TEM-TE1))
```

```
      TE3=0.5    *TE2/TE4
      GO TO 111
C
  112 TE3= SQRT(C.5   *(TEM+TE1))
      IF(TE2)110,20C,200
  110 TE3=-TE3
  200 TE4=C.5    *TE2/TE3
  111 TE7=TE13+TE3
      TE8=TE14+TE4
      TE9=TE13-TE3
      TE10=TE14-TE4
      TE1=2.0    *TE15
      TE2=2.0    *TE16
      IF(TE7*TE7+TE8*TE8-TE9*TE9-TE10*TE10)204,204,205
  204 TE7=TE9
      TE8=TE10
  205 TEM=TE7*TE7+TE8*TE8
      TE3=(TE1*TE7+TE2*TE8)/TEM
      TE4=(TE2*TE7-TE1*TE8)/TEM
      AXR=ALP3R+TE3*TE5-TE4*TE6
      AXI=ALP3I+TE3*TE6+TE4*TE5
      ALP4R=AXR
      ALP4I=AXI
      M=4
      GO TO 99
C
   15 N6=1
C**************************************************************************
   38 IF( ABS(HELL)+ ABS(BELL)-1.CE-20)18,18,16
   16 TE7= ABS(ALP3R-AXR)+ ABS(ALP3I-AXI)
      IF(TE7/( ABS(AXR)+ ABS(AXI))-1.0E-7)18,18,17
C**************************************************************************
   17 N3=N3+1
      ALP1R=ALP2R
      ALP1I=ALP2I
      ALP2R=ALP3R
      ALP2I=ALP3I
      ALP3R=ALP4R
      ALP3I=ALP4I
      BET1R=BET2R
      BET1I=BET2I
      BET2R=BET3R
      BET2I=BET3I
      BET3R=TEMR
      BET3I=TEMI
      IF(N3-10C)14,18,18
   18 N4=N4+1
      ROOTR(N4)=ALP4R
      ROOTI(N4)=ALP4I
      N3=0
   41 IF(N4-N1)3C,37,37
C**************************************************************************
```

```
 30     IF( ABS(ROOTI(N4))-1.0E-5)10,10,31
 31     GO TO (32,10),L
 32     AXR=ALP1R
        AXI=-ALP1I
        ALP1I=-ALP1I
        M=5
        GO TO 99
 33     BET1R=TEMR
        BET1I=TEMI
        AXR=ALP2R
        AXI=-ALP2I
        ALP2I=-ALP2I
        M=6
        GO TO 99
C
 34     BET2R=TEMR
        BET2I=TEMI
        AXR=ALP3R
        AXI=-ALP3I
        ALP3I=-ALP3I
        L=2
        M=3
 99     TEMR=COE(1)
        TEMI=0.0
        DO 100 I=1,N1
        TE1=TEMR*AXR-TEMI*AXI
        TEMI=TEMI*AXR+TEMR*AXI
 100    TEMR=TE1+COE(I+1)
        FELL=TEMR
        BELL=TEMI
 42     IF(N4)102,103,102
 102    DO 101 I=1,N4
        TEM1=AXR-ROOTR(I)
        TEM2=AXI-ROOTI(I)
        TE1=TEM1*TEM1+TEM2*TEM2
        TE2=(TEMR*TEM1+TEMI*TEM2)/TE1
        TEMI=(TEMI*TEM1-TEMR*TEM2)/TE1
 101    TEMR=TE2
 103    GO TO (11,12,13,15,33,34),M
C
 37     CONTINUE
        RETURN
        END
```

```
SUBROUTINE IIIX3
COMMON COMON(4400)
COMMON / 5 /AA(12,12,11)
EQUIVALENCE (AORDER,COMON(    1))
EQUIVALENCE (RHO    ,COMON( 754))
EQUIVALENCE (U      ,COMON( 755))
EQUIVALENCE (S      ,COMON( 728))
EQUIVALENCE (GWT    ,COMON( 730))
EQUIVALENCE (BSPAN  ,COMON( 732))
EQUIVALENCE (ZIXB   ,COMON(1456))
EQUIVALENCE (ZIZB   ,COMON(1457))
EQUIVALENCE (ZIXZB  ,COMON(1458))
EQUIVALENCE (ALFAI  ,COMON(1459))
EQUIVALENCE (ALFAA  ,COMON( 758))
EQUIVALENCE (ALFAX  ,COMON( 759))
EQUIVALENCE (GAMA   ,COMON( 757))
EQUIVALENCE (CYB    ,COMON(1425))
EQUIVALENCE (CYBD   ,COMON(1426))
EQUIVALENCE (CYP    ,COMON(1427))
EQUIVALENCE (CYR    ,COMON(1428))
EQUIVALENCE (CYDR   ,COMON(1429))
EQUIVALENCE (CYD1   ,COMON(1430))
EQUIVALENCE (CLB    ,COMON(1431))
EQUIVALENCE (CLBD   ,COMON(1432))
EQUIVALENCE (CLP    ,COMON(1433))
EQUIVALENCE (CLR    ,COMON(1434))
EQUIVALENCE (CLDR   ,COMON(1435))
EQUIVALENCE (CLD1   ,COMON(1436))
EQUIVALENCE (CNB    ,COMON(1437))
EQUIVALENCE (CNBD   ,COMON(1438))
EQUIVALENCE (CNP    ,COMON(1439))
EQUIVALENCE (CNR    ,COMON(1440))
EQUIVALENCE (CNDR   ,COMON(1441))
EQUIVALENCE (CND1   ,COMON(1442))
EQUIVALENCE (CYD2   ,COMON(1443))
EQUIVALENCE (CYD3   ,COMON(1444))
EQUIVALENCE (DA     ,COMON(1445))
EQUIVALENCE (D1     ,COMON(1446))
EQUIVALENCE (D2     ,COMON(1447))
EQUIVALENCE (D3     ,COMON(1448))
EQUIVALENCE (CLD2   ,COMON(1449))
EQUIVALENCE (CLD3   ,COMON(1450))
EQUIVALENCE (ALT    ,COMON( 756))
EQUIVALENCE (CND2   ,COMON(1454))
EQUIVALENCE (CND3   ,COMON(1455))
EQUIVALENCE (INDCLS,COMON( 725))
EQUIVALENCE (INDBAF,COMON( 744))
EQUIVALENCE (INDGST,COMON( 726))
EQUIVALENCE (TITLE  ,COMON( 795))
EQUIVALENCE (ANRP   ,COMON(1491))
EQUIVALENCE (ANPP   ,COMON(1490))
EQUIVALENCE (ANBDP  ,COMON(1489))
```

```
      EQUIVALENCE (ANBP   ,COMON(1488))
      EQUIVALENCE (ALRP   ,COMON(1487))
      EQUIVALENCE (ALPP   ,COMON(1486))
      EQUIVALENCE (ALBDP  ,COMON(1485))
      EQUIVALENCE (ALBP   ,COMON(1484))
      EQUIVALENCE (YR     ,COMON(1483))
      EQUIVALENCE (YP     ,COMON(1482))
      EQUIVALENCE (YBD    ,COMON(1481))
      EQUIVALENCE (YB     ,COMON(1480))
      INTEGER AORDER(12,12)
      DIMENSION ROOTR(5),ROOTI(5),DEL(5)
      DIMENSION ROOTRD(5),ROOTID(5),TITLE(10)
      COMPLEX  CCM     , CCMA    , COMB    , ANUM    , ADEN
      DATA DTR/57.29578/
      DATA G,EE / 32.174,2.71828 /
      FUNP(X)=1./A*(AP*X**3+BP*X**2+CP*X+DP)
      FUNB(X)=1./A*(AB*X**3+BB*X**2+CB*X+DB)
      PT(T)=XKP+XKPR*EE**(XTR*T)+XKPS*EE**(XTS*T)+
     1 XKPDR*EE**(CCN*T)*COS(CCNA*T+PSIP/DTR)
      BT(T)=XKP+XKBR*EE**(XTR*T)+XKBS*EE**(XTS*T)+
     1 XKBDR*EE**(CCN*T)*COS(CONA*T+PSIB/DTR)
      PDA(T)=XKP*T+XKPR*TR*(1.-EE**(XTR*T))+XKPS*TS*
     1 (1.-EE**(XTS*T))+CON2*(EE**(CCN*T)*(CCN*COS(CCNA*T+PSIPR)
     2 +CONA*SIN(CONA*T+PSIPR))+CCN3)
C     M=0 USE NONDIMENSIONAL INPUT DATA (STAB AXIS)
C FOR J=0. USE NON-DIMEN. STAB. DERIVATIVES WITH UNITS OF 1 PER RADIAN.
 107  WRITE(6,202)RHO,ALFAI,CYB,CYBD,CYP,CYR,U,ALFAA,CLB,CLBD,CLP,CLR,
     1 S,GAMA,CNB,CNBD,CNP,CNR,BSPAN,ALT,GWT,DA,CYDR,CYD1,CYD2,CYD3,
     2 ZIXB,D1,CLDR,CLD1,CLD2,CLD3,ZIZB,D2,CNDR,CND1,CND2,CND3,ZIXZB,D3
 202  FORMAT(T2,*INPUT DATA (NON-DIMENSIONAL) PER RADIAN     LATERAL MODE
     1*,/T2,13(*---*),4X,6(*--*),
     1 /1HO,1X,*RHO =*,1PE12.4,T24,*ALFAI =*,1PE12.4,T46,* CYB*,1PE12.4,
     2 T67,*CYBD =*,1PE12.4,T86,* CYP =*,1PE12.4,T107,* CYR =*,1PE12.4/
     3  T5,*V =*,1PE12.4,T24,*ALFAA =*,1PE12.4,T46,* CLB =*,1PE12.4,
     4 T67,*CLBD =*,1PE12.4,T86,* CLP =*,1PE12.4,T107,* CLR =*,1PE12.4/
     5  T5,*S =*,1PE12.4,T24,* GAMA =*,1PE12.4,T46,* CNB =*,1PE12.4,
     6 T67,*CNBD =*,1PE12.4,T86,* CNP =*,1PE12.4,T107,* CNP =*,1PE12.4/
     7 T2,*SPAN =*,1PE12.4,T24,*  ALT =*,1PE12.4/
     8T3,*GWT =*,1PE12.4,T24,*   DA =*,1PE12.4,T46,*CYDR =*,1PE12.4,
     9 T67,*CYD1 =*,1PE12.4,T86,*CYD2 =*,1PE12.4,T107,*CYD3 =*,1PE12.4/
     AT3,*IXI =*,1PE12.4,T24,*   D1 =*,1PE12.4,T46,*CLDR =*,1PE12.4,
     B T67,*CLD1 =*,1PE12.4,T86,*CLD2 =*,1PE12.4,T107,*CLD3 =*,1PE12.4/
     CT3,*IZI =*,1PE12.4,T24,*   D2 =*,1PE12.4,T46,*CNDR =*,1PE12.4,
     D T67,*CND1 =*,1PE12.4,T86,*CND2 =*,1PE12.4,T107,*CND3 =*,1PE12.4/
     E T2,*IXZI =*,1PE12.4,T24,*   D3 =*,1PE12.4)
1000  ALFA2=(ALFAX-ALFAA)/57.295779
      SINA=SIN(ALFA2)
      COSA=COS(ALFA2)
      CYDA=(CYD1*D1+CYD2*D2+CYD3*D3)/DA
      CLDA=(CLD1*D1+CLD2*D2+CLD3*D3)/DA
      CNDA=(CND1*D1+CND2*D2+CND3*D3)/DA
      SCLDA=CLDA*SINA
```

```
      CLDA=CLDA*COSA-CNDA*SINA
      CNDA=CNDA*COSA+SCLDA
      SCLDR=CLDR*SINA
      CLDR=CLDR*COSA-CNDR*SINA
      CNDR=CNDR*COSA+SCLDR
      SCLBD=CLBD*SINA
      CLBD=CLBD*COSA-CNBD*SINA
      CNBD=CNBD*COSA+SCLBD
      SCLB=CLB*SINA
      CLB=CLB*COSA-CNB*SINA
      CNB=CNB*COSA+SCLB
      SCYP=CYP*SINA
      CYP=CYP*COSA-CYR*SINA
      CYR=CYR*COSA+SCYP
      SCLP=CLP*SINA**2
      SCCLP=(CLP-CNR)*SINA*COSA
      SCLR=CLR*SINA**2
      SCCLR=(CLR+CNP)*SINA*COSA
      CLP=CLP*COSA**2+CNR*SINA**2-SCCLR
      CLR=CLR*COSA**2-CNP*SINA**2+SCCLP
      CNP=CNP*COSA**2-SCLP+SCCLP
      CNR=CNR*COSA**2+SCLP+SCCLR
      ADD=(ALFAI-ALFAX)/DTR
      SA=SIN(ADD)
      CA=COS(ADD)
      TAA=2.0*ADD
      STA=SIN(TAA)
      CTA=COS(TAA)
      ZIXS=ZIXB*CA**2   +ZIZB*SA**2   -ZIXZB*STA
      ZIZS=ZIZB*CA**2   +ZIXB*SA**2   +ZIXZB*STA
      ZIXZS=ZIXZB*CTA+.5*(ZIXB-ZIZB)*STA
146   RSU=RHO*S*U
      ZMASS=GWT/32.174
      RSUM=RSU/ZMASS
      RSUX=RSU*BSPAN/ZIXS
      RSUZ=RSU*BSPAN/ZIZS
      YV=(RSUM/2.0 )*CYB
      YB=U*YV
      YVD=(RSUM*BSPAN/(4.0*U))*CYBD
      YBD=U*YVD
      YP=(RSUM*BSPAN/4.0)*CYP
      YR=(RSUM*BSPAN/4.0)*CYR
      YDA=(RSUM*U/2.0)*CYDA
      YDR=(RSUM*U/2.0)*CYDR
      ALB=(RSUX*U/2.0)*CLB
      ALBD=(RSUX*BSPAN/4.0)*CLBD
      ALP=(RSUX*BSPAN/4.0)*CLP
      ALR=(RSUX*BSPAN/4.0)*CLR
      ALDA=(RSUX*U/2.0)*CLDA
      ALDR=(RSUX*U/2.0)*CLDR
      ANB=(RSUZ*U/2.0)*CNB
      ANBD=(RSUZ*BSPAN/4.0)*CNBD
```

```
      ANP=(RSUZ*BSPAN/4.0)*CNP
      ANR=(RSUZ*BSPAN/4.0)*CNR
      ANDA=(RSUZ*C/2.0)*CNDA
      ANDR=(RSUZ*C/2.0)*CNDR
      WRITE(6,300)YB,YBD,YP,YR,YDA,YDR,ALB,ALBD,ALP,ALR,ALDA,ALDR,
     1 ANB,ANBD,ANP,ANR,ANDA,ANDR
  300 FORMAT(1H0,5X33HDIMENSIONAL STABILITY DERIVATIVES
     1 /1H0,2X4HYB =1PE12.4,4X5HYBD =1PE12.4,5X4HYP =1PE12.4,5X4HYR =,
     2 1PE12.4,4X5HYDA =1PE12.4,4X5HYDR =1PE12.4/3X,4HLB =1PE12.4,4X,
     3 5HLBD =1PE12.4,5X4HLP =1PE12.4,5X4HLR =1PE12.4,4X5HLDA =1PE12.4,
     4 4X5HLDR =1PE12.4/3X4HNB =1PE12.4,4X5HNBD =1PE12.4,5X4HNP =1PE12.4
     5 ,5X4HNR =1PE12.4,4X5HNDA =1PE12.4,4X5HNDR =1PE12.4)
  145 XM=ZIXZS/ZIXS
      ZM=ZIXZS/ZIZS
      DXZ=1.0-((ABS(ZIXZS)**2)/(ZIXS*ZIZS))
      ALBP=(ALB+XM*ANB)/DXZ
      ALBDP=(ALBD+XM*ANBD)/DXZ
      ANBP=(ANB+ZM*ALB)/DXZ
      ANBDP=(ANBD+ZM*ALBD)/DXZ
      ALPP=(ALP+XM*ANP)/DXZ
      ANPP=(ANP+ZM*ALP)/DXZ
      ALRP=(ALR+XM*ANR)/DXZ
      ANRP=(ANR+ZM*ALR)/DXZ
      ALDAP=(ALDA+XM*ANDA)/DXZ
      ANDAP=(ANDA+ZM*ALDA)/DXZ
      ALDRP=(ALDR+XM*ANDR)/DXZ
      ANDRP=(ANDR+ZM*ALDR)/DXZ
      YP=YP+U*SIN(ALFAX/DTR)
      YR=YR+U*(1.-COS(ALFAX/DTR))
      GAMA=GAMA+ALFAX
      WRITE(6,301)ALFAI,ALFAA,ZIXS,ZIZS,ZIXZS,ALBP,ALBDP,
     1 ALPP,ALRP,ALDAP,ALDRP,ANBP,ANBDP,ANPP,ANRP,ANDAP,ANDRP
  301 FORMAT(1H0,5X40HDIMENSIONAL STABILITY DERIVATIVES PRIMED
     1 /1H0,6HALFAI=1PE12.4,3X,6HALFAA=1PE12.4,26X4HIX =1PE12.4,
     2 5X4HIZ =1PE12.4,4X5HIXZ =1PE12.4,
     X /7H   LBP =1PE12.4,3X6HLBDP =1PE12.4,4X5HLPP =1PE12.4,
     3 4X5HLRP =1PE12.4,3X6HLDAP =1PE12.4,3X6HLDRP =1PE12.4/7H   NBP =,
     4 1PE12.4,3X6HNBDP =1PE12.4,4X5HNPP =1PE12.4,4X5HNRP =1PE12.4,
     5 3X6HNDAP =1PE12.4,3X6HNDRP =1PE12.4)
      XKON=2.0*3.14159
      GDD=GAMA/DTR
      SG=SIN(GDD)
      CG=COS(GDD)
      GSG=G*SG
      GCG=G*CG
      IF(INDBAF.NE.0) GO TO 250
      RZERO=0.0
      IROOT=1
C               LATERAL-DIRECTIONAL DENOMINATOR
      A=1.0-YVD
      B0=-ALPP-ANRP-YV+ANBDP*(1.0-(YR/U))-ALBDP*(YP/U)
     1    +YVD*(ANRP+ALPP)
```

```
      C=ANRP*ALPP-ALRP*ANPP+ANBP*(1.0-(YR/U))+YV*(ALPP+ANRP)
     1    -(YP/U)*ALBP-ANBDP*(ALPP*(1.0-(YR/U))+(YP/U)*ALRP+(GSG/U))
     2     +ALBDP*(ANPP*(1.C-(YR/U))+(YP/U)*ANRP-(GCG/U))
     3      +YVD*(ALRP*ANPP-ANRP*ALPP)
      D=ALRP*ANPP*YV-ANRP*ALPP*YV+(YP/U)*(ANRP*ALBP-ALRP*ANBP)
     1  +(1.0-(YR/U))*(ALBP*ANPP-ALPP*ANBP)-(GCG/U)*ALBP
     2    -(GSG/U)*ANBP+ANBDP*((GSG/U)*ALPP-(GCG/U)*ALPP)
     3    +ALBDP*((GCG/U)*ANRP-(GSG/U)*ANPP)
      E=(GCG/U)*(ANRP*ALBP-ALRP*ANBP)+(GSG/U)*(ALPP*ANBP-ALBP*ANPP)
      WRITE (6,176)
  176 FORMAT(/T7,*LATERAL DIRECTIONAL DENOMINATOR*)
      DEL(1)=A
      DEL(2)=BD
      DEL(3)=C
      DEL(4)=D
      DEL(5)=E
      N=4
      CALL  DMULR (DEL,N,ROOTRD,ROOTID)
      M=1
   60 WRITE(6,400)
  400 FORMAT(T2,*ROOTS (COMPLEX FORM)*)
      WRITE(6,401)RZERO,RZERO
  401 FORMAT(T2,4X,F4.1,13X,F4.1)
 1007 DO 1002 I=1,N
      IF( ABS(ROOTID(I)).LT.1.E-5)GO TO 1001
      WRITE(6,21)ROOTRD(I),ROOTID(I)
 21   FORMAT(1H ,3X1PE12.4,5X1PE12.4)
      GO TO 1002
 1001 WRITE(6,21)ROOTRD(I)
 1002 CONTINUE
      DO 800 I=1,N
      ROOTR(I)=ROOTRD(I)
 800  ROOTI(I)=ROOTID(I)
 94   IF(1.E-4-ABS(ROOTI(1)))119,120,120
 119  W1=SQRT(ROOTR(1)**2+ROOTI(1)**2)
      WD1=ABS(ROOTI(1))
      Z1=-ROOTR(1)/W1
      W3=W1/XKON
      WD3=WD1/XKON
 115  IF(1.E-4-ABS(ROOTI(3)))116,117,117
 116  WD=SQRT(ROOTR(3)**2+ROOTI(3)**2)
      WDD=ABS(ROOTI(3))
      ZD=-ROOTR(3)/WD
      W4=WD/XKON
      WD4=WDD/XKON
      IF(ABS(ROOTI(1)).LT..001)GO TO 91
      IROOT=2
      IF(WD-W1)173,173,174
 174  WRITE(6,39)Z1,W1,ZD,WD,WDD,W3,W4,WD4
      I1=1
      GO TO 222
 173  WRITE(6,39)ZD,WD,Z1,W1,WD1,W4,W3,WD3
```

```
39    FORMAT(1H0,*Z1 =*,1PE14.6,1X,*W1 =*,1PE14.6,1X,*RAD/SEC*,4X,*Z2 =*
     1,1PE14.6,1X,*W2 =*,1PE14.6,1X,*RAD/SEC*,4X,*WDDR =*,1PE14.6,1X,
     2 *RAD/SEC*/T2,22X,*=*,1PE14.6,1X,*CYCLES/SEC*,23X,*=*,1PE14.6,1X,
     3 *CYCLES/SEC*,6X,*=*,1PE14.6,1X,*CYCLES/SEC*)
      DUMB=Z1
      Z1=ZD
      ZD=DUMB
      STUPE=W1
      W1=WD
      WD=STUPE
      STUPID=WD1
      WD1=WDD
      WDD=STUPID
      I1=1
      GO TO 222
120   TD1=-1./ROOTR(1)
      ROOTI(1)=0.0
      IF(1.E-4-ABS(ROOTI(2)))130,131,131
130   WD=SQRT(ROOTR(2)**2+ROOTI(2)**2)
      WDD=ABS(ROOTI(2))
      ZD=-ROOTR(2)/WD
      W4=WD/XKON
      WD4=WDD/XKON
      TD2=-1./ROOTR(4)
      GO TO 91
131   TD2=-1./ROOTR(2)
      GO TO 115
91    I1=2
      IF(ABS(TD1).LT.ABS(TD2))GO TO 89
      WRITE(6,170)TD1,TD2,ZD,WD,WDD,W4,WD4
170   FORMAT(1H0,1X,4HTS =1PE14.6,3X,4HTR =1PE14.6,3X,5HZDR =1PE14.6,3X,
     15HWDR =1PE14.6,8H RAD/SEC,6X,6HWDDR =1PE14.6,8H RAD/SEC/1H ,69X,
     2*=*,1PE14.6,11H CYCLES/SEC,8X,*=*,1PE14.6,11H CYCLES/SEC)
      TS=TD1
      TR=TD2
      I1=2
      GO TO 222
89    WRITE(6,170)TD2,TD1,ZD,WD,WDD,W4,WD4
      TS=TD2
      TR=TD1
      GO TO 222
117   TD3=-1./ROOTR(3)
      TD4=-1./ROOTR(4)
      WD=W1
      WDD=WD1
      ZD=Z1
      IF(ABS(ROOTI(1)).GT..001)GO TO 109
      WRITE (6,171)TD1,TD2,TD3,TD4
      IROOT=0
171   FORMAT(1H0,7H    T1  =1PE14.6,4X,7H    T2  =1PE14.6,4X,7H    T3  =1PE14.
     16,4X,6H   T4 =1PE14.6)
      GO TO 221
```

```
109   I1=2
      IF(ABS(TD3).LT.ABS(TD4))GC TO 124
      WRITE(6,170)TD3,TD4,Z1,W1,WD1,W3,WD3
      TS=TD3
      TR=TD4
      GO TO 222
124   WRITE(6,170)TD4,TD3,Z1,W1,WD1,W3,WD3
      TS=TD4
      TR=TD3
222   PER=XKON/(WD*SQRT(1.-ABS(ZD)**2))
      TDR=XKON/WD
      TTO1=.69315/(ABS(ZD)*WD)
      TTO2=2.30259/(ABS(ZD)*WD)
      CTO1=TTO1/PER
      CTO2=TTO2/PER
      CTO3=1.0/CTO1
      CTO4=1.0/CTO2
      IF(ZD)223,223,224
224   WRITE(6,114)TDR,TTO1,TTO2,PER,CTO1,CTC2,CTO3,CTC4
 114  FORMAT(/T2,*DUTCH ROLL MODE*/T8,*TDR   =*,E13.5,
     1 13X19HTIME TO HALF AMP. =E13.5,16X24HTIME TO CNE TENTH AMP. =,
     2 E13.5,/1H ,6X6HTDDR =E13.5,11X21HCYCLES TO HALF AMP. =,
     X E13.5,14X26HCYCLES TO CNE TENTH AMP. =E13.5,
     3/28X30HONE OVER CYCLES TO HALF AMP. =E13.5,5X35HCNE CVER CYCLES TO
     4 ONE TENTH AMP. =E13.5)
      GO TO 165
223   WRITE(6,402)TDR,TTO1,TTO2,PER,CTC1,CTO2
 402  FORMAT(/T2,*DUTCH ROLL MCDE*/T13,*TDR   =*,E13.5,
     1 6X21HTIME TC DOUBLE AMP. =E13.5,16X24HTIME TC TEN TIMES AMP. =,
     2 E13.5,/1H ,11X,6HTDDR =E13.5,4X,23HCYCLES TC DCUBLE AMP. =,
     3 E13.5,14X26HCYCLES TO TEN TIMES AMP. =E13.5)
165   GO TO(149,221),I1
149   PER=XKON/(W1*SQRT(1.-ABS(Z1)**2))
      TDR=XKON/W1
      TTO1=.69315/(ABS(Z1)*W1)
      TTO2=2.30259/(ABS(Z1)*W1)
      CTO1=TTO1/PER
      CTO2=TTO2/PER
      CTO3=1.0/CTO1
      CTO4=1.0/CTO2
      IF(Z1)164,164,169
169   WRITE(6,177)TDR,TTO1,TTO2,PER,CTO1,CTO2,CTO3,CTO4
 177  FORMAT(/T2,*LONG PERICD MCDE*,/T8,*TDR   =*,E13.5,
     1 13X19HTIME TO HALF AMP. =E13.5,16X24HTIME TO CNE TENTH AMP. =,
     2 E13.5,/1H ,6X6HTDDR =E13.5,11X21HCYCLES TO HALF AMP. =,
     X E13.5,14X26HCYCLES TO CNE TENTH AMP. =E13.5,
     3/28X30HONE OVER CYCLES TC HALF AMP. =E13.5,5X35HCNE CVER CYCLES TO
     4 ONE TENTH AMP. =E13.5)
      GO TO 221
164   WRITE(6,178)TDR,TTO1,TTO2,PER,CTO1,CTC2
 178  FORMAT(/T2,*LONG PERICD MCDE*,/T13,*TDR   =*,E13.5,
     1 6X21HTIME TO DOUBLE AMP. =E13.5,16X24HTIME TC TEN TIMES AMP. =,
```

```
    2 E13.5,/1H ,11X,6HTDDR =E13.5,4X,23HCYCLES TC DOUBLE AMP. =,
    3 E13.5,14X26HCYCLES TO TEN TIMES AMP. =E13.5)
221  WRITE (6,201)A,BC,C,D,E
201  FORMAT(/T2,*CHARACTERISTIC EQUATICN COEFFICIENTS*/T5,36(*-*)/T5,*A
    1 =*,1PE13.5,9H S**5 B =,1PE13.5,9H S**4 C =,1PE13.5,9H S**3 D =,
    2 1PE13.5,9H S**2 E =,1PE13.5,18H S**1 F = 0.0 S**0)
95   CON = -ZC*WL
     CONA = WD*SQRT(1.-ABS(ZD)**2)
     COM = CMPLX(CCN,CONA)
     COMA = COM*COM
     COMB = COMA*COM
     ANUM = (ALBDP*(YR/U-1.)+ALRP-YVD*ALRP)*CCMA
    1        +((YR/U-1.)*ALBP+ALBDP*GSG/U-YV)*CCM+ALBP*GSG/U
     ADEN = (YR/U-1.)*COMB+(ALRP*YP/U+GSG/U-ALPP*(YR/U-1.))*CCMA
    1        +(ALRP*GCG/U-ALPP*GSG/U)*CCM
     PTOB = SQRT((REAL(ANUM)**2+AIMAG(ANUM)**2)/
    1    (REAL(ADEN)**2+AIMAG(ADEN)**2))
     SIGMA=RHO/2.3769E-03
     PVMAG=DTR*PTOB /(U*SQRT(SIGMA))
     FSPTOB=WC**2*PTOB
     WRITE(6,500)PTOB,PVMAG,FSPTCB
500  FORMAT(/T2,*PHI TO BETA RATIO =*,1PE12.4,5X,*PHI TC EQUIV VEL =*,
    1 1PE12.4,5X,*FREQ. SQUARED TIMES PHI TO BETA RATIC =*,1PE12.4)
250  CALL FIL3X3(1)
     AORDER(1,4)=0
     AORDER(1,5)=0
     AORDER(2,4)=0
     AORDER(2,5)=0
     AORDER(3,4)=0
     AORDER(3,5)=0
     AA(1,4,1)=-YDA/U
     AA(1,5,1)=-YDR/U
     AA(2,4,1)=-ALDAP
     AA(2,5,1)=-ALDRP
     AA(3,4,1)=-ANDAP
     AA(3,5,1)=-ANDRP
     IF(INDGST.EQ.0)RETURN
     AA(1,6,1)=-YP/U
     AA(1,7,1)=-YB/U
     AA(1,8,1)=-YR/U
     AA(2,6,1)=-ALPP
     AA(2,7,1)=-ALBP
     AA(2,8,1)=-ALPP
     AA(3,6,1)=-ANPP
     AA(3,7,1)=-ANBP
     AA(3,8,1)=-ANPP
     RETURN
     END
```

```
      SUBROUTINE FIL3X3(IRC)
      COMMON COMON(4400)
      COMMON / 5 / D(12,12,11)
      EQUIVALENCE (NORD   ,COMON(    1))
      EQUIVALENCE (ANRP   ,COMON(1491))
      EQUIVALENCE (ANPP   ,COMON(1490))
      EQUIVALENCE (ANBDP  ,COMON(1489))
      EQUIVALENCE (ANBP   ,COMON(1488))
      EQUIVALENCE (ALRP   ,COMON(1487))
      EQUIVALENCE (ALPP   ,COMON(1486))
      EQUIVALENCE (ALBDP  ,COMON(1485))
      EQUIVALENCE (ALBP   ,COMON(1484))
      EQUIVALENCE (YR     ,COMON(1483))
      EQUIVALENCE (YP     ,COMON(1482))
      EQUIVALENCE (YBD    ,COMON(1481))
      EQUIVALENCE (YB     ,COMON(1480))
      EQUIVALENCE (GAMA   ,COMON( 757))
      EQUIVALENCE (U      ,COMON( 755))
C
C     TO INITIALIZE A 12X12 DETERMINANT WITH POLYNOMIAL TERMS AND
C     FILL IN A VARIABLE 3X3 DETERMINANT AT THE INDICATED
C     POSITION ON THE DIAGONAL
C           IRC - POSITION OF FIRST TERM ON DIAGONAL
C           NORD - ARRAY OF ORDERS OF POLYNOMIAL TERMS
C           D - ARRAY OF COEFFICIENTS OF POLYNOMIAL TERMS
C
C    1805 DELETED
      DIMENSION NORD(12,12)
      INTEGER ORD(3,3)
C
      DATA ORD / 4*1,2,3*1,2 /
      DATA DTR , G / 57.29578 , 32.174 /
C
C
C
C     ZERO DETERMINANT
C
      DO 10 I=1,3
      DO 10 J=1,3
      NORD(I,J)=-1
      DO 10 K=1,11
 10   D(I,J,K)=0.0
C
C     SET DIAGONAL TO 1.0
C
      DO 20 I=1,3
      NORD(I,I)=0
 20   D(I,I,1)=1.0
C
C     FILL ARRAY WITH ORDERS OF POLYNOMIALS OF 3X3 DETERMINANT
C
      IR=IRC-1
```

```
      DO 30 I=1,3
      IR=IR+1
      IC=IRC-1
      DO 30 J=1,3
      IC=IC+1
 30   NORD(IR,IC)=ORD(I,J)
C
C     FILL ARRAY WITH COEFFICIENTS OF POLYNOMIALS OF 3X3 DETERMINANT
C
      CDD=GAMA/DTR
      GSG=G*SIN(CDD)
      GCG=G*COS(CDD)
      IR=IRC
      IC=IRC
      D(IR,IC,1)=-YB/U
      D(IR,IC,2)=1.-YBD/U
      IC=IC+1
      D(IR,IC,1)=-GCG/U
      D(IR,IC,2)=-YP/U
      IC=IC+1
      D(IR,IC,1)=-GSG/U
      D(IR,IC,2)=1.-YR/U
      IR=IR+1
      IC=IRC
      D(IR,IC,1)=-ALBP
      D(IR,IC,2)=-ALBDP
      IC=IC+1
      D(IR,IC,1)=0.
      D(IR,IC,2)=-ALPP
      D(IR,IC,3)=1.
      IC=IC+1
      D(IR,IC,1)=0.
      D(IR,IC,2)=-ALRP
      IR=IR+1
      IC=IRC
      D(IR,IC,1)=-ANBP
      D(IR,IC,2)=-ANBDP
      IC=IC+1
      D(IR,IC,1)=0.
      D(IR,IC,2)=-ANPP
      IC=IC+1
      D(IR,IC,1)=0.
      D(IR,IC,2)=-ANRP
      D(IR,IC,3)=1.
      RETURN
      END
```

```
      SUBROUTINE GUST
      COMMON COMON(4400)
      COMMON / 5 / A(12,12,11),DUM(15),B(12,12,11),DUM1(16)
      EQUIVALENCE (AORDER,COMON(    1))
      EQUIVALENCE (BORDER,COMON( 145))
      EQUIVALENCE (INDGST,COMON( 726))
      EQUIVALENCE (INDMOD,COMON( 727))
      EQUIVALENCE (SPANF ,COMON( 732))
      EQUIVALENCE (INDTST,COMON( 746))
      EQUIVALENCE (V     ,COMON( 755))
      EQUIVALENCE (ALT   ,COMON( 756))
      EQUIVALENCE (LU    ,COMON( 760))
      EQUIVALENCE (LW    ,COMON( 761))
      EQUIVALENCE (SIGU  ,COMON( 762))
      EQUIVALENCE (SIGW  ,COMON( 763))
      EQUIVALENCE (XA    ,COMON(1463))
      EQUIVALENCE (XAD   ,COMON(1464))
      EQUIVALENCE (XQ    ,COMON(1465))
      EQUIVALENCE (XU    ,COMON(1467))
      EQUIVALENCE (ZA    ,COMON(1469))
      EQUIVALENCE (ZAD   ,COMON(1470))
      EQUIVALENCE (ZQ    ,COMON(1471))
      EQUIVALENCE (ZU    ,COMON(1473))
      EQUIVALENCE (MA    ,COMON(1475))
      EQUIVALENCE (MAD   ,COMON(1476))
      EQUIVALENCE (MQ    ,COMON(1477))
      EQUIVALENCE (MU    ,COMON(1479))
      INTEGER AORDER(12,12),BORDER(12,12)
      REAL LU,LV,LW,LVOV
      REAL MA,MAD,MQ,MU,LWOV
      IF(INDGST.EQ.0)GO TO 140
      XH = ALT
      IF(XH.LT.100.)XH=100.
      IF(XH.GE.1750..OR.INDTST.NE.0)GO TO 100
      LW=XH
      LV=145.*XH**.33
      LU=LV
      IF(XH.GT.1000.)GO TO 80
      SIGW=7.49-.33*ALOG10 (XH)
 60   SIGV=SQRT(LV*SIGW**2/LW)
      SIGU=SIGV
      GO TO 120
 80   SIGW=10.25-1.25*ALOG10(XH)
      GO TO 60
 100  LW=1750.
      LU=LW
      LV=LW
      IF(INDTST.EQ.0)GO TO 80
      SIGW=21.
      SIGU=21.
      SIGV=21.
 120  CONTINUE
```

```
      DO 130 I=6,8
      AORDER(1,I)=0
      AORDER(2,I)=C
      AORDER(3,I)=0
      AORDER(I,I)=I-5
  130 BORDER(I,I-5)=I-6
      PI=3.14159
      PIV=PI*V
      FBOPV=(4.*SPANF)/PIV
      WRITE(6,1OCC)LW,LU,SIGW,SIGU
 1000 FORMAT(3(/),2CX,*RESULTS OF GUST EQUATICNS*/20X,*LW =*,F8.2,* LU =
     1*,F8.2/2CX,*SIGMA W =*,F8.3,* SIGMA U =*,F8.3/)
      IF(INDMOD.NE.0)GO TO 14C
C     A(ROW,COLUMN,N) WHERE N=1 IS USED FCR CCNSTANT TERM
C                           N=2 IS USED FOR LAMBDA TERM
C                           N=3 IS USED FOR LAMBDA SQUARED TERM ETC.
      LWOV=LW/V
      A(1,6,1)=-XU
      A(2,6,1)=-ZU
      A(3,6,1)=-MU
      A(6,6,1)=1.
      A(6,6,2)=LU/V
      A(1,7,1)=-XA
      A(2,7,1)=-ZA
      A(3,7,1)=-MA
      A(7,7,1)=1.
      A(7,7,2)=2.*LWOV
      A(7,7,3)=LWOV**2
      A(1,8,1)=XAD-XQ
      A(2,8,1)=ZAD-ZQ
      A(3,8,1)=MAD-MQ
      A(8,8,1)=1.
      A(8,8,2)=(2.*LWOV)+FBOPV
      A(8,8,3)=LWOV**2+((2.*LWOV)*FBOPV)
      A(8,8,4)=LWOV**2*FBOPV
      B(6,1,1)=SIGU*SQRT(2.*LU/V)
      B(7,2,1)=(SIGW/V)*SQRT(LWOV)
      B(7,2,2)=1.732051*SQRT(LWOV)*LWOV*(SIGW/V)
      B(8,3,2)=B(7,2,1)
      B(8,3,3)=B(7,2,2)
      GO TO 999
  140 CONTINUE
      SRLWV=SQRT(LW*V)
      LVOV=LV/V
      SRLVOV=SQRT(LVOV)
      FBOPV=(3.*SPANF)/PIV
      A(6,6,1)=SRLWV
      A(6,6,2)=SRLWV*FBOPV
      A(7,7,1)=1.
      A(7,7,2)=2.*LVOV
      A(7,7,3)=LVOV**2
      A(8,8,1)=1.
```

```
      A(8,8,2)=2.*LVOV+TBOPV
      A(8,8,3)=LVOV**2+2.*LVOV*TBOPV
      A(8,8,4)=LVOV**2*TBOPV
      PART=SQRT(.8*((PI*LW)/(4.*SPANF))**.3333)
      B(6,1,1)=((SIGW*1.7725)/(SQRT(LW*V)))*PART
      B(7,2,1)=(SIGV/V)*SQRT(LVCV)
      B(7,2,2)=B(7,2,1)*1.73205*LVOV
      B(8,3,2)=-B(7,2,1)
      B(8,3,3)=-B(7,2,2)
  999 RETURN
      END
```

```
      OVERLAY(MCPILOT,3,0)
      PROGRAM MISER
C
C     MAIN FOR MISER--MULTITERM ISE REDUCTION
C
      EXTERNAL KLOCK,KLOCK1
      COMMON COMON(4400)
      COMMON / 5 / A(1599),B(1601)
      INTEGER AORDER(12,12,1),BORDER(12,12,1)
      INTEGER RUNLINK
      DIMENSION AP(5),ASKBJI(12,12,11),CNSTMN(5)
      DIMENSION CNSTMX(5),CUTOFF(5),DIRCOS(5),KKK(5),LLL(5)
      DIMENSION MORD(12,12),NIN(8),NOUT(8),P(5)
      DIMENSION TITLE(10)
      DIMENSION TFDEN(37),TFNUM(37)
      EQUIVALENCE (CNSTMN,COMON( 704))
      EQUIVALENCE (CNSTMX,COMON( 719))
      EQUIVALENCE (CUTOFF,COMON( 734))
      EQUIVALENCE (ILINK ,COMON( 748))
      EQUIVALENCE (DIRCOS,COMON( 764))
      EQUIVALENCE (P     ,COMON( 780))
      EQUIVALENCE (TOLISE,COMON( 805))
      EQUIVALENCE (PISE  ,COMON( 806))
      EQUIVALENCE (TFDEN ,COMON( 823))
      EQUIVALENCE (TFNUM ,COMON( 904))
      EQUIVALENCE (ASKBJI,COMON(1007))
      EQUIVALENCE (AORDER,COMON(    1))
      EQUIVALENCE (BORDER,COMON( 145))
      EQUIVALENCE (MORD  ,COMON( 305))
      EQUIVALENCE (NIN   ,COMON( 465))
      EQUIVALENCE (NOUT  ,COMON( 481))
      EQUIVALENCE (TITLE ,COMON( 795))
      EQUIVALENCE (NUMEQ ,COMON( 505))
      EQUIVALENCE (IEQ   ,COMON( 508))
      EQUIVALENCE (ISETOL,COMON( 509))
      EQUIVALENCE (ISETRM,COMON( 510))
      EQUIVALENCE (ISTABL,COMON( 511))
      EQUIVALENCE (NNNISE,COMON( 512))
      EQUIVALENCE (KOUNT ,COMON( 513))
      EQUIVALENCE (KNTISE,COMON( 514))
      EQUIVALENCE (KPARAM,COMON( 515))
      EQUIVALENCE (INDCLS,COMON(4317))
      EQUIVALENCE (NORDEN,COMON( 517))
      EQUIVALENCE (NORNUM,COMON( 518))
      EQUIVALENCE (KTIM  ,COMON(4318))
      MTIM=0
      CALL RUNLINK(KLOCK1,-10.)
  604 FORMAT(///,* YOUR STARTING POINT LEAVES SOMETHING TO BE DESIRED AS
     . THE ABOVE STATEMENT INDICATES*)
  605 FORMAT(*1ATTENTION, YOU HAVE CNSTMN(*,I2,*)=*,0PE10.4,* GREATER TH
     .AN CNSTMX(*,I2,*)=*,0PE10.4)
   10 KNTISE=0
```

```
      KOUNT=0
      IA=0
      DO 70 I=1,KPARAM
      IF(CNSTMN(I).LE.CNSTMX(I))GO TO 70
      WRITE(6,605)I,CNSTMN(I),I,CNSTMX(I)
      STOP 3
 70   CONTINUE
      CALL BLANK
      CALL WRITE1
C    CALL MINI TO CHECK FOR AND CREATE A STABLE STARTING CONFIGURATION.
      CALL MINI(F,P,KPARAM)
C    IF MINI FAILS, THE FAILURE WILL BE DISCOVERED IN ISE.
C
C     CALL ISE TO INITIALIZE PISE
C
      CALL ISE
C
C     GO TO 320 TO WRITE OUT DATA FOR FIRST ISE CALCULATIONS
C
      IF(ISTABL.GT.0)GO TO 320
      WRITE(6,604)
      STOP 1
C
C     CHECK TO SEE IF CNSTMN.LE.P.LE.CNSTMX
C
 150  DO 200 I=1,KPARAM
      KKK(I)=-10
      LLL(I)=-10
      IF(P(I).GT.CNSTMN(I))GO TO 175
      P(I)=CNSTMN(I)
      KKK(I)=10
      GO TO 200
 175  IF(P(I).LT.CNSTMX(I))GO TO 200
      P(I)=CNSTMX(I)
      LLL(I)=10
 200  CONTINUE
C
C     IF P WAS CONSTRAINED, CALL ISE BEFORE CALLING GRADI
C
      DO 225 I=1,KPARAM
      IF((KKK(I).LT.0).AND.(LLL(I).LT.0))GO TO 225
      CALL ISE
      GO TO 250
 225  CONTINUE
 250  CALL GRADI
C
C     IF P WAS CONSTRAINED, SET APPROPRIATE DIRCOS(*)'S TO 0.D0
C
      DO 300 I=1,KPARAM
      IF((KKK(I).GT.0).AND.(DIRCOS(I).GT.0.D0))DIRCOS(I)=0.D0
      IF(LLL(I).GT.0.AND.DIRCOS(I).LT.0.D0)DIRCOS(I)=0.D0
 300  AP(I)=P(I)
```

```
      AISE=PISE
      DO 310 I=1,KPARAM
      IF(DIRCOS(I).NE.C.00)GO TO 310
      CALL GRADII
      GO TO 315
  310 CONTINUE
C
C     MINIMIZE ALONG GRADIENT BY CALLING MINIMI
C
  315 CALL MINIMI
  320 NTIM=IRUNLNK(KLOCK,DUM)
      IF(LT.GT.0.AND.IA.GT.C)GO TO 321
      IF(KNTISE.EQ.1)GO TO 321
      IF(NTIM.CT.KTIM)GO TO 352
  321 CALL WROTE3
      DO 350 NNNISE=1,ISETRM
      MATCH=1
      IEC=NUMEO
      DO 325 I=1,IEQ
      DO 325 J=1,IEQ
      MORD(I,J)=AORDER(I,J,MATCH)
      DO 325 K=1,11
      NOW=NEW(I,J,K,MATCH)
  325 ASKBJI(I,J,K)=A(NOW)
      CALL SKBJI
      NORDEN=NORNUM
      NORNUM=NORNUM+1
      DO 330 I=1,NORNUM
  330 TFDEN(I)=TFNUM(I)
      DO 335 I=1,IEQ
      NOUT1=NOUT(NNNISE)
      NIN1=NIN(NNNISE)
      MORD(I,NCUT1)=BORDER(I,NIN1,MATCH)
      DO 335 K=1,11
      NOW=NEW(I,NIN1,K,MATCH)
  335 ASKBJI(I,NOUT1,K)=B(NOW)
      CALL SKBJI
  350 CALL WROTE4
  352 CONTINUE
      LT=0
      IF(KNTISE.EQ.1)GO TO 15C
C
C     IF PROGRAM STUCK, GO TO 3000
C
      IF(AISE.EQ.PISE)GO TO 300C
C
C     DECIDE WHETHER TO STOP, DEPENDING ON ISETOL
C
      IF(ISETOL.LT.0)GO TO 450
      DO 400 I=1,KPARAM
      IF( ABS(AP(I)-P(I)).GT.CUTOFF(I))GO TC 150
      IF(P(I).LT.CNSTMN(I))GO TO 150
```

```
      IF(P(I).GT.CNSTMX(I))GO TO 150
 400  CONTINUE
      IF(ISETOL.EQ.0)GO TO 475
 450  IF( ABS(AISE-PISE).GT.TOLISE)GO TO 150
 475  CONTINUE
      MTIM=MTIM+1
      IF(MTIM.LT.2)GO TO 321
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
 6000 FORMAT(1H ,20X,10A6)
      WRITE(6,603)
 603  FORMAT(1H ,21(/),50X,*AMEN BROTHER, YOU HAVE ARRIVED*,//,
     .                    50X,*----YOU HAVE HIT A NEW LOW----*)
      ILINK=4
      CALL OVRLAY
C
C     IF PROGRAM STUCK, MOVE P A SMALL DISTANCE
C
C
C     IA IS THE CUMULATIVE SUM OF TIMES THE PROGRAM IS STUCK
C
 3000 IF(IA.LT.5)GO TO 3002
      WRITE(6,3001)
 3001 FORMAT(20(/),9X,*THE SEARCH FOR THIS CASE HAS STOPPED SINCE THE PR
     .OGRAM HAS RETURNED TO THE SAME POINT FIVE TIMES (CUMULATIVE).*,//,
     .19X,*EITHER ACCEPT THE LAST POINT AS THE MINIMUM*,/,19X,*OR      ST
     .ART THE SEARCH IN A DIFFERENT REGION TO SEE IF IT APPROACHES THE S
     .AME POINT*,//,19X,*OR ELSE*,//,19X,*EITHER TIGHTEN THE TOLERANCE*,
     ./,19X,*OR      CHANGE DEL( ).*)
      ILINK=4
      CALL OVRLAY
 3002 MATCH=0
      LT=1
      J=0
      K=0
      DO 3010 I=1,KPARAM
      MATCH=MATCH+1
      IF(KKK(I).GT.0)J=J+1
      IF(LLL(I).GT.0)K=K+1
      IF(J.NE.MATCH.AND.K.NE.MATCH)GO TO 3015
 3010 CONTINUE
      WRITE(6,602)
 602  FORMAT(1H1,21(/),44X,*THE CONSTRAINTS HAVE STOPPED FURTHER SEARCHI
     .NG*)
      STOP 3
 3015 DO 3025 I=1,KPARAM
 3025 P(I)=P(I)-DIRCOS(I)*.00100
      WRITE(6,3040)
 3040 FORMAT(*OOLD ISE EQUALED NEW ISE, PROGRAM STUCK, NEW P ATTEMPTED*)
      KOUNT=KOUNT+1
      CALL ISE
      IF(ISTABL.LT.0)GO TO 3100
```

```
      CALL WROTF3
      IA=IA+1
      GO TO 150
 3101 FORMAT(//,* MISER IS UNABLE TO CONTINUE SINCE THE NEW POINT IT CAL
     .CULATED IS IN AN UNSTABLE REGION*)
 3100 WRITE(6,3101)
      STOP 3101
      END
```

```
      SUBROUTINE WRITE1
C
C     THIS SUBROUTINE PRINTS ALL OUTPUT EXCEPT DIAGNOSTIC MESSAGES
C
C
C
      COMMON COMON(4400)
      COMMON / 6 / SAVE(310)
      COMMON / 5 / A(1599),B(1600)
      INTEGER AORDER(12,12,1),BORDER(12,12,1),NO(15)
      DIMENSION DUMMY(12)
      REAL    JUMP,MODELD(8,11),MODELN(8,11)
      DIMENSION AISE(8),ADDPR2(37),ANDPR2(37)
      DIMENSION CNSTMN(5),CNSTMX(5),CUTOFF(5),DEL(5),DIRCOS(5)
      DIMENSION MODDOR(8),MODNOR(8),NIN(8),NMOD(8),NOUT(8),NSYST(8)
      DIMENSION NTYPE(8),P(5),SCLSYS(8)
      DIMENSION TFDEN(37),TFNUM(37),TITLE(10),WEIGHT(8),WORK(115)
      DIMENSION ROOTNR(36),ROOTNI(36),ROOTDR(36),ROOTDI(36),RMULT(37)
      EQUIVALENCE (MODELD,COMON( 519))
      EQUIVALENCE (MODELN,COMON( 607))
      EQUIVALENCE (AISE  ,COMON( 695))
      EQUIVALENCE (AJP   ,COMON( 703))
      EQUIVALENCE (CNSTMN,COMON( 704))
      EQUIVALENCE (CNSTMX,COMON( 719))
      EQUIVALENCE (INDCLS,COMON(4317))
      EQUIVALENCE (SAVEN ,SAVE(15))
      EQUIVALENCE (IORDN ,SAVE( 1))
      EQUIVALENCE (CUTOFF,COMON( 734))
      EQUIVALENCE (DEL   ,COMON( 749))
      EQUIVALENCE (DIRCOS,COMON( 764))
      EQUIVALENCE (JUMP  ,COMON( 779))
      EQUIVALENCE (P     ,COMON( 780))
      EQUIVALENCE (TITLE ,COMON( 795))
      EQUIVALENCE (TOLISE,COMON( 805))
      EQUIVALENCE (TOTISE,COMON( 806))
      EQUIVALENCE (WEIGHT,COMON( 807))
      EQUIVALENCE (SCLSYS,COMON( 815))
      EQUIVALENCE (TFDEN ,COMON( 823))
      EQUIVALENCE (TFNUM ,COMON( 904))
      EQUIVALENCE (ADDPR2,COMON( 985))
      EQUIVALENCE (ANDPR2,COMON(1022))
      EQUIVALENCE (ROOTDR,COMON(1059))
      EQUIVALENCE (ROOTDI,COMON(1095))
      EQUIVALENCE (ROOTNR,COMON(1131))
      EQUIVALENCE (ROOTNI,COMON(1167))
      EQUIVALENCE (WORK  ,COMON(1203))
      EQUIVALENCE (AORDER,COMON(   1))
      EQUIVALENCE (BORDER,COMON( 145))
      EQUIVALENCE (MODDOR,COMON( 289))
      EQUIVALENCE (MODNOR,COMON( 297))
      EQUIVALENCE (NIN   ,COMON( 465))
      EQUIVALENCE (NMOD  ,COMON( 473))
```

```
      EQUIVALENCE (NOUT  ,COMON( 481))
      EQUIVALENCE (NSYST ,COMON( 489))
      EQUIVALENCE (NTYPE ,COMON( 497))
      EQUIVALENCE (NUMEQ ,COMON( 505))
      EQUIVALENCE (ISETOL,COMON( 509))
      EQUIVALENCE (ISETRM,COMON( 510))
      EQUIVALENCE (NNNISE,COMON( 512))
      EQUIVALENCE (KOUNT ,COMON( 513))
      EQUIVALENCE (KNTISE,COMON( 514))
      EQUIVALENCE (KPARAM,COMON( 515))
      EQUIVALENCE (METHOD,COMON( 516))
      EQUIVALENCE (NORDEN,COMON( 517))
      EQUIVALENCE (NORNUM,COMON( 518))
      DIMENSION SAVEN(8,37),IORDN(8)
      DATA NO/1,2,3,4,5,6,7,8,9,10,11,12,13,14,15/
      AJP=0.00
      DO 5 I=1,8
      AISE(I)=0.00
5     SCLSYS(I)=0.00
      DO 10 I=1,5
10    DIRCOS(I)=0.00
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
      WRITE(6,6015)
6015  FORMAT(1H ,21(/),55X,*BEGIN CHECK OF DATA*,///,
     .                    54X,*DATA IS FOR FIRST ISE*)
      L=1
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
      WRITE(6,6007)
6007  FORMAT(1H+,110X,*A MATRIX*)
      DO 30 I=1,12
      IF(I.EQ.5.OR.I.EQ.9)CALL CLASS(INDCLS)
      IF(I.EQ.5.OR.I.EQ.9)WRITE(6,6000)TITLE
      WRITE(6,707) (AORDER(I,J,L),J=1,12)
 707  FORMAT(/,1X,12(4X,I2,5X))
      DO 30 K=1,11
      M=K-1
      DO 300 J=1,12
      NOW=NEW(I,J,K,L)
300   DUMMY(J)=A(NOW)
  30  WRITE(6,747) (DUMMY(J),J=1,12)
747   FORMAT(1X,12(1PE11.4))
6010  FORMAT(1X,I3,3X,8(1PE12.4,2X))
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
      WRITE(6,6020)
6020  FORMAT(1H+,110X,*B MATRIX*)
      DO 31 I=1,12
      IF(I.EQ.5.OR.I.EQ.9)CALL CLASS(INDCLS)
      IF(I.EQ.5.OR.I.EQ.9)WRITE(6,6000)TITLE
      WRITE(6,707) (BORDER(I,J,L),J=1,12)
```

```
      DO 31 K=1,11
      M=K-1
      DO 310 J=1,12
      NOW=NEW(I,J,K,L)
310   DUMMY(J)=B(NOW)
   31 WRITE(6,747) (DUMMY(J),J=1,12)
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
      WRITE(6,6011)
6011  FORMAT(1H0,8X,*MODEL 1*,7X,*MODEL 2*,7X*MODEL 3*,7X,*MODEL 4*,7X,*
     .MODEL 5*,7X,*MODEL 6*,7X,*MODEL 7*,7X,*MODEL 8*,/,7X,8(1X,*NUMERAT
     .OR*,4X))
      WRITE(6,6024)(MODNOR(N),N=1,8)
6024  FORMAT(6X,8(2X,*ORDER = *,I2,2X))
      DO 32 K=1,11
      M=K-1
32    WRITE(6,6010)M,(MODELN(N,K),N=1,8)
      WRITE(6,6012)
6012  FORMAT(//,8X,8(*DENOMINATOR*,3X))
      WRITE(6,6024)(MODDOR(N),N=1,8)
      DO 33 K=1,11
      M=K-1
33    WRITE(6,6010)M,(MODELD(N,K),N=1,8)
      WRITE(6,6013)ISETRM
6013  FORMAT(////,45X,I1,* ISE TERMS WILL BE CALCULATED FOR EACH ISE*//,
     .40X,*NIN*,5X,*NMOD*,4X,*NOUT*,3X,*NSYST*,3X,*NTYPE*,9X,*WEIGHT*,/)
      WRITE(6,6014)(NIN(N),NMOD(N),NOUT(N),NSYST(N),NTYPE(N),WEIGHT(N),N
     .              =1,ISETRM)
6014  FORMAT((38X,5(2X,I2,4X),4X,1PE11.3))
      WRITE(6,6037)KPARAM
6037  FORMAT(2(/),45X,I2,* PARAMETERS WILL BE VARIED TO OPTIMIZE ISE*)
      WRITE(6,6038)(CNSTMN(I),CNSTMX(I),CUTOFF(I),DEL(I),P(I),I=1,KPARAM
     .)
6038  FORMAT(/,44X,*CNSTMN*,4X,*CNSTMX*,4X,*CUTOFF*,5X,*DEL*,8X,*P*,/,
     .       (42X,5(1PE10.3)))
      WRITE(6,6030)NUMEQ,NUMEQ
6030  FORMAT(/,* SYSTEM 1 IS A *,I2,* X *,I2,* MATRIX *)
      WRITE(6,6033)ISETOL
6033  FORMAT(*0ISETOL = *,I2,* THEREFORE THE PROGRAM WILL STOP WHEN*)
      IF(ISETOL.LT.0)WRITE(6,6034)TOLISE
6034  FORMAT(1H+,49X,*ISE IS WITHIN TOLISE = *,1PE10.3)
      IF(ISETOL.EQ.0)WRITE(6,6035)
6035  FORMAT(1H+,49X,*EACH P IS WITHIN ITS RESPECTIVE CUTOFF*)
      IF(ISETOL.GT.0)WRITE(6,6036)TOLISE
6036  FORMAT(1H+,49X,*EACH P IS WITHIN ITS RESPECTIVE CUTOFF AND ISE IS
      WITHIN TOLISE = *,1PE10.3)
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
      WRITE(6,6019)
6019  FORMAT(1H ,22(/),56X,*END CHECK OF DATA*)
      RETURN
C
```

```
      ENTRY WROTE3
C
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
 6000 FORMAT(1H ,20X,10A6)
      WRITE(6,6041)KOUNT,KNTISE,JUMP,(NO(2*I-1),AISE(2*I-1),NO(2*I),
     .              AISE(2*I),NO(2*I-1),SCLSYS(2*I-1),NO(2*I),SCLSYS(2*I)
     .              ,I=1,3),TOTISE,AJP,NO(7),AISE(7),NO(8),AISE(8),NO(7),
     .              SCLSYS(7),NO(8),SCLSYS(8)
 6041 FORMAT(//,6X,*KOUNT*,14X,*JUMP*,25X,*ISE TERMS*,33X,*SCALE*,/,5X,
     .I2,*-*,I4,10X,1PE10.3,9X,2(I1,1X,1PE14.6,3X),6X,2(I1,1X,1PE10.3,3X
     .),/,41X,2(I1,1X,1PE14.6,3X),6X,2(I1,1X,1PE10.3,3X),/,6X,*TOTISE*,
     .13X,*AJP*,13X,2(I1,1X,1PE14.6,3X),6X,2(I1,1X,1PE10.3,3X),/,1X,
     .1PE16.8,5X,1PE10.3,9X,2(I1,1X,1PE14.6,3X),6X,2(I1,1X,1PE10.3,3X))
      WRITE(6,6042)
 6042 FORMAT(2(/),30X,*P*,69X,*DIRCOS*)
      J=KPARAM/3
      L=3*J
      IF(J.EQ.0)GO TO 36
      WRITE(6,6043)(NO(I),P(I),NO(I+1),P(I+1),NO(I+2),P(I+2),NO(I),
     .      DIRCOS(I),NO(I+1),DIRCOS(I+1),NO(I+2),DIRCOS(I+2),I=1,L,3)
 6043 FORMAT((1X,3(I2,1X,1PE14.6,3X),12X,3(I2,1X,1PE14.6,3X)))
 36   K=KPARAM-L
      IF(K.EQ.1)WRITE(6,6044)NO(L+1),P(L+1),NO(L+1),DIRCOS(L+1)
 6044 FORMAT(21X,I2,1X,1PE14.6,55X,I2,1X,1PE14.6)
      IF(K.EQ.2)WRITE(6,6045)NO(L+1),P(L+1),NO(L+2),P(L+2),NO(L+1),
     .            DIRCOS(L+1),NO(L+2),DIRCOS(L+2)
 6045 FORMAT(11X,2(I2,1X,1PE14.6,3X),32X,2(I2,1X,1PE14.6,3X))
      RETURN
C
      ENTRY WROTE4
C
      L=NORNUM+1
      DO 41 I=1,L
      J=1+L-I
 41   ANDPR2(I)=TFNUM(J)
      TOL1=.0001
      CALL POLYRT(NORNUM,ANDPR2,TOL1,ROOTNR,ROOTNI,RMULT,NRR,NERR)
      IF(NERR.EQ.1)GO TO 42
      IF(NERR.EQ.2)WRITE(6,6070)NERR
 6070 FORMAT(5(/),* NERR = *,I2,* UNABLE TO SOLVE FOR NUMERATOR ROOTS AF
     .TER 500 ITERATIONS*//)
      IF(NERR.EQ.3)WRITE(6,6071)NERR
 6071 FORMAT(5(/),* NERR = *,I2,* ORDER OF NUMERATOR GREATER THAN 99*//)
 42   L=NORDEN+1
      DO 43 I=1,L
      J=1+L-I
 43   ADDPR2(I)=TFDEN(J)
      CALL POLYRT(NORDEN,ADDPR2,TOL1,ROOTDR,ROOTDI,RMULT,NRR,NERR)
      IF(NERR.EQ.1)GO TO 50
      IF(NERR.EQ.2)WRITE(6,6072)NERR
 6072 FORMAT(5(/),* NERR = *,I2,* UNABLE TO SOLVE FOR DENOMINATOR ROOTS
```

```
     .AFTER 50C ITERATIONS*//)
      IF(NERR.EQ.3) WRITE(6,6C73)NERR
6073  FORMAT(5(/),* NERR = *,I2,* ORDER OF DENOMINATCR GREATER THAN 99*)
50    IF(NNNISE.EQ.3.OR.NNNISE.EQ.5.CR.NNNISE.EQ.7)CALL CLASS(INDCLS)
      IF(NNNISE.EQ.3.OR.NNNISE.EQ.5.OR.NNNISE.EQ.7)WRITE(6,6000)TITLE
      WRITE(6,60C2)NNNISE
6002  FORMAT(2(/),56X,* FOR ISE TERM*,I2,/,11X,*SYSTEM NUMERATOR*,10X,*S
     .YSTEM DENOMINATOR*,21X,*NUMERATOR ROOTS*,17X,*DENCMINATCR ROOTS*)
      WRITE(6,6005)NORNUM,NORDEN
6005  FORMAT(2(14X,*ORDER = *,I2,3X))
      NCOEFN=NORNUM + 1
      IORDN(NNNISE)=NORNUM
      DO 55 I=1,NCOEFN
55    SAVEN(NNNISE,I)=TFNUM(I)
      I=0
      WRITE(6,6CC4)TFNUM(1),I,TFDEN(1),I
6004  FORMAT(2X,2(7X,1PE13.6,2X,3HS**,I2),7X,2(8X,1PE11.4,3X,1PE11.4))
      NERR=MINC(NORNUM,NORDEN)+1
      DO 57 K=2,NERR
      I=K-1
57    WRITE(6,6CC4)TFNUM(K),I,TFDEN(K),I,ROOTNR(I),ROOTNI(I),ROOTDR(I),
                   ROOTDI(I)
      J=NERR+1
      IF(NORDEN.GT.NORNUM)GO TO 80
      IF(NORDEN.EQ.NORNUM)RETURN
      NERR=NORNUM+1
      DO 65 K=J,NERR
      I=K-1
65    WRITE(6,6CC1)TFNUM(K),I,ROOTNR(I),ROOTNI(I)
6001  FORMAT(9X,1PE13.6,2X,3HS**,I2,42X,1PE11.4,3X,1PE11.4)
      RETURN
80    NERR=NORDEN+1
      DO 85 K=J,NERR
      I=K-1
85    WRITE(6,60C3)TFDEN(K),I,ROOTDR(I),ROOTDI(I)
6003  FORMAT(36X,1PE13.6,2X,3HS**,I2,48X,1PE11.4,3X,1PE11.4)
      RETURN
      END
```

```
      SUBROUTINE MINIMI
C
C     THIS SUBROUTINE FINDS BY GOLDEN SECTION THE POINT P AT WHICH ISE I
C     MINIMUM ALONG  A RAY WHICH EMANATES FRCM THE POINT P-START AND WHI
C     FOLLOWS THE DIRECTION DEFINED BY MINUS DIRCOS(I)
C
C
C     THE INPUTS  TO THIS SUBROUTINE ARE CUTOFF,DIRCOS,ISTABLE,JUMP,KOUN
C                                        KPARAM,P,PISE
C     THE OUTPUTS OF THIS SUBROUTINE ARE AJP,IWHICH,JUMP,KOUNT,P,PISE
C
C
C     DEFINITIONS
C
C             KOUNT IS A COUNTER, IT IS INCREASED BY 1 BY THIS SUBPROGR
C
      COMMON COMON(4400)
      REAL    JUMP
      DIMENSION A(5),C(5),CNSTMN(5),CUTOFF(5),DIRCOS(5)
      DIMENSION E(5),G(5),P(5),STEP(5)
      DIMENSION CNSTMX(5)
      EQUIVALENCE (AJP   ,COMON( 703))
      EQUIVALENCE (CNSTMN,COMON( 704))
      EQUIVALENCE (CNSTMX,COMON( 719))
      EQUIVALENCE (CUTOFF,COMON( 734))
      EQUIVALENCE (DIRCOS,COMON( 764))
      EQUIVALENCE (JUMP  ,COMON( 779))
      EQUIVALENCE (P     ,COMON( 780))
      EQUIVALENCE (TOLISE,COMON( 805))
      EQUIVALENCE (PISE  ,COMON( 806))
      EQUIVALENCE (ISETOL,COMON( 509))
      EQUIVALENCE (ISTABL,COMON( 511))
      EQUIVALENCE (KOUNT ,COMON( 513))
      EQUIVALENCE (KPARAM,COMON( 515))
      KOUNT=KOUNT+1
C
C     BEGIN DETERMINING END POINTS--(A,G) OF INTERVAL
C
      DO 25 I=1,KPARAM
 25   A(I)=P(I)
      AISE=PISE
      CISE=1.E10
      EISE=1.E10
      IC=100
      IE=100
      LL=0
 50   DO 75 I=1,KPARAM
 75   STEP(I)=-DIRCOS(I)*JUMP
 100  DO 125 I=1,KPARAM
 125  P(I)=A(I)+STEP(I)
      CALL ISE
      IF(ISTABL.GT.0)GO TO 150
```

```
      JUMP=JUMP*.500
      GO TO 50
C
C     CHECK IF G IS WITHIN CONSTRAINSTS
C
  150 DO 155 I=1,KPARAM
      IF(P(I).GE.CNSTMN(I).AND.P(I).LE.CNSTMX(I))GO TO 155
      RETURN
  155 CONTINUE
      LL=LL+1
      IF(PISE.GT.AISE)GO TO 175
      DO 160 I=1,KPARAM
      C(I)=A(I)
  160 A(I)=P(I)
      CISE=AISE
      AISE=PISE
      SAVE=JUMP
      GO TO 100
  175 IF(LL.LT.2)GO TO 190
      DO 180 I=1,KPARAM
      A(I)=C(I)
  180 G(I)=P(I)
      AISF=CISE
      GISE=PISE
      AJP=SAVE+JUMP
      JUMP=AJP
      GO TO 200
  190 AJP=JUMP
      DO 195 I=1,KPARAM
  195 G(I)=P(I)
      GISE=PISE
      JUMP=JUMP*.500
C
C     END DETERMINING END POINTS--(A,G) OF INTERVAL
C
  200 IF(ISETOL.LT.C)GO TO 206
      DO 205 I=1,KPARAM
      IF( ABS(A(I)-G(I)).GT.CUTOFF(I))GO TO 208
  205 CONTINUE
      IF(ISETOL.EQ.C)GO TO 300
  206 IF( ABS(AISE-GISE).GT.TOLISE)GO TO 208
      GO TO 300
C
C     BEGIN GOLDEN SECTION
C
  208 AJP=AJP*.618033988700
      DO 210 I=1,KPARAM
  210 STEP(I)=-DIRCOS(I)*AJP
      IF(IC.LT.C)GO TO 225
      DO 215 I=1,KPARAM
  215 P(I)=G(I)-STEP(I)
      CALL ISE
```

```
      DO 220 I=1,KPARAM
  220 C(I)=P(I)
      CISE=PISE
  225 IF(IE.LT.0)GO TO 230
      DO 226 I=1,KPARAM
  226 P(I)=A(I)+STEP(I)
      CALL ISE
      DO 228 I=1,KPARAM
  228 E(I)=P(I)
      EISE=PISE
  230 IF(CISE.LT.EISE)GO TO 250
      DO 240 I=1,KPARAM
      A(I)=C(I)
  240 C(I)=E(I)
      AISE=CISE
      CISE=EISE
      IC=-100
      IE=+100
      GO TO 200
  250 DO 260 I=1,KPARAM
      G(I)=E(I)
  260 E(I)=C(I)
      GISE=EISE
      EISE=CISE
      IC=+100
      IE=-100
      GO TO 200
C
C     END GOLDEN SECTION
C
C     DETERMINE SMALLEST ISE BEFORE RETURNING
C
  300 IF(EISE.LT.AISE)RETURN
      IF(AISE.LT.GISE)GO TO 350
      DO 310 I=1,KPARAM
  310 P(I)=G(I)
      PISE=GISE
      RETURN
  350 DO 360 I=1,KPARAM
  360 P(I)=A(I)
      PISE=AISE
      RETURN
      END
```

```
      SUBROUTINE GRADI
C
C     THIS PROGRAM CALCULATES THE GRADIENT OF A POINT P
C
C     THE GRADIENT IS COMPUTED ONLY ON ONE SIDE OF P.  IF DEL(I) IS TOO
C     THE GRADI SUBROUTINE MIGHT NOT ONLY COMPUTE A POOR APPROXIMATION T
C     TRUE GRADIENT AT P BUT ALSO COMPUTE A GRADIENT OF THE WRONG SIGN (
C     IS NEARER THE MINIMUM THAN DEL(I)).  FOR THIS REASON, KEEP DEL(I)
C
C     THE INPUTS  TO THIS SUBROUTINE ARE DEL,ISTABL,KPARAM,P,PISE
C     THE OUTPUTS OF THIS SUBROUTINE ARE DIRCOS
C
C         DEFINITIONS
C
C              DEL(I) IS THE DISTANCE P WILL BE INCREMENTED IN THE I-TH
C                   DIMENSION TO CALCULATE DIRCOS IN THE I-TH DIMENSION
C              DIRCOS(I) IS THE DIRECTION COSINE AT P IN THE I-TH DIMENS
C              P(I) IS THE VALUE OF I-TH COORDINATE OF P
C              PISE IS THE VALUE OF ISE AT P
C
      COMMON COMON(4400)
      DIMENSION A(5),DEL(5),DIRCOS(5),P(5)
      EQUIVALENCE (DEL    ,COMON( 749))
      EQUIVALENCE (DIRCOS,COMON( 764))
      EQUIVALENCE (P      ,COMON( 780))
      EQUIVALENCE (PISE   ,COMON( 806))
      EQUIVALENCE (ISTABL,COMON( 511))
      EQUIVALENCE (KPARAM,COMON( 515))
      EQUIVALENCE(TISE,DENOM)
      DO 5 I=1,KPARAM
  5   A(I)=P(I)
      AISE=PISE
      DO 15 I=1,KPARAM
      P(I)=A(I)+DEL(I)
      CALL ISE
      IF(ISTABL.GT.0)GO TO 10
      P(I)=A(I)-DEL(I)
      CALL ISE
      DIRCOS(I)=(AISE-PISE)/DEL(I)
      GO TO 15
 10   DIRCOS(I)=(PISE-AISE)/DEL(I)
 15   P(I)=A(I)
      PISE=AISE
      ENTRY GRADII
      DENOM=0.D0
      DO 25 I=1,KPARAM
 25   DENOM=DENOM+DIRCOS(I)*DIRCOS(I)
      DENOM= SQRT(DENOM)
      IF(DENOM.EQ.0.)WRITE(6,50)
 50   FORMAT(3(/),20X,*CONSTRAINED MINIMUM  (DIRECTION COSINES EQUAL ZER
     1O)*)
      DENOM=1.D0/DENOM
```

```
      DO 30 I=1,KPARAM
30    DIRCOS(I)=DIRCOS(I)*DENOM
      RETURN
      END
```

```
      SUBROUTINE SKBJI
C
C     THIS SUBROUTINE FINDS THE DETERMINANT OF A MATRIX WHOSE ELEMENTS
C                  ARE POLYNOMIALS IN S
C
      COMMON COMON(44CC)
      EQUIVALENCE (ANS    ,COMON( 504))
      EQUIVALENCE (TEMP   ,COMCN(1CO7))
      EQUIVALENCE (A      ,COMON(2591))
      EQUIVALENCE (MORD   ,COMCN( 305))
      EQUIVALENCE (N      ,COMCN( 508))
      EQUIVALENCE (NA     ,COMCN( 518))
C
C
      DIMENSION IDO(24)
      DIMENSION SUBPR2(21),SUBPR3(31),SUBPR4(41),SUBPR5(51),SUBPR6(61)
     .    .     SUBPR7(71),SUBPR8(81),SUBPR9(91),SUBP10.101),SUBP11(111)
     .    .     SUBP12(121)
      DIMENSION NORD(12,12),A(12,12,11),ANS(37)
      DIMENSION MORD(12,12),TEMP(12,12,11)
      EQUIVALENCE(IDO( 1),  I1GO),(IDO( 2),I1STOP)
     .    .     (IDO( 3),  J1GO),(IDO( 4),J1STOP)
     .    .     (IDO( 5),  K1GO),(IDC( 6),K1STOP)
     .    .     (IDO( 7),  L1GO),(IDO( 8),L1STOP)
     .    .     (IDO( 9),  M1GO),(IDO(10),M1STOP)
     .    .     (IDO(11),  I2GC),(IDC(12),I2STOP)
     .    .     (IDO(13),  J2GO),(IDO(14),J2STCP)
     .    .     (IDO(15),  K2GO),(IDO(16),K2STOP)
     .    .     (IDO(17),  L2GC),(IDO(18),L2STOP)
     .    .     (IDO(19),  M2GO),(IDO(20),M2STOP)
     .    .     (IDO(21),  I3GC),(IDO(22),I3STOP)
     .    .     (IDO(23),  J3GO),(IDO(24),J3STCP)
C
C     IT IS REALIZED THAT PROGRAMMING CAN BE CONDENSED BY USING SUBSCRIP
C     VARIABLES IN THIS PROGRAM, BUT USE HAS SHOWN THE NON-ELEGANT SCHEM
C     HERE IS FASTER--AND IN THIS PROGRAM TIME IS THE CRITERION
C
      NX=13-N
      DO 11 I=1,12
      CO 11 J=1,12
      NORD(I,J)=-1
      DO 11 K=1,11
   11 A(I,J,K) = 0
      DO 12 I=1,12
      NORD(I,I)=C
   12 A(I,I,1) = 1.
      II = 0
      DO 13 I=NX,12
      II=II+1
      JJ=0
      DO 13 J=NX,12
      JJ=JJ+1
```

```
      NORD(I,J)=MORD(II,JJ)
      DO 13 K=1,11
   13 A(I,J,K)=TEMP(II,JJ,K)
C
C     SET UP LIMITS ON DO LOOPS
C
      DO 44 I=1,12
      DO 43 J=1,12
      IF(NORD(I,J).EQ.-1)GO TO 43
      IDO(2*I-1)=J
      GO TO 44
 43   CONTINUE
 44   CONTINUE
      DO 48 I=1,12
      DO 47 J=1,12
      NINEJ=13-J
      IF(NORD(I,NINEJ).EQ.-1)GO TO 47
      IDO(2*I)=NINEJ
      GO TO 48
 47   CONTINUE
 48   CONTINUE
C
C   ********************************************************************
C   *  ZEROES THE ANSWER AND THE ORDER OF THE ANSWER.                  *
C   ********************************************************************
C
      NA=0
      DO 60 I=1,37
   60 ANS(I) = 0
C
C     FIND A ROW AND A COLUMN COMBINATION
C
      KFLAG=2
      DO 156 I1=I1GO,I1STOP
      IF(NORD(1,I1).EQ.-1)GO TO 156
      DO 155 J1=J1GO,J1STOP
      IF(NORD(2,J1).EQ.-1)GO TO 155
      IF(J1.EQ.I1)GO TO 155
      DO 154 K1=K1GO,K1STOP
      IF(NORD(3,K1).EQ.-1)GO TO 154
      IF(K1.EQ.I1.OR.K1.EQ.J1)GO TO 154
      DO 153 L1=L1GO,L1STOP
      IF(NORD(4,L1).EQ.-1)GO TO 153
      IF(L1.EQ.I1.OR.L1.EQ.J1.OR.L1.EQ.K1)GO TO 153
      DO 152 M1=M1GO,M1STOP
      IF(NORD(5,M1).EQ.-1)GO TO 152
      IF(M1.EQ.I1.OR.M1.EQ.J1.OR.M1.EQ.K1.OR.M1.EQ.L1)GO TO 152
      DO 151 I2=I2GO,I2STOP
      IF(NORD(6,I2).EQ.-1)GO TO 151
      IF(I2.EQ.I1.OR.I2.EQ.J1.OR.I2.EQ.K1.OR.I2.EQ.L1.OR.I2.EQ.M1)
     .GO TO 151
      DO 150 J2=J2GO,J2STOP
```

```
      IF(NORD(7,J2).EQ.-1)GO TO 150
      IF(J2.EQ.I1.OR.J2.EQ.J1.OR.J2.EQ.K1.OR.J2.EQ.L1.OR.J2.EQ.M1.OR.
     .   J2.EQ.I2)GO TO 150
      DO 149 K2=K2GO,K2STOP
      IF(NORD(8,K2).EQ.-1)GO TO 149
      IF(K2.EQ.I1.OR.K2.EQ.J1.OR.K2.EQ.K1.OR.K2.EQ.L1.OR.K2.EQ.M1.OR.
     .   K2.EQ.I2.OR.K2.EQ.J2)GO TO 149
      DO 148 L2=L2GO,L2STOP
      IF(NORD(9,L2).EQ.-1)GO TO 148
      IF(L2.EQ.I1.OR.L2.EQ.J1.OR.L2.EQ.K1.OR.L2.EQ.L1.OR.L2.EQ.M1.OR.
     .   L2.EQ.I2.OR.L2.EQ.J2.OR.L2.EQ.K2)GO TO 148
      DO 147 M2=M2GO,M2STOP
      IF(NORD(10,M2).EQ.-1)GO TO 147
      IF(M2.EQ.I1.OR.M2.EQ.J1.OR.M2.EQ.K1.OR.M2.EQ.L1.OR.M2.EQ.M1.OR.
     .   M2.EQ.I2.OR.M2.EQ.J2.OR.M2.EQ.K2.OR.M2.EQ.L2)GO TO 147
      DO 146 I3=I3GO,I3STOP
      IF(NORD(11,I3).EQ.-1)GO TO 146
      IF(I3.EQ.I1.OR.I3.EQ.J1.OR.I3.EQ.K1.OR.I3.EQ.L1.OR.I3.EQ.M1.OR.
     .   I3.EQ.I2.OR.I3.EQ.J2.OR.I3.EQ.K2.OR.I3.EQ.L2.OR.I3.EQ.M2)
     .   GO TO 146
      DO 145 J3=J3GO,J3STOP
      IF(NORD(12,J3).EQ.-1)GO TO 145
      IF(J3.EQ.I1.OR.J3.EQ.J1.OR.J3.EQ.K1.OR.J3.EQ.L1.OR.J3.EQ.M1.OR.
     .   J3.EQ.I2.OR.J3.EQ.J2.OR.J3.EQ.K2.OR.J3.EQ.L2.OR.J3.EQ.M2.OR.
     .   J3.EQ.I3)GO TO 145
C
C     CALCULATE SUBORDERS IN JORD* AND SUBPRODUCTS IN SUBPR*
C
C     KFLAG SIGNALS WHERE TO START CALCULATING SUBPRODUCTS
C
      GO TO (85,85,89,93,97,101,105,109,113,117,121,125),KFLAG
 85   KKK=NORD(1,I1)+1
      LLL=NORD(2,J1)+1
      JORD2=KKK+LLL-2
      INDT=JORD2+1
      DO 86 IND=1,INDT
   86 SUBPR2(IND) = 0
      DO 87 KKKK=1,KKK
      DO 87 LLLL=1,LLL
      IND=KKKK+LLLL-1
 87   SUBPR2(IND)=SUBPR2(IND)+A(1,I1,KKKK)*A(2,J1,LLLL)
      J1SIGN=1
      IF(J1.LT.I1)J1SIGN=-J1SIGN
 89   KKK=JORD2+1
      LLL=NORD(3,K1)+1
      JORD3=KKK+LLL-2
      INDT=JORD3+1
      DO 90 IND=1,INDT
 90   SUBPR3(IND)=0
      DO 91 KKKK=1,KKK
      DO 91 LLLL=1,LLL
      IND=KKKK+LLLL-1
```

```
91    SUBPR3(IND)=SUBPR3(IND)+SUBPR2(KKKK)*A(3,K1,LLLL)
      K1SIGN=J1SIGN
      IF(K1.LT.I1)K1SIGN=-K1SIGN
      IF(K1.LT.J1)K1SIGN=-K1SIGN
93    KKK=JORD3+1
      LLL=NORD(4,L1)+1
      JORD4=KKK+LLL-2
      INDT=JORD4+1
      DO 94 IND=1,INDT
94    SUBPR4(IND)=0
      DO 95 KKKK=1,KKK
      DO 95 LLLL=1,LLL
      IND=KKKK+LLLL-1
95    SUBPR4(IND)=SUBPR4(IND)+SUBPR3(KKKK)*A(4,L1,LLLL)
      L1SIGN=K1SIGN
      IF(L1.LT.I1)L1SIGN=-L1SIGN
      IF(L1.LT.J1)L1SIGN=-L1SIGN
      IF(L1.LT.K1)L1SIGN=-L1SIGN
97    KKK=JORD4+1
      LLL=NORD(5,M1)+1
      JORD5=KKK+LLL-2
      INDT=JORD5+1
      DO 98 IND=1,INDT
98    SUBPR5(IND)=0
      DO 99 KKKK=1,KKK
      DO 99 LLLL=1,LLL
      IND=KKKK+LLLL-1
99    SUBPR5(IND)=SUBPR5(IND)+SUBPR4(KKKK)*A(5,M1,LLLL)
      M1SIGN=L1SIGN
      IF(M1.LT.I1)M1SIGN=-M1SIGN
      IF(M1.LT.J1)M1SIGN=-M1SIGN
      IF(M1.LT.K1)M1SIGN=-M1SIGN
      IF(M1.LT.L1)M1SIGN=-M1SIGN
101   KKK=JORD5+1
      LLL=NORD(6,I2)+1
      JORD6=KKK+LLL-2
      INDT=JORD6+1
      DO 102 IND=1,INDT
 102  SUBPR6(IND) = 0
      DO 103 KKKK=1,KKK
      DO 103 LLLL=1,LLL
      IND=KKKK+LLLL-1
103   SUBPR6(IND)=SUBPR6(IND)+SUBPR5(KKKK)*A(6,I2,LLLL)
      I2SIGN=M1SIGN
      IF(I2.LT.I1)I2SIGN=-I2SIGN
      IF(I2.LT.J1)I2SIGN=-I2SIGN
      IF(I2.LT.K1)I2SIGN=-I2SIGN
      IF(I2.LT.L1)I2SIGN=-I2SIGN
      IF(I2.LT.M1)I2SIGN=-I2SIGN
105   KKK=JORD6+1
      LLL=NORD(7,J2)+1
      JORD7=KKK+LLL-2
```

```
      INDT=JORD7+1
      DO 106 IND=1,INDT
  106 SUBPR7(IND) = 0
      DO 107 KKKK=1,KKK
      DO 107 LLLL=1,LLL
      IND=KKKK+LLLL-1
  107 SUBPR7(IND)=SUBPR7(IND)+SUBPR6(KKKK)*A(7,J2,LLLL)
      J2SIGN=I2SIGN
      IF(J2.LT.I1)J2SIGN=-J2SIGN
      IF(J2.LT.J1)J2SIGN=-J2SIGN
      IF(J2.LT.K1)J2SIGN=-J2SIGN
      IF(J2.LT.L1)J2SIGN=-J2SIGN
      IF(J2.LT.M1)J2SIGN=-J2SIGN
      IF(J2.LT.I2)J2SIGN=-J2SIGN
  109 KKK=JORD7+1
      LLL=NORD(8,K2)+1
      JORD8=KKK+LLL-2
      INDT=JORD8+1
      DO 110 IND=1,INDT
  110 SUBPR8(IND) = 0
      DO 111 KKKK=1,KKK
      DO 111 LLLL=1,LLL
      IND=KKKK+LLLL-1
  111 SUBPR8(IND)=SUBPR8(IND)+SUBPR7(KKKK)*A(8,K2,LLLL)
      K2SIGN=J2SIGN
      IF(K2.LT.I1)K2SIGN=-K2SIGN
      IF(K2.LT.J1)K2SIGN=-K2SIGN
      IF(K2.LT.K1)K2SIGN=-K2SIGN
      IF(K2.LT.L1)K2SIGN=-K2SIGN
      IF(K2.LT.M1)K2SIGN=-K2SIGN
      IF(K2.LT.I2)K2SIGN=-K2SIGN
      IF(K2.LT.J2)K2SIGN=-K2SIGN
  113 KKK=JORD8+1
      LLL=NORD(9,L2)+1
      JORD9=KKK+LLL-2
      INDT=JORD9+1
      DO 114 IND=1,INDT
  114 SUBPR9(IND) = 0
      DO 115 KKKK=1,KKK
      DO 115 LLLL=1,LLL
      IND=KKKK+LLLL-1
  115 SUBPR9(IND)=SUBPR9(IND)+SUBPR8(KKKK)*A(9,L2,LLLL)
      L2SIGN=K2SIGN
      IF(L2.LT.I1)L2SIGN=-L2SIGN
      IF(L2.LT.J1)L2SIGN=-L2SIGN
      IF(L2.LT.K1)L2SIGN=-L2SIGN
      IF(L2.LT.L1)L2SIGN=-L2SIGN
      IF(L2.LT.M1)L2SIGN=-L2SIGN
      IF(L2.LT.I2)L2SIGN=-L2SIGN
      IF(L2.LT.J2)L2SIGN=-L2SIGN
      IF(L2.LT.K2)L2SIGN=-L2SIGN
  117 KKK=JORD9+1
```

```
      LLL=NORD(10,M2)+1
      JORD10=KKK+LLL-2
      INDT=JORD10+1
      DO 118 IND=1,INDT
  118 SUBP10(IND) = 0
      DO 119 KKKK=1,KKK
      DO 119 LLLL=1,LLL
      IND=KKKK+LLLL-1
  119 SUBP10(IND)=SUBP10(IND)+SUBPR9(KKKK)*A(10,M2,LLLL)
      M2SIGN=L2SIGN
      IF(M2.LT.I1)M2SIGN=-M2SIGN
      IF(M2.LT.J1)M2SIGN=-M2SIGN
      IF(M2.LT.K1)M2SIGN=-M2SIGN
      IF(M2.LT.L1)M2SIGN=-M2SIGN
      IF(M2.LT.M1)M2SIGN=-M2SIGN
      IF(M2.LT.I2)M2SIGN=-M2SIGN
      IF(M2.LT.J2)M2SIGN=-M2SIGN
      IF(M2.LT.K2)M2SIGN=-M2SIGN
      IF(M2.LT.L2)M2SIGN=-M2SIGN
      I3SIGN=M2SIGN
      IF(I3.LT.I1)I3SIGN=-I3SIGN
      IF(I3.LT.J1)I3SIGN=-I3SIGN
      IF(I3.LT.K1)I3SIGN=-I3SIGN
      IF(I3.LT.L1)I3SIGN=-I3SIGN
      IF(I3.LT.M1)I3SIGN=-I3SIGN
      IF(I3.LT.I2)I3SIGN=-I3SIGN
      IF(I3.LT.J2)I3SIGN=-I3SIGN
      IF(I3.LT.K2)I3SIGN=-I3SIGN
      IF(I3.LT.L2)I3SIGN=-I3SIGN
      IF(I3.LT.M2)I3SIGN=-I3SIGN
  121 KKK=JORD10+1
      LLL=NORD(11,I3)+1
      JORD11=KKK+LLL-2
      INDT=JORD11+1
      DO 122 IND=1,INDT
  122 SUBP11(IND) = 0
      DO 123 KKKK=1,KKK
      DO 123 LLLL=1,LLL
      IND=KKKK+LLLL-1
  123 SUBP11(IND)=SUBP11(IND)+SUBP10(KKKK)*A(11,I3,LLLL)
      J3SIGN=I3SIGN
      IF(J3.LT.I1)J3SIGN=-J3SIGN
      IF(J3.LT.J1)J3SIGN=-J3SIGN
      IF(J3.LT.K1)J3SIGN=-J3SIGN
      IF(J3.LT.L1)J3SIGN=-J3SIGN
      IF(J3.LT.M1)J3SIGN=-J3SIGN
      IF(J3.LT.I2)J3SIGN=-J3SIGN
      IF(J3.LT.J2)J3SIGN=-J3SIGN
      IF(J3.LT.K2)J3SIGN=-J3SIGN
      IF(J3.LT.L2)J3SIGN=-J3SIGN
      IF(J3.LT.M2)J3SIGN=-J3SIGN
      IF(J3.LT.I3)J3SIGN=-J3SIGN
```

```
125   KKK=JORD11+1
      LLL=NORD(12,J3)+1
      JORD12=KKK+LLL-2
      INDT=JORD12+1
      DO 126 IND=1,INDT
 126  SUBP12(IND) = C
      DO 127 KKKK=1,KKK
      DO 127 LLLL=1,LLL
      IND=KKKK+LLLL-1
127   SUBP12(IND)=SUBP12(IND)+SUBP11(KKKK)*A(12,J3,LLLL)
      IND=JORD12+1
      IF(J3SIGN.EQ.-1)GO TO 140
      DO 139 LITTLE=1,IND
139   ANS(LITTLE)=ANS(LITTLE)+SUBP12(LITTLE)
      GO TO 142
140   DO 141 LITTLE=1,IND
141   ANS(LITTLE)=ANS(LITTLE)-SUBP12(LITTLE)
142   NA=MAXO(NA,JORD12)
      KFLAG=12
145   CONTINUE
146   IF(KFLAG.GT.11)KFLAG=11
147   IF(KFLAG.GT.10)KFLAG=10
148   IF(KFLAG.GT.9)KFLAG=9
149   IF(KFLAG.GT.8)KFLAG=8
150   IF(KFLAG.GT.7)KFLAG=7
151   IF(KFLAG.GT.6)KFLAG=6
152   IF(KFLAG.GT.5)KFLAG=5
153   IF(KFLAG.GT.4)KFLAG=4
154   IF(KFLAG.GT.3)KFLAG=3
155   KFLAG=2
156   CONTINUE
402   NR=NA+1
      ZANS=ANS(NR)
      IF(ABS(ZANS).GT.1.E-60) GO TO 400
      NA=NA-1
      IF(NA.LT.0)GO TO 500
      GO TO 402
400   RETURN
500   WRITE(6,501)
 501  FORMAT(62H SOMETHING IS WRONG.  SKBJI HAS GENERATED A ZEROTH POLYN
     1OMIAL        )
      STOP 500
      END
```

```
      SUBROUTINE ISE
C
C     THIS SUBROUTINE CALCULATES THE TOTAL ISE BY KRASOVSKII'S METHOD AF
C     HAS DETERMINED THE SYSTEM TRANSFER FUNCTION AND TESTED FOR STABILI
C
      COMMON COMON(4400)
      COMMON / 5 / A(1599),BMATRX(1600)
      INTEGER AORDER(12,12,1),BORDER(12,12,1),ORDNMO,ORDDMO
      REAL    MODELN(8,11),MODELD(8,11)
      DIMENSION AISE(8),AMD(11),AMN(11),AN1(47),AN2(47)
      DIMENSION ASD(37),ASKBJI(12,12,11),ASN(37),B(47)
      DIMENSION ICOST(8)
      DIMENSION CELT(47),MODDOR(8),MODNOR(8),MORD(12,12)
      DIMENSION NOUT(8),NSYST(8),NTYPE(8),NUMEQ(1),P(5),R(36,18)
      DIMENSION SCLSYS(8),TEMPDM(11),TEMPNM(11),TFDEN(37),TFNUM(37)
      DIMENSION U(47),V(47),AMAT(2209),VV(47),VL(47),WEIGHT(8)
      DIMENSION NIN(8),NMOD(8)
      EQUIVALENCE (MODELD,COMON( 519))
      EQUIVALENCE (MODELN,COMON( 607))
      EQUIVALENCE (AISE  ,COMON( 695))
      EQUIVALENCE (INDMOD,COMON( 727))
      EQUIVALENCE (ICOST ,COMON(4331))
      EQUIVALENCE (P     ,COMON( 780))
      EQUIVALENCE (TOTISE,COMON( 806))
      EQUIVALENCE (WEIGHT,COMON( 807))
      EQUIVALENCE (SCLSYS,COMON( 815))
      EQUIVALENCE (AMAT,TFDEN,ASD,COMON(823))
      EQUIVALENCE (TFNUM,ASN,COMON(904))
      EQUIVALENCE (TEMPDM,AMD,COMON(985))
      EQUIVALENCE (TEMPNM,AMN,COMON(996))
      EQUIVALENCE (ASKBJI,AN1,COMON(1007))
      EQUIVALENCE (AN2   ,COMON(1054))
      EQUIVALENCE (R     ,COMON(1101))
      EQUIVALENCE (AORDER,COMON(   1))
      EQUIVALENCE (BORDER,COMON( 145))
      EQUIVALENCE (MODDOR,COMON( 289))
      EQUIVALENCE (MODNOR,COMON( 297))
      EQUIVALENCE (MORD  ,COMON( 305))
      EQUIVALENCE (NIN   ,COMON( 465))
      EQUIVALENCE (NMOD  ,COMON( 473))
      EQUIVALENCE (NOUT  ,COMON( 481))
      EQUIVALENCE (NSYST ,COMON( 489))
      EQUIVALENCE(NTYPE,COMON(497))
      EQUIVALENCE (NUMEQ ,COMON( 505))
      EQUIVALENCE (IEQ   ,COMON( 508))
      EQUIVALENCE (ISETRM,COMON( 510))
      EQUIVALENCE (ISTABL,COMON( 511))
      EQUIVALENCE (NISE  ,COMON( 512))
      EQUIVALENCE (KNTISE,COMON( 514))
      EQUIVALENCE (KPARAM,COMON( 515))
      EQUIVALENCE (NORDEN,ND,IMAX,COMON(517))
      EQUIVALENCE (NORNUM,NC,COMON(518))
```

```
      EQUIVALENCE(VV(1),DELT(1)),(AOK,SCALE),(NA,ORDNMD),(NB,ORDDMD)
C
C     THE BLANK SUBROUTINE CALCULATES THOSE ELEMENTS OF THE A AND THE RH
C                   MATRICES THAT ARE FUNCTIONS OF P(*)
C
      CALL BLANK
      CALL COSTI
      KNTISE=KNTISE+1
C
C     BEGIN ITERATION OF ISE TERMS
C
      TOTISE=0.00
      DO 99999 NNNISE=1,ISETRM
      IF(ICOST(NNNISE).LT.1.OR.ICOST(NNNISE).GT.2)GO TO 99999
      K1=NIN(NNNISE)
      K2=NMOD(NNNISE)
      K3=NOUT(NNNISE)
      K4=NSYST(NNNISE)
      K5=NTYPE(NNNISE)
C
C     FINAL ITH-SYSTEM CALCULATIONS (DETERMINATION OF TRANSFER FUNCTION-
C                                           TFNUM/TFDEN)
C
      IEQ=NUMEQ(K4)
      DO 49 I=1,IEQ
      DO 49 J=1,IEQ
      MORD(I,J)=AORDER(I,J,K4)
      DO 49 K=1,11
      NOW=NEW(I,J,K,K4)
 49   ASKBJI(I,J,K)=A(NOW)
      CALL SKBJI
      NORDEN=NORNUM
      NORNUM=NORNUM+1
      DO 50 I=1,NORNUM
 50   TFDEN(I)=TFNUM(I)
      DO 51 I=1,IEQ
      MORD(I,K3)=BORDER(I,K1,K4)
      DO 51 K=1,11
      NOW=NEW(I,K1,K,K4)
 51   ASKBJI(I,K3,K)=BMATRX(NOW)
      CALL SKBJI
      ORDNMD=MODNOR(K2)
      IF(ORDNMD.EQ.-1)ORDNMD=0
      ORDDMD=MODDOR(K2)
      IF(ORDDMD.EQ.-1)ORDDMD=0
      DO 55 K=1,11
      TEMPNM(K)=MODELN(K2,K)
 55   TEMPDM(K)=MODELD(K2,K)
C
C     ADJUST FOR INPUT (BOTH SYSTEM AND MODEL)
C
C                        THE CALCULATION OF ISE IS SET UP FOR A STEP INPUT
```

```
C                           THE INPUT IS NOT A STEP, BOTH THE SYSTEM AND THE
C                           MUST BE MANIPULATED BY MULTIPLYING EITHER NUMERAT
C                           DENOMINATOR OF BOTH BY S OR S**2 OR ...
      IF(K5-1)60,59,80
 60   I=-K5+1
      IF(NORDEN.LT.NORNUM+I)GO TO 70
      IF(NORNUM+I.GT.36)GO TO 75
      AOK=1.E50
      IF(ORDDMD.LT.ORDNMD+I)GO TO 72
 61   IF(ORDNMD+I.GT.10)GO TO 77
      NORNUM=NORNUM+I
      NU=-K5+2
      KMAX=NORNUM+1
      DO 65 K=NU,KMAX
      L=KMAX+NU-K
      J=L-NU+1
 65   TFNUM(L)=TFNUM(J)
      DO 66 K=1,I
 66   TFNUM(K)=0.00
      ORDNMD=ORDNMD+I
      KMAX=ORDNMD+1
      DO 67 K=NU,KMAX
      L=KMAX+NU-K
      J=L-NU+1
 67   TEMPNM(L)=TEMPNM(J)
      DO 68 K=1,I
 68   TEMPNM(K)=0.00
      IF(AOK.LT.1.E25)ORDNMD=0
      GO TO 100
 70   WRITE(6,6070)K5,K4
 6070 FORMAT(30H1ERROR AT 70--FOR INPUT 1/(S**,I2,*), DENOMINATOR IS TOO
     .SMALL FOR NUMERATOR--SYSTEM*,I2)
      STOP 70
 72   AOK=0.00
      KMAX=ORDNMD+1
      DO 73 K=1,KMAX
 73   AOK=AOK+ ABS(TEMPNM(K))
      IF(AOK.LE.1.E-50)GO TO 61
 74   WRITE(6,6074)K5,K2
 6074 FORMAT(30H1ERROR AT 74--FOR INPUT 1/(S**,I2,*), DENOMINATOR IS TOO
     .SMALL FOR NUMERATOR--MODEL*,I2)
      STOP 74
 75   WRITE(6,6075)K5,K4
 6075 FORMAT(30H1ERROR AT 75--FOR INPUT 1/(S**,I2,*), NUMERATOR IS GREAT
     .ER THAN OR EQUAL TO 36--SYSTEM*,I2)
      STOP 75
 77   WRITE(6,6077)K5,K2
 6077 FORMAT(30H1ERROR AT 77--FOR INPUT 1/(S**,I2,*), NUMERATOR IS GREAT
     .ER THAN OR EQUAL TO 10--MODEL*,I2)
      STOP 77
 80   IF(NORDEN+K5.GT.37)GO TO 90
      IF(ORDDMD+K5.GT.11)GO TO 97
```

```
      NORDEN=NORDEN+K5-1
      KMAX=NORDEN+1
      DO 85 K=K5,KMAX
      L=KMAX+K5-K
      J=L-K5+1
 85   TFDEN(L)=TFDEN(J)
      J=K5-1
      DO 86 I=1,J
 86   TFDEN(I)=0.CC
      ORDDMD=ORDDMD+K5-1
      KMAX=ORDDMD+1
      DO 87 K=K5,KMAX
      L=KMAX+K5-K
      J=L-K5+1
 87   TEMPDM(L)=TEMPDM(J)
      J=K5-1
      DO 88 K=1,J
 88   TEMPDM(K)=C.CC
      GO TO 10C
 90   WRITE(6,6090)K5,K4
 6090 FORMAT(30H1ERROR AT 6C--FOR INPUT 1/(S**,I2,*), DENOMINATOR IS ALR
     .EADY TOO BIG--SYSTEM*,I2)
      STOP 60
 97   WRITE(6,6C97)K5,K2
 6097 FORMAT(30H1ERROR AT 67--FCR INPUT 1/(S**,I2,*), DENOMINATOR IS ALR
     .EADY TOO BIC--MODEL*,I2)
      STOP 67
 98   WRITE(6,6C98)NA,NB,NC,ND
 6098 FORMAT(1H1,T10,*MODEL   NUMERATOR   ORDER = *,I2,
     .          /,T1C,*MODEL   DENOMINATOR CRDER = *,I2,
     .          /,T10,*SYSTEM NUMERATOR   ORDER = *,I2,
     .          /,T1C,*SYSTEM DENOMINATOR CRDER = *,I2,
     .         //,T1C,*AT LEAST CNE OF THE ABOVE IS TCC BIG*)
      STOP 66
 99   IF((NA.GT.10).OR.(NB.GT.10).OR.(NC.GT.36).CR.(ND.GT.36))GO TO 98
 100  NA=ORDNMD+1
      NB=ORDDMD+1
      NC=NORNUM+1
      ND=NORDEN+1
C
C     CHECK FOR FREE S'S IN NUMERATOR AND DENCMINATOR AND CANCELL IF POS
C
 103  IF( ABS(TFDEN(1)).GT.1.E-60)GO TO 110
      IF( ABS(TFNUM(1)).GT.1.E-60)GO TC 109
      DO 106 I=2,NC
      J=I-1
 106  TFNUM(J)=TFNUM(I)
      DO 107 I=2,ND
      J=I-1
 107  TFDEN(J)=TFDEN(I)
      NC=NC-1
      ND=ND-1
```

```
      GO TO 103
 109  WRITE(6,6109)
 6109 FORMAT(*1ERRCR AT 106--THERE IS A FREE S IN THE DENOMINATOR AND NO
     .FREE S IN THE NUMERATOR CF THE SYSTEM--INPUT ACCOUNTED FOR*)
      STOP 106
 110  IF( ABS(TEMPDM(1)).GT.1.E-60)GO TO 120
      IF( ABS(TEMPNM(1)).GT.1.E-60)GO TO 119
      DO 116 I=2,NA
      J=I-1
 116  TEMPNM(J)=TEMPNM(I)
      DO 117 I=2,NB
      J=I-1
 117  TEMPDM(J)=TEMPDM(I)
      NA=NA-1
      NB=NB-1
      GO TO 110
 119  WRITE(6,6119)
 6119 FORMAT(*1ERROR AT 116--THERE IS A FREE S IN THE DENOMINATOR AND NO
     .FREE S IN THE NUMERATOR CF THE MODEL--INPUT ACCOUNTED FOR*)
      STOP 116
C
C     BEGIN ROUTH TEST
C
C  TEST FOR ODD OR EVEN NUMBER OF COEFFICIENTS & DEFINE JMAX
C
 120  JMAX=IMAX
      IF((-1)**IMAX.LT.0)JMAX=JMAX+1
      JMAX=JMAX/2
C
C  LOAD FIRST TWO ROWS OF ROUTH ARRAY
C
      DO 306 J=1,JMAX
      K1=IMAX-2*J+2
      K2=K1-1
      R(1,J)=ASD(K1)
      IF(K2.GT.0)GO TO 305
      R(2,J)=0.00
      GO TO 306
 305  R(2,J)=ASD(K2)
 306  CONTINUE
C
C  COMPUTE ROUTH ARRAY & CHECK STABILITY AFTER EACH ROW IS ESTABLISHED
C
      IF( ABS(R(1,1)).GT.1.E-64)GO TO 325
      WRITE(6,6325)
 6325 FORMAT(/,* THE 1ST ENTRY IN THE ROUTH ARRAY IS ZERO*)
      GO TO 335
 325  RR=R(2,1)/R(1,1)
      DO 330 I=3,IMAX
      R(I,JMAX)=0.00
      JMA1=JMAX-1
      IF(RR.LE.0.00)GO TO 330
```

```
      DO 327 J=1,JMA1
      K3=I-2
      K4=J+1
      K5=I-1
 327  R(I,J)=R(K3,K4)-R(K3,1)*R(K5,K4)/R(K5,1)
      RR=R(I,1)/R(K5,1)
 330  CONTINUE
      IF(RR.GT.0.00) GO TO 350
      WRITE(6,6330)
 6330 FORMAT(*1ROUTH TEST FAILED, SYSTEM IS UNSTABLE*)
 335  WRITE(6,6006)(I,P(I),I=1,KPARAM)
 6006 FORMAT(///,* THIS DIFFICULTY OCCURS AT*,(/,2X,*P(*,I2,*) = *,
     .       OPE15.8))
      ISTABL=-1
      NISF=NNNISE
      NORNUM=NORNUM-1
      NORDEN=NORDEN-1
      WRITE(6,6336)
 6336 FORMAT(//,1X,*THE FOLLOWING TRANSFER FUNCTION(S) HAVE BEEN ADJUSTE
     .D FOR THEIR RESPECTIVE INPUT AND FREE S**S IN NUMERATOR AND DENOMI
     .NATOR HAVE BEEN*,/,1X,*CANCELLED WHERE POSSIBLE*)
      CALL WROTE4
      RETURN
C
C     END ROUTH TEST
C
C       FINAL SYSTEM CALCULATIONS (ISE POLYNOMIAL COEFICIENTS)
C
C         COMPUTE COST FUNCTION
C
 350  AISF(NNNISE)=0.00
C
C     BEGIN EVALUATION OF THE ERROR EXPRESSION
C
C     SCALE SYSTEM STEADY STATE TO MODEL STEADY STATE
C
      IF(ABS(AMN(1)).GT.1.E-64)GO TO 356
      IF(ABS(ASN(1)).LT.1.E-64)GO TO 358
  352 WRITE(6,6352)
 6352 FORMAT(*1THE MODEL AND THE SYSTEM CANNOT BE MADE TO HAVE EQUAL STE
     .ADY STATE RESPONSES TO A STEP*)
      GO TO 335
  356 IF(ABS(ASN(1)).GT.1.E-64)GO TO 364
  358 AOK=0.
      DO 360 I=1,NC
  360 AOK=AOK+ABS(ASN(I))
      IF(AOK.GT.0.)GO TO 362
      WRITE(6,6362)
 6362 FORMAT(*1THE FORWARD LOOP GAIN IS ZERO*)
      GO TO 335
  362 IF(ABS(AMN(1)).GT.1.E-64)GO TO 352
      GO TO 375
```

```
 464 SCLSYS(NNNISE)=AMN(1)*ASD(1)/AMD(1)/ASN(1)
     DO 367 I=1,NC
 367 ASN(I)=ASN(I)*SCLSYS(NNNISE)
 375 CALL PMPY(AN1,N1,AMN,NA,ASD,ND)
     CALL PMPY(AN2,N2,ASN,NC,AMD,NB)
     CALL PSUB(U,NU,AN1,N1,AN2,N2)
     DO 466 J=1,NU
     NU1 = NU + 1 - J
     IF( ABS(U(NU1)).LT.1.E-64)GO TO 465
     IORDEN=NU1
     GO TO 467
 465 IORDEN=NU-J
 466 CONTINUE
 467 CONTINUE
     NU=IORDEN
     CALL PMPY(V,NV,AMD,NB,ASD,ND)
C
C    U(1) MUST BE ZERO
C
     U(1)=0.00
C
C    END EVALUATION OF THE ERROR EXPRESSION
C
C  SCALE V TO PREVENT VMAT INVERSION OVERFLOW
     SCALE=0.00
     DO 488 I=1,NV
 488 SCALE=SCALE+ALOG(ABS(V(I)))
     SCALE=SCALE/NV
     SCALE= EXP(SCALE)
     DO 492 J=1,NV
 492 V1(J)=V(J)/SCALE
C
C  ESTABLISH 'V' MATRIX (VMAT)
C
     NV1=NV-1
     DO 504 J=1,NV1
     DO 504 I=1,NV1
     IJ=I+NV1*(J-1)
     K=NV+I-2*J
     IF(K.GT.NV)GO TO 502
     IF(K.GT.0)GO TO 503
 502 AMAT(IJ)=0.00
     GO TO 504
 503 AMAT(IJ)=V1(K)*(-1.00)**(I+J)
 504 CONTINUE
C
C  ESTABLISH 'V' COLUMN VECTOR (VV)
C
     K1=NV1-2
     DO 505 I=1,K1
 505 VV(I)=0.00
     K2=NV1-1
```

```
      VV(K2)=V1(1)
      VV(NV1)=V1(2)
C
      CALL SIMQ(AMAT,VV,NV1,KS)
      IF(KS.EQ.1) WRITE(6,6611) KS
 6611 FORMAT(1H0,2X,*SINGULAR SET OF EQUATIONS --- KS = *,I1)
C
C  COMPUTE B(I) VALUES   I=1,NU
C
      UOV=U(NU)/V(NV)
      TERM2=0.00
      IF(NU.EQ.NV)TERM2=1.00
      B(NU)=TERM2*UOV*V(1)
      B(NU)=B(NU)*B(NU)
      M=NU-1
      DO 620 K=1,M
      KA=M-K+1
      K1=K+1
      B(KA)=U(K1)-TERM2*UOV*V(K1)
      B(KA)=B(KA)*B(KA)
      DO 620 J=1,K
      KB=K+1-J
      KC=K+1+J
      RR=(-1.00)**J
      IF(KB.LE.0.OR.KC.GT.NU)GO TO 620
      AOK=2.00*RR*(U(KB)-TERM2*UOV*V(KB))*(U(KC)-TERM2*UOV*V(KC))
      B(KA)=B(KA)+AOK
 620  CONTINUE
C
C  COMPUTE ISE (AISE)
C
      AOK=1.00/(V(1)*V(1))
      AISE(NNNISE)=-AOK*(-TERM2*UOV*V(1))*(U(2)-TERM2*UOV*V(2))
      AOK=.500*AOK
      K=1
      IF(NU.EQ.NV)K=2
      DO 630 L=K,NU
      L1=NV1-NU+L
      RR=AOK*B(L)*DELT(L1)
 630  AISE(NNNISE)=AISE(NNNISE)+RR
      IF(ICOST(NNNISE).NE.2)GO TO 99999
      AISE1=ABS(AISE(NNNISE))
      AIS=AISE(NNNISE)/AISE1
      AISE1=SQRT(AISE1)
      AISE(NNNISE)=AIS*AISE1
99999 TOTISE=TOTISE+AISE(NNNISE)*WEIGHT(NNNISE)
      ISTABL=1
      RETURN
      END
```

```
      SUBROUTINE POLYRT(M,C,TOL1,RX,RY,RMULT,NR,ISW)
      DIMENSION C(100),RX(100),RY(100),RMULT(100),RTX(2),RTY(2),IRX(2),
     .          IRY(2),FRX(2),FRY(2),L(4),NRF(4),
     .          P(2),Q(2),A(100,4),RF(4)
      EQUIVALENCE (X,FRX(1),IRX(1)),(Y,FRY(1),IRY(1))
      FCTR = 4294967296.
      TOLM = TOL1*TOL1
      N = M + 1
      NR = 0
      DO 2000 I=1,N
      IF(C(I)) 2002,2000,2002
2000  CONTINUE
2002  N = N-I+1
      IF(N-100) 2010,2010,2005
2005  ISW = 3
2008  NR = -1
      RETURN
2010  L(3) =N
      L(4) =N-1
      K= I-1
      DO 2015 I=1,N
      J = I+K
2015  A(I,1) = C(J)
      NZR = 0
2020  IF(N-1) 2060,2044,2022
2022  IF(A(N,1)) 2024,2023,2024
2023  NZR = NZR +1
      N = N - 1
      GO TO 2020
2024  T1 =  ABS(A(N,1))
      T2 = 16.* ABS(A(1,1))
      I = 0
2028  IF(T1-T2)2032,2032,2030
2030  T1 = T1/16.
      I = I+1
      GO TO 2028
2032  NF = (4*I)/(N-1)
      RFCTR = 1.
      IF(NF) 2044,2044,2034
2034  DO 2035 I=1,NF
2035  RFCTR = RFCTR*2.
      T1 = 1.
      K = 0
      DO 2042 I=2,N
      IF(T1-1.E50) 2040,2040,2036
2036  K = K+1
      DO 2038 J=1,K
2038  A(I,1) = A(I,1)/RFCTR
      GO TO 2042
2040  T1 = T1*RFCTR
2042  A(I,1) = A(I,1)/T1
2044  K = L(3)
```

```
      DO 2045 I=1,K
      A(I,3) =A(I,1)
2045  A(I,4) = A(I,3)*(N-I)
      IF(NZR) 2050,2050,2048
2048  RX(1) = 0.
      RY(1) = C.
      RMULT(1) = NZR
      NR = 1
2050  NT = N-1
      L(1) = N
      L(2) = NT
      DO 2056 I=1,N
2056  A(I,2) = A(I,1)*(N-I)
      IF(N-1) 2060,2060,2100
2060  ISW = 1
      RETURN
2100  XO = .50010IE-2
      YO = .1001C1E-1
      NSG = 0
      ACHG = 1.
2110  DO 2111 I=1,4
      NRF(I) = 0
2111  RF(I) = 1.
      IF(NSG-NSG/2) 2112,2112,2113
2112  XO = -10.*XO
      YO = -10.*YO
2113  T = YO
      YO =XO
      XO =T
      NSG = NSG+1
      ACHG =ACHG*1000.
      XR = XO
      YR = YO
      K1 = 0
      NI = 1
      NFI = 0
2114  IVSW = -2
      ARL = 1.E50
2118  K = 0
2120  K = K+1
      K1 = K1+1
      P(K) = A(1,K1)*RF(K1)
      Q(K) = 0.
      NRT = L(K1)
      IF(NRT-2) 2170,2121,2121
2121  DO 2165 I=2,NRT
      T = P(K)*XR - Q(K)*YR + A(I,K1)*RF(K1)
      Q(K) = P(K)*YR + Q(K)*XR
      P(K) = T
2122  ARS = ABS(P(K)) + ABS(Q(K))
      IF(ARS-1.E30) 2130,2130,2124
2124  P(K) = P(K)/FCTR
```

```
      Q(K) = Q(K)/FCTR
      IF(NRF(K1)-7) 2126,2128,2128
 2126 RF(K1) = RF(K1)/FCTR
 2128 NRF(K1) = NRF(K1) + 1
      GO TO 2122
 2130 IF(ARS-1.E-20) 2132,2165,2165
 2132 IF(NRF(K1)) 2165,2165,2134
 2134 P(K) = P(K)*FCTR
      Q(K) = Q(K)*FCTR
      IF(NRF(K1)-7)2136,2136,2138
 2136 RF(K1) = RF(K1)*FCTR
 2138 NRF(K1) = NRF(K1) - 1
      ARS =  ABS(P(K))+ ABS(Q(K))
      GO TO 2130
 2165 CONTINUE
 2170 IF(K-2) 2120,2190,2190
 2190 DIV = P(2)*P(2) + Q(2)*Q(2)
      IF(DIV) 2196,2207,2196
 2196 DX = -(P(1)*P(2) +Q(1)*Q(2))/DIV
      DY = (P(1)*Q(2)-Q(1)*P(2))/DIV
      NRT = NRF(K1) - NRF(K1-1)
      IF(NRT) 2198,2205,2200
 2198 NRT = -NRT
      T2 = 1./FCTR
      GO TO 2202
 2200 T2 = FCTR
 2202 DO 2204 I=1,NRT
      DX = DX/T2
 2204 DY = DY/T2
 2205 IF( ABS(DX)+ ABS(DY) - ACHG) 2207,2207,2206
 2206 DX =  SIGN(ACHG/1000.,DX)
      DY =  SIGN(ACHG/1000.,DY)
      IVSW = -2
 2207 XR = XR + DX
      YR = YR + DY
      ARS =  ABS(DX)+ ABS(DY)
      IF(ARS-( ABS(XR)+ ABS(YR))*1.E-6) 2216,2216,2228
 2216 IF(ARS-ARL) 2227,2218,2218
 2218 IVSW = IVSW+1
      IF(IVSW) 2227,2227,2224
 2224 IF(NFI) 2226,2226,2250
 2226 NFI = 1
      XT =XR
      YT =YR
      K1 =2
      GO TO 2114
 2227 ARL = ARS
 2228 IF(NFI) 2230,2230,2240
 2230 NI = NI + 1
      K1 = 0
      IF(NI-500) 2118,2118,2232
 2232 IF(NSG-8) 2110,2234,2234
```

```
2234 ISW = 2
     GO TO 2008
2240 NFI = NFI+1
     KI = 2
     IF(NFI-100) 2118,2118,2250
2250 IF( ABS(XR-XT)+ ABS(YR-YT)-.1) 2295,2295,2260
2260 IF(NFI-100) 2262,2262,2280
2262 IF(ABS-1.E-6) 2264,2264,2280
2264 RTX(1) = XR*RFCTR
     RTY(1) = YR*RFCTR
     DO 2265 I=1,NR
     IF( ABS(RTX(1)-RX(I))+ ABS(RTY(1)-RY(I))-TOL1) 2280,2280,2265
2265 CONTINUE
     GO TO 2295
2280 IF(NSG-4) 2110,2290,2290
2290 X = XT
     Y = YT
     GO TO 2300
2295 X = XR
     Y = YR
2300 IF( ABS(Y)-1.E-4) 2305,2315,2315
2305 Y = 0.
     DO 2310 I=2,NT
2310 A(I,1) = A(I,1)+X*A(I-1,1)
     GO TO 2325
2315 T = X+X
     DIV = -(X*X + Y*Y)
     A(2,1) = A(2,1) + T*A(1,1)
     DO 2320 I=3,NT
2320 A(I,1) = A(I,1) + T*A(I-1,1) + DIV*A(I-2,1)
2325 X = X*RFCTR
     Y=Y*RFCTR
     IF(IRX(2)) 2330,2340,2340
2330 IRX(1) = IRX(1) + 1
     IF(FRX(1)) 2340,2335,2340
2335 IRX(1) = IRX(1) + 1048576
2340 IF(IRY(2)) 2345,2355,2355
2345 IRY(1) = IRY(1) + 1
     IF(FRY(1)) 2355,2350,2355
2350 IRY(1) = IRY(1) +1048576
2355 RTX(1) = X
     RTY(1) = Y
     IF(Y) 2365,2360,2365
2360 NRT = 1
     GO TO 2400
2365 NRT = 2
     RTX(2) = X
     RTY(2) = -Y
2400 N = N-NRT
     DO 2470 I=1,NRT
     J = 1
     JFL = 0
```

```
2405 IF(J-NR) 2410,2410,2455
2410 IF((RX(J)-RTX(I))*(RX(J)-RTX(I))+(RY(J)-RTY(I))*(RY(J)-RTY(I))
    1    -TOLM) 2415,2415,2420
2415 RMULT(J) = RMULT(J) + 1.
     GO TO 2462
2420 IF(RX(J)-RTX(I)) 2440,2425,2435
2425 IF(ABS(RY(J))-ABS(RTY(I)))2435,2430,2436
2430 IF(RY(J)) 2436,2436,2435
2435 J = J+1
     GO TO 2405
2436 IF(JFL) 2438,2438,2435
2438 JFL = J
     GO TO 2435
2440 IF(RTX(I)-RX(J)-TOL1) 2436,2441,2441
2441 IF(JFL) 2444,2444,2442
2442 J = JFL
2444 K = NR+1
2445 IF(K-J) 2460,2460,2450
2450 K1 = K-1
     RX(K) = RX(K1)
     RY(K) = RY(K1)
     RMULT(K) = RMULT(K1)
     K = K1
     GO TO 2445
2455 IF(JFL) 2460,2460,2442
2460 RMULT(J) = 1.
     NR = NR+1
2462 RX(J)= RTX(I)
     NR1 = NR1 + 1
2470 RY(J) = RTY(I)
     GO TO 2050
     END
```

```
      SUBROUTINE PMPY (Z,IDIMZ,X,IDIMX,Y,IDIMY)
C
C     THIS IBM-SSP SUBROUTINE MULTIPLYS TWO POLYNOMIALS TOGETHER
C
      DIMENSION Z(1),X(1),Y(1)
      IF(IDIMX*IDIMY.GT.0)GO TO 20
      IDIMZ=0
      GO TO 50
   20 IDIMZ=IDIMX+IDIMY-1
      DO 30 I=1,IDIMZ
   30 Z(I)=0.
      DO 40 I=1,IDIMX
      DO 40 J=1,IDIMY
      K=I+J-1
   40 Z(K)=X(I)*Y(J)+Z(K)
   50 RETURN
      END
```

```
      SUBROUTINE POLPRO
C
C     THIS SUBROUTINE MULTIPLYS NNN POLYNOMIALS TOGETHER
C
C     POLYNOMIALS CAN ALL BE IN ASCENDING ORDER OR CAN ALL BE IN DESCEND
C     ORDER.  THE ANZ(*) PRODUCT POLYNOMIAL WILL BE IN THE SAME ORDER AS
C     FACTOR POLYNOMIALS.
C
C     GLOBAL VARIABLES ARE PASSED BY THE PAN COMMON BLOCK
C
      COMMON COMON(4400)
      DIMENSION ANZ(81),ATERM(47,47),NNORD(10),TEMP(81)
      EQUIVALENCE (ANZ    ,COMON( 704))
      EQUIVALENCE (ATERM  ,COMON( 805))
      EQUIVALENCE (TEMP   ,COMON( 977))
      EQUIVALENCE (IDIMR  ,COMON( 497))
      EQUIVALENCE (NNORD  ,COMON( 508))
      EQUIVALENCE (NNN    ,COMON( 518))
      DO 5 I=1,NNN
      IF(NNORD(I).GE.0)GO TO 5
      ANZ(1)=0.0C
      IDIMR=0
      GO TO 70
    5 CONTINUE
      IDIMR=NNORD(1)+1
      DO 10 J=1,IDIMR
   10 ANZ(J)=ATERM(1,J)
      IF(NNN.LT.2)GO TO 70
      DO 60 L=2,NNN
      IDIM=NNORD(L)+1
      DO 20 I=1,IDIMR
   20 TEMP(I)=ANZ(I)
      ITEMP=IDIMR
      IF(ITEMP*IDIM.GT.0)GO TO 40
      IDIMR=0
      GO TO 70
   40 IDIMR=ITEMP+IDIM-1
      DO 50 I=1,IDIMR
   50 ANZ(I)=0.DC
      DO 60 I=1,ITEMP
      DO 60 J=1,IDIM
      K=I+J-1
   60 ANZ(K)=TEMP(I)*ATERM(L,J)+ANZ(K)
   70 RETURN
      END
```

```
      SUBROUTINE PSUB (Z,IDIMZ,X,IDIMX,Y,IDIMY)
C
C     THIS IBM-SSP SUBROUTINE SUBTRACTS ONE POLYNOMIAL FROM ANOTHER
C
      DIMENSION Z(1),X(1),Y(1)
      NDIM = IDIMX
      IF(IDIMX.GE.IDIMY)GO TO 20
      NDIM=IDIMY
 20   IF(NDIM.LE.0)GO TO 90
      DO 80 I=1,NDIM
      IF(I.GT.IDIMX)GO TO 60
      IF(I.GT.IDIMY)GO TO 70
      Z(I)=X(I)-Y(I)
      GO TO 80
   60 Z(I)=-Y(I)
      GO TO 80
   70 Z(I) = X(I)
   80 CONTINUE
   90 IDIMZ=NDIM
      RETURN
      END
```

```
      SUBROUTINESIMQ(A,B,N,KS)
      DIMENSIONA(1),B(1)
C
C     FORWARD SOLUTION
C
      TOL=0.0
      KS=0
      JJ=-N
      DO65J=1,N
      JY=J+1
      JJ=JJ+N+1
      BIGA=0
      IT=JJ-J
      DO30I=J,N
C
C     SEARCH FOR MAXIMUM COEFFICIENT IN COLUMN
C
      IJ=IT+I
      IF(ABS(BIGA)-ABS(A(IJ)))20,30,30
20    BIGA=A(IJ)
      IMAX=I
30    CONTINUE
C
C     TEST FOR PIVOT LESS THAN TOLERANCE (SINGULAR MATRIX)
C
      IF(ABS(BIGA)-TOL)35,35,40
35    KS=1
      RETURN
C
C     INTERCHANGE ROWS IF NECESSARY
C
40    I1=J+N*(J-2)
      IT=IMAX-J
      DO50K=J,N
      I1=I1+N
      I2=I1+IT
      SAVE=A(I1)
      A(I1)=A(I2)
      A(I2)=SAVE
C
C     DIVIDE EQUATION BY LEADING COEFFICIENT
C
50    A(I1)=A(I1)/BIGA
      SAVE=B(IMAX)
      B(IMAX)=B(J)
      B(J)=SAVE/BIGA
C
C     ELIMINATE NEXT VARIABLE
C
      IF(J-N)55,70,55
55    IQS=N*(J-1)
      DO65IX=JY,N
```

```
      IXJ=IQS+IX
      IT=J-IX
      DO60JX=JY,N
      IXJX=N*(JX-1)+IX
      JJX=IXJX+IT
60    A(IXJX)=A(IXJX)-(A(IXJ)*A(JJX))
65    B(IX)=B(IX)-(B(J)*A(IXJ))
C
C     BACK SOLUTION
C
70    NY=N-1
      IT=N*N
      DO80J=1,NY
      IA=IT-J
      IB=N-J
      IC=N
      DO80K=1,J
      B(IB)=B(IB)-A(IA)*B(IC)
      IA=IA-N
80    IC=IC-1
      RETURN
      END
```

```
      SUBROUTINE MINI(F,X,N)
C     DESCRIPTION OF PARAMETERS
C        F  - SINGLE VARIABLE CONTAINING THE FUNCTION VALUE ON
C           RETURN, I.E., F=F(X)
C        X  - VECTOR OF DIMENSION N CONTAINING THE ITITIAL
C             ARGUMENTS WHERE THE ITERATION STARTS.  CN RETURN,
C             X HOLDS THE ARGUMENT CORRESPONGING TO THE COMPUTED
C             X HOLDS THE ARGUMENT CORRESPONDING TO THE COMPUTED
C             MINIMUM FUNCTION VALUE.
C        N  - NUMBER OF VARIABLES.
C     REMARKS--
C        THIS SUBROUTINE WAS ORIGINALLY PROGRAMMED BY MIKE
C        ZUCKERMAN TO NUMERICALLY SOLVE A GENERAL CONSTRAINED
C        MINIMIZATION PROBLEM VIA MEANS OF A PATTERN SEARCH
C        TECHNIQUE (REF. D. J. WILDE, OPTIMUM SEEKING METHODS).
C        THE ROUTINE HAS BEEN MODIFIED FOR PAPER PILOT AND IS
C        USED TO FIND A SET OF PILOT PARAMETERS FOR WHICH THE
C        SYSTEM IS STABLE.  INSTEAD OF SEARCHING FOR A MINIMUM
C        VALUE OF THE COST FUNCTION, F, THE SUBROUTINE IS EXITED
C     WHEN F IS LESS THAN C.  THE COST FUNCTION USED IS
C        THE MAXIMUM OF THE REAL PARTS OF THE EIGENVALUES OF
C        THE A MATRIX.  THUS WHEN THE COST FUNCTION GOES STRICTLY
C        NEGATIVE, THE A MATRIX IS ASYMPTOTICALLY STABLE.
C     SUBROUTINE CALLED
C        COST (F,X,N)
C        COST IS A USER WRITTEN SUBROUTINE.  IT MUST BE
C        WRITTEN SO THAT FOR A GIVEN VALUE OF THE ARGUMENTS, X,
C        THE FUNCTION VALUE, F, AND THE VECTOR CONSTRAINT FUNCTION
C        VALUES ARE RETURNED.  N AND M MUST NOT BE MODIFIED
C        BUT CAN BE USED TO INDICATE THE DIMENSION OF X.
      COMMON DUME(739),MSKIP
      DIMENSION X(N),TTX(5),STEPX(5),TX(5),A(5),DELX(5),CNG(5)
      IF(MSKIP.GT.0)RETURN
      IF(N.GT.5) GO TO 100
      WRITE(6,9999)
 9999 FORMAT(1X,10HINNER MINI)
    1 DO 2 I=1,N
      DELX(I) = .05
      TTX(I) = X(I)
      TX(I)  = X(I)
      STEPX(I) = 0
      DNG(I) = DELX(I)
    2 CONTINUE
      DENOM = 5.
      EPS = .0001
      ITER = 0
      TEST=1.0/(2.0**DENOM)
      KOUNT=0
      FACTOR=1.0
      NMIN=N
      CALL COST(F,X,NMIN)
    3 WRITE (6,442)
```

```
442 FORMAT(1X,16HINITIAL POINT IS)
    WRITE(6,25)F,(X(I),I=1,N)
    TF=F
 75 DO 20 I=1,N
500 TX(I)=X(I)+DELX(I)
    CALL COST(TPMA,TX,NMIN)
    IF(TPMA.GT.TF) GO TO 9
  8 A(I)= 1.
 18 X(I)=TX(I)
    TF=TPMA
    IF(TF.LT.0.) GO TO 437
    GO TO 20
  9 TX(I)=X(I)-DELX(I)
    CALL COST(TPMA,TX,NMIN)
    IF(TPMA.GT.TF) GO TO 555
 16 A(I)=-1.
    GO TO 18
555 TX(I)=X(I)
 20 CONTINUE
    WRITE(6,800)
800 FORMAT(1X,17HINNER SEARCH MOVE)
    WRITE(6,25)TF,(X(I),I=1,N)
    ITER = ITER + 1
    ATF=TF*1000000.
    AF=F*1000000.
    FIX=ABS(ATF-AF)
    IF(FIX.LE.0.9) GO TO 22
    IF(ATF.GE.AF) GO TO 22
    VAL=ABS(TF-F)
    IF(VAL-EPS) 66,66,24
 66 F=TF
    WRITE(6,26)
 26 FORMAT(1X,41HDELTA COST LESS THAN EPS.FINAL VALUES ARE )
 88 WRITE(6,25) F,(X(I),I=1,N)
 25 FORMAT(1X,7E12.3)
    RETURN
437 F=TF
    WRITE(6,9998)
9998 FORMAT(1X,26HSTABLE STARTING PARAMETERS)
    WRITE(6,25)F,(X(I),I=1,N)
    RETURN
 24 DO 27 L=1,N
    STEPX(L)=X(L)-TTX(L)+STEPX(L)
    TTX(L)=X(L)+STEPX(L)
    X(L)=TTX(L)
    A(L)=0
    TX(L)=X(L)
 27 CONTINUE
    F=TF
    KOUNT=0
    CALL COST(TF,X,NMIN)
    GO TO 513
```

```
512 TF=3.
513 WRITE(6,885)
885 FORMAT(1X,12HPATTERN MOVE)
    WRITE(6,25) TF,(X(I),I=1,N)
    IF(TF.LT.-.1) GO TO 437
    IF(ITER.GE.200) RETURN
    GO TO 75
 22 KOUNT=KOUNT+1
    DO 30 II=1,N
    X(II)=X(II)-STEPX(II)-A(II)*DELX(II)
 30 CONTINUE
    WRITE(6,888)
888 FORMAT(1X,37HREVERT TO LAST GOOD POINT AND RESTART)
 45 IF(KOUNT-2) 40,41,41
 40 TF=F
    DO 42 IJ=1,N
    STEPX(IJ)=C
    A(IJ)=0
    TX(IJ)=X(IJ)
    TTX(IJ)=X(IJ)
 42 CONTINUE
    GO TO 75
 41 IF(FACTOR-TEST) 46,46,52
 46 WRITE(6,47)
 47 FORMAT(1X,50HCREEP STEP LESS THAN ORIGINAL/100.FINAL VALUES ARE)
    GO TO 88
 52 FACTOR=FACTOR/2.
    WRITE(6,889)
889 FORMAT(1X,21HREDUCING STEP SIZE TO)
    DO 111 KM=1,N
    DELX(KM)=ONC(KM)*FACTOR
    STEPX(KM) = C
    A(KM)=0
    TX(KM)=X(KM)
    TTX(KM)=X(KM)
111 CONTINUE
    TF=F
    GO TO 75
100 WRITE(6,101)
    RETURN
101 FORMAT(1X,71H***SUBROUTINE MINI - N GREATER THAN 5 - CORRECT DIMEN
   1SION STATEMENT*** )
    END
```

```
      SUBROUTINE COST(F,X,N)
      COMMON COMCN(44CC)
      COMMON / 5 / A(1599),B(16CC)
      EQUIVALENCE (NORDEN,COMON( 518))
      EQUIVALENCE (NUMEQ ,COMCN( 505))
      EQUIVALENCE (IEQ   ,CCMCN( 5C8))
      EQUIVALENCE (MORD  ,COMCN( 305))
      EQUIVALENCE (ACRDER,CCMON(   1))
      EQUIVALENCE (TFDEN ,COMCN( 904))
      EQUIVALENCE (ROOTDR,COMON(1059))
      EQUIVALENCE (ROOTDI,CCMON(1C95))
      EQUIVALENCE (ASKBJI,CCMCN(1007))
      EQUIVALENCE (P     ,COMON( 780))
      DIMENSION ADDPR2(37)
      DIMENSION ASKBJI(12,12,11)
      DIMENSION AORDER(12,12),MORD(12,12),P(5)
      DIMENSION RMULT(37),RCOTDI(36),ROOTDR(36),TFDEN(81),X(5)
      INTEGER AORDER
      DATA TOL1 / .CCC1 /
      DO 1 I=1,N
    1 P(I) = X(I)
      DO 2 I=1,36
    2 ROOTDR(I) = -1.
      DO 4 I=1,12
      DO 4 J=1,12
    4 MORD(I,J) = -1
      CALL BLANK
      IEQ = NUMEC
      DO 49 I=1,IEQ
      DO 49 J=1,IEQ
      MORD(I,J) = AORDER(I,J)
      DO 49 K=1,11
      NOW = NEW(I,J,K,1)
   49 ASKBJI(I,J,K) = A(NOW)
      CALL SKBJI
      L = NORDEN + 1
      DO 43 I=1,L
      J = 1 + L - I
   43 ADDPR2(I) = TFDEN(J)
      CALL POLYRT(NORDEN,ADDPR2,TCL1,ROOTDR,ROOTDI,RMULT,NRR,NERR)
      F = ROOTDR(1)
      DO 3 I=2,NORDEN
      IF ( ROOTDR(I) .GT. F ) F = ROOTDR(I)
    3 CONTINUE
      RETURN
      END
```

```
      OVERLAY(MCPILOT,4,0)
      PROGRAM AUXIL
C     THIS PROGRAM CALCULATES THE SS. FREQUENCY RESPONSE OF A TRANSFER
C     FUNCTION INPUT AS A NUMERATOR/DENOMINATOR POLYNOMIAL RATIO.
C     USER INPUTS ORDERS NUM & NDEN,COEFNS & COEFDS IN DESCENDING ORDER,
C     AND IFREQ OMEGAS, (RADIANS/SEC).
C     ******************************************************************
      COMMON COMON(4400)
      COMMON / 6 / SAVE(310)
      DIMENSION STFNUM(8,37),OMEGA(21),TFDEN(37),ISEAUX(8),IORDER(8)
      DIMENSION COEFN(37),COEFD(37),TITLE(10)
      EQUIVALENCE (STFNUM(1,1),SAVE(15))
      EQUIVALENCE (IORDER(1),SAVE(1))
      EQUIVALENCE (NORDEN,COMON( 517))
      EQUIVALENCE (INDCLS,COMON( 725))
      EQUIVALENCE (NISEAP,COMON(4320))
      EQUIVALENCE (IFREQ ,COMON(4330))
      EQUIVALENCE (ILINK ,COMON( 748))
      EQUIVALENCE (ISEAUX,COMON(4321))
      EQUIVALENCE (TITLE ,COMON( 795))
      EQUIVALENCE (TFDEN ,COMON( 823))
      EQUIVALENCE (OMEGA ,COMON(4296))
      EQUIVALENCE (PR    ,COMON(4345))
      REAL MAGN,MAGNDB
      COMPLEX GNUM,GDEN,GTF,S
      COMON(725)=COMON(4317)
      CALL PRATEI
      WRITE(6,5000)PR
 5000 FORMAT(1H ,5(/),42X,47(*+*)/42X,*+*,45X,*+*/42X,*+*,4X,*THE CALCUL
     .ATED PILOT RATING IS *,F6.3,4X,*+*/42X,*+*,45X,*+*/42X,47(*+*))
      IF(NISEAP.EQ.0)GO TO 999
      NDEN=NORDEN
      MDEN=NDEN+1
      DO 4 K=1,MDEN
 4    COEFD(MDEN-K+1)=TFDEN(K)
      M=1
C     FIND OUT WHICH ISE TERM
      DO 700 L=1,8
      IF(L.NE.ISEAUX(M))GO TO 699
      CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE
 6000 FORMAT(1H ,20X,10A6)
       WRITE(6,6002)L
 6002 FORMAT(1H ,20X,* FREQUENCY RESPONSE FOR ISE NO. *,I2)
 1    CONTINUE
      NUM=IORDER(L)
      MUM=NUM+1
      WRITE(6,410)NUM
 410  FORMAT(/2X,*THE INPUT NUMERATOR POLYNOMIAL IS OF ORDER *,I2)
      DO 3 K=1,MUM
 3    COEFN(MUM-K+1)=STFNUM(L,K)
      WRITE(6,411)(COEFN(II),II=1,MUM)
```

```
411   FORMAT(2X,*ITS COEFFICIENTS IN DESCENDING ORDER ARE-*,/6(3X,1PE15.
     .7))
      WRITE(6,412)NDEN
412   FORMAT(/2X,*THE INPUT DENOMINATOR POLYNOMIAL IS OF ORDER *,I2)
      WRITE(6,411)(COEFD(JJ),JJ=1,MDEN)
      WRITE(6,600)
      DO 102 I=1,IFREQ
      S=CMPLX(0.,OMEGA(I))
      GNUM=COEFN(1)
      IF(MUM.LT.2)GO TO 60
      DO 50 J=2,MUM
50    GNUM=COEFN(J)+GNUM*S
60    CONTINUE
      GDEN=COEFD(1)
      IF(MDEN.LT.2)GO TO 70
      DO 75 J=2,MDEN
75    GDEN=COEFD(J)+GDEN*S
70    CONTINUE
      GTF=GNUM/GDEN
      MAGN=CABS(GTF)
      MAGNDB=20.*ALOG10(MAGN)
      PHASE=57.29578*ATAN2(AIMAG(GTF),REAL(GTF))
100   WRITE(6,601)OMEGA(I),GTF,MAGN,MAGNDB,PHASE
102   CONTINUE
600   FORMAT(1H0,6X,*OMEGA*,6X,*REAL PART*,5X,*IMAG PART*,8X,*MAGN*,9X,*
     1MAGNDB*,6X,*PHASE-DEG*)
601   FORMAT(2X,F10.4,1X,5(3X,1PE11.4))
      M=M+1
      IF(M.GT.NISEAP)GO TO 701
699   CONTINUE
700   CONTINUE
701   CONTINUE
      CALL TIMRP
999   CONTINUE
      ILINK=5
C     CALL MATLOC
      CALL OVRLAY
      STOP
      END
```

```
      SUBROUTINE TIMRP
C
C     VARIABLE NAME DICTIONARY
C
C     COEFN IS THE COEFFICIENTS OF THE NUMERATOR IN DESCENDING ORDER
C     KORDN IS THE ORDER OF THE NUMERATOR
C     COEFD IS THE COEFFICIENTS OF THE DENOMINATOR IN DESCENDING ORDER
C     NDEN IS THE ORDER OF THE DENOMINATOR
C     R IS READ FROM GRAPH
C     NOTE THAT Y IS YDOT(1), THE FIRST DIFFERENTIAL OF Y IS YDOT(8), ET
C
      COMMON COMON(4400)
      COMMON / 6 / SAVE(310)
      COMMON / INTVAR / YDOT(259)
      COMMON / CINT / T,HMAX,HMIN,EMIN,EMAX,XS(8),HZ,CUTERR,IP,IV,
     1                M,IPR,IFIN,IVAL,IAD(8),IND(8),J
      DIMENSION TIME(60),RR(60)
      DIMENSION COEFN(37),COEFD(37),TITLE1(10)
      DIMENSION STFNUM(8,37),TFDEN(37),ISEAUX(8),IORDER(8)
      DIMENSION TPLOT(400,1),RPLOT(400,1),CPLOT(400,1),MRAY(1),ARAY(1),
     1  TITLE(10)
      EQUIVALENCE (STFNUM(1,1),SAVE(15))
      EQUIVALENCE (IORDER(1),SAVE(1))
      EQUIVALENCE (INDTRO,COMON( 506))
      EQUIVALENCE (HMAX1 ,COMON( 507))
      EQUIVALENCE (NORDEN,COMON( 517))
      EQUIVALENCE (INDCLS,COMON( 725))
      EQUIVALENCE (NISEAP,COMON(4320))
      EQUIVALENCE (NTIME ,COMON(4329))
      EQUIVALENCE (ISEAUX,COMON(4321))
      EQUIVALENCE (TITLE1,COMON( 795))
      EQUIVALENCE (TFDEN ,COMON( 823))
      EQUIVALENCE (TIME  ,COMON(4176))
      EQUIVALENCE (RR    ,COMON(4236))
      EQUIVALENCE (MOI   ,COMON(4379))
      DATA (TITLE(I),I=1,10)/5*10H          ,10HVS TIME
     1                     4*10H          /
      DATA TI,TO/10H    INPUT ,10H   OUTPUT /
      DATA ARAY(1)/1HA/
      HMAX=HMAX1
      NDEN=NORDEN
      N=NDEN*7+1
      NCOEFD=NDEN+1
      DO 4 IK=1,NCOEFD
    4 COEFD(NCOEFD-IK+1)=TFDEN(IK)
      IF(HMAX.LE.0..OR.HMAX.GT.100.)HMAX=.02
      HMIN = HMAX * 2. ** (-15)
      EMIN=1.D-6
      EMAX=1.D-4
      IP=5
      IV=0
C     IV=0 IMPLIES A VARIABLE INTERVAL SIZE CONTROLLED BY INTEG AND
```

```
C       UPDATE - APPLICABLE TO PREDICTOR-CORRECTOR ONLY
C       IV=1 IMPLIES A CONSTANT INTERVAL SIZE
C
C       M=0 IMPLIES THAT PREDICTOR-CORRECTOR IS TO BE USED AND
C       M=1 IMPLIES THAT RUNGE-KUTTA IS TO BE USED
        M=MOI
        MM=1
        DO 700 L=1,8
        IF(L.NE.ISEAUX(MM))GO TO 700
        CALL CLASS(INDCLS)
        WRITE(6,6000)TITLE1
 6000   FORMAT(1H ,20X,10A6)
         WRITE(6,6002)L
 6002   FORMAT(1H ,20X,* TIME RESPONSE FOR ISE NO. *,I2)
        DO 5 I=1,259
 5      YDOT(I)=0.
        KOUNT=0
        KORDN=IORDER(L)
        NCOEFN=KORDN+1
        DO 3 IK=1,NCOEFN
 3      COEFN(NCOEFN-IK+1)=STFNUM(L,IK)
        IF(INDTRO.EQ.1)GO TO 30
        WRITE(6,6003)
 6003   FORMAT(1H ,20X,* USING UNIT STEP INPUT *)
        GO TO 40
 30     WRITE(6,6004)
 6004   FORMAT(1H ,20X,* USING UNIT IMPULSE INPUT *)
        KORDN=KORDN+1
        NCOEFN=KORDN+1
        COEFN(NCOEFN)=0.
 40     K=KORDN*7+1
        WRITE(6,410)KORDN
 410    FORMAT(//2X,*THE INPUT NUMERATOR POLYNOMIAL IS OF ORDER *,I2,/12X,
       **----------*)
        WRITE(6,411)(COEFN(II),II=1,NCOEFN)
 411    FORMAT(2X,*ITS COEFFICIENTS IN DESCENDING ORDER ARE-*,/6(3X,1PE15.
       .7))
        WRITE(6,412)NDEN
 412    FORMAT(//2X,*THE INPUT DENOMINATOR POLYNOMIAL IS OF ORDER *,I2,/12
       *X,*------------*)
        WRITE(6,411)(COEFD(II),II=1,NCOEFD)
        WRITE(6,2000)
 2000   FORMAT(1H )
        T=0.
        CALL SETUP(DUMMY,DUMMY)
        DO 50 I=1,N,7
        CALL INUPD(I,DUMMY,DUMMY)
 50     CONTINUE
 90     CONTINUE
        CALL DISCO12(T,DUM,TIME,RR,DUM,1,NTIME,DUM,R,DUM)
        NN=NDEN*7-6
        DSUM=0.
```

```
      DO 100 I=1,NDEN
      DSUM=DSUM+COEFD(I+1)*YDOT(NN)
      NN=NN-7
100   CONTINUE
      YDOT(N)=(R-DSUM)/COEFD(1)
      NN=N
121   CALL INTEG(YDOT(NN),YDOT(NN-7))
      NN=NN-7
      IF(NN.GE.8)GO TO 121
      IF(IPR.NE.C)GO TO 15C
      C=COEFN(KORDN+1)*YDCT(1)
      IF(KORDN.EC.C)GO TO 130
      KK=K
      DO 120 I=1,KORDN
      C=C+COEFN(I)*YDOT(KK)
      KK=KK-7
120   CONTINUE
130   CONTINUE
      IF(KOUNT.GE.4CC)GO TO 140
      KOUNT=KOUNT+1
      CPLOT(KOUNT,1)=C
      TPLOT(KOUNT,1)=T
      RPLOT(KOUNT,1)=R
140   CONTINUE
      WRITE(6,125)T,R,C
125   FORMAT(1H 5X,*T=*,F6.3,3X,*R=*,F12.5,3X,*C=*,F12.7)
150   CONTINUE
      CALL UPDAT(DUMMY,DUMMY)
      IF(T.GT.TIME(NTIME))GC TC 190
      GO TO 90
190   CONTINUE
      MRAY(1)=KOUNT
      TITLE(5)=TI
      CALL PLOTR(TPLOT,RPLOT,MRAY,ARAY,1,7,0,1,0,TITLE,400,6)
      TITLE(5)=TC
      CALL PLOTR(TPLOT,CPLOT,MRAY,ARAY,1,7,0,1,0,TITLE,400,6)
200   CONTINUE
      MM=MM+1
      IF(MM.GT.NISEAP)RETURN
700   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE PCCLT (   YD, Y   )
      DIMENSION          XMN1(8),X(8),                     Y(7),YD(7)
      COMMON / INTVAR / COMINT(1)
      COMMON / IZZZRM / IPTATL,IPTOTL, LOCNAM(50)
      COMMON /  CINT   / T, HMAX, HMIN, EMIN, EMAX, XS(8), HZ,  CUTERR,
     1                   IP, IVARH, IMTH, IPRNT, IFIN, IVAL, IAD(8),
     2                IND(8), J
C   C1=55/24     C2=9/24        C3=1/6     C4=-59/55       C5=-37/59
C   C6=-9/37     C7=19/9        C8=-5/19  C9=19/270
      DATA C1/2.291666666666667/,C2/.375/,C3/.1666666666666667/
      DATA C4/-1.072727272727272727/,C5/-.6271186440677797/
      DATA C6/-.243243243243243/,C7/2.111111111111111/
      DATA C8/-.2631578947368842/,C9/.0703703703703370/
      ENTRY SETUP
      IPTATL  = C
      IPTOTL  = C
      IERROR  = 0
      ISTEP=1
      I=0
      IVAL=0
      HZ       = HMAX*2.**(-IP)
      IPT2=2**IP
      IPT1=0
      IPRNT=0
      IFIN=0
      INCRH   =  1
      IB5=1
      IB1=1
      IALP=4
      LIST=0
      INUPD=0
      IF(IMTH)1CC,10,100
   10 IB2=1
      JJ=7
      IB3=2
      ISCNT=C
      IBETA=3
      IGAM=-1
      IF(IVARH) 2C,3C,20
   20 IB4=1
      IB6=2
      GO TO 40
   30 IB4=2
      IB6=1
   40 IB7=1
      GO TO 110
  100 IB2=2
      JJ=6
      IB3=1
      IB7=2
  110 HD2     = HZ / 2.
      H=HD2
```

```
      A1=C1*HZ
      A2=C2*HZ
      RKTME=T
      RETURN
      ENTRY INTEG
      GO TO(120,160,200,240),IB5
  120 GO TO(125,130,140,125),ISTEP
  125 Y(JJ)=Y(1)+H*YD(1)
      RETURN
  130 Y(JJ)=Y(2)+H*YD(1)
      RETURN
  140 Y(JJ)=Y(3)+H*YD(1)
      RETURN
  160 Y(JJ)=Y(4)+C3*H*(YD(4)+2.00*(YD(3)+YD(2))+YD(1))
      RETURN
  200 Y(6)=Y(1)+A1*(YD(1)+C4*(YD(2)+C5*(YD(3)+C6*YD(4))))
      RETURN
  240 CN1=Y(2)+A2*(YD(1)+C7*(YD(2)+C8*(YD(3)-YD(4)/5.00)))
      XM= ABS(Y(1)-CN1)
      Y(1)=CN1+C9*(Y(1)-CN1)
      IF(IB6-2)260,250,260
  250 RETURN
  260 IF(CN1)280,290,280
  280 XM1     = ABS( XM / CN1 )
      IF(XM-XM1)290,290,285
  285 XM=XM1
  290 IF(XM-EMAX)310,300,300
  300 IF ( HZ .GT. HMIN ) IVAL = -8300
      IVAL = IVAL + 1
      RETURN
  310 IF(XM-EMIN)330,320,320
  320 IVAL=IVAL+1
  330 RETURN
      ENTRY UPDAT
      IFIN=1
      IF(IPRNT)350,340,350
  340 IPT1=IPT2
  350 IF(IB1-2)360,800,360
  360 IF(IALP-1)700,370,700
  370 I=I+1
      IPT1=IPT1-1
      GO TO(400,380),IB2
  375 IBETA=IBETA-1
  380 IALP=4
      ISTEP=1
      H = H / 2.
      IFIN=0
      IB5=1
      GO TO(450,390),IB3
  390 DO 391 IMVER = 1,IPTOTL
      KMVER  = LOCNAM( IMVER )
      COMINT( KMVER + 1 ) = COMINT( KMVER + 3 )
```

```
      COMINT( KMVER + 2 ) = COMINT( KMVER + 4 )
      COMINT( KMVER + 3 ) = COMINT( KMVER + 5 )
      COMINT( KMVER + 4 ) = COMINT( KMVER + 5 )
  391 COMINT( KMVER     ) = COMINT( KMVER + 6 )
      GO TO 455
  450 DO 451 IMVER = 1,IPTOTL
      KMVER  = LOCNAM ( IMVER )
      COMINT( KMVER     ) = COMINT( KMVER + 5 )
  451 COMINT( KMVER + 1 ) = COMINT( KMVER + 3 )
  455 IPRNT=IPT1
      RETURN
  400 IF(IBETA-1)375,460,375
  460 IB1=2
      IB5=3
      IFIN=0
      IF(IVARH)465,390,465
  465 IB6=2
      GO TO 390
  700 IALP=IALP-1
      IF(IALP-1)710,720,710
  710 ISTEP=ISTEP+1
      GO TO 730
  720 IB5=2
  730 IF(IALP-2)740,750,740
  740 T=T+HD2
      GO TO 760
  750 H=HZ
  760 GO TO(770,780),IB7
  770 DO 771 IMVER = 1,IPTOTL
      KMVER  = LOCNAM( IMVER )
      COMINT( KMVER + 5 ) = COMINT( KMVER + 4)
      COMINT( KMVER + 4 ) = COMINT( KMVER + 3)
      COMINT( KMVER + 3 ) = COMINT( KMVER + 2)
      COMINT( KMVER + 2 ) = COMINT( KMVER + 1)
      COMINT( KMVER + 1 ) = COMINT( KMVER    )
  771 COMINT( KMVER     ) = COMINT( KMVER + 6)
      GO TO 455
  780 DO 781 IMVER = 1,IPTOTL
      KMVER  = LOCNAM( IMVER )
      COMINT( KMVER + 3 ) = COMINT( KMVER + 2)
      COMINT( KMVER + 2 ) = COMINT( KMVER + 1)
      COMINT( KMVER + 1 ) = COMINT( KMVER    )
  781 COMINT( KMVER     ) = COMINT( KMVER + 5)
      GO TO 455
  800 IGAM=-IGAM
      IF(IGAM)820,820,810
  810 IB5=4
      T=T+H
      GO TO 780
  820 IB5=3
      GO TO(830,840),IB4
  830 IPT1=IPT1-1
```

```
      IFIN=0
      GO TO 856
840   IF(IVAL) 845,860,845
845   ISCNT=0
      IF(IVAL)900,900,855
855   INDRH=1
      IPT1=IPT1-1
      I=I+1
      IFIN=0
856   IVAL=0
      GO TO 455
860   IF(ISCNT-2)865,865,870
865   ISCNT=ISCNT+1
      GO TO 855
870   IF ( 2*(IPT1/2) .EQ. IPT1 ) GO TO 855
875   IF(H-HMAX)880,855,855
880   IPT1=IPT1/2
      ISCNT=0
      IBETA=3
      IALP=4
      IB1=1
      IB5=1
      ISTEP=1
      INDRH=0
      IPT2   =   IPT2/ 2
      RKTME=T
      HD2=H
      HZ      = 2. * H
      IFIN=0
885   I=C
      A1=C1*HZ
      A2=C2*HZ
      GO TO 856
900   IF(IPT1)855,905,905
905   IBETA=3
      IALP=4
      ISTEP=1
      IB1=1
      IB5=1
      IF(I-3)910,920,910
910   T=T-H
      RKTME=T
      IPT1=2*IPT1
      DO 911 IMVER = 1,IPTOTL
      KMVER  = LCCNAM( IMVER )
911   COMINT( KMVER ) = CCMINT( KMVER + 1)
      GO TO 930
920   T=RKTMF
      IPT1=2*(IPT1+3)
      DO 921 IMVER = 1,IPTOTL
      KMVER  = LCCNAM( IMVER )
921   COMINT( KMVER ) = COMINT( KMVER + 4)
```

```
 930 HZ = H/2.
     IF ( HZ .LT. HMIN    ) HZ = HMIN
     HD2 = HZ / 2.
     H=HD2
     IPT2   =   2*IPT2
     INDRH=-1
     GO TO 885
     ENTRY CUT
     IF(IFIN)940,950,940
 940 J=0
     IERROR = 1
     RETURN
 950 K=1
 955 IF ( K   .LE. IPTATL ) GO TO 980
 960 IF(K-1)965,940,965
 965 IK=K-1
     DO 970 I = 1,IK
     KMVER   = IAD(I)
     XMN1(I)= COMINT( KMVER )
 970 CONTINUE
     GO TO 940
 980 KMVER   = IAD(K)
     X(K)    = COMINT( KMVER )
     XU      = XS(K) + (ABS(XS(K))+ 1.)*CUTERR
     XL      = XS(K) - (ABS(XS(K))+ 1.)*CUTERR
     IF(IND(K))985,1005,985
 985 IF(X(K)-XU)990,1010,1010
 990 IF(X(K)-XL)1040,1040,1035
1005 IF(X(K)-XL)1010,1010,1020
1010 K=K+1
     IF ( K .LT. 9 ) GO TO 955
     GO TO 965
1020 IF ( X(K) - XU ) 1035, 1040, 1040
1035 J=K
     IERROR = 1
     RETURN
1040 IF ( IERROR .NE. 0 ) GO TO 1054
     WRITE(6,1051)K
1051 FORMAT(1H0,28H********** CUTOFF PASSED BY ,I1,56HTH CUTOFF VARIABL
    1E ON THE INITIAL CALL TO CUT **********      )
     STOP
1054 CONTINUE
     T = T-HZ
     HZ      = ( HZ*(XS(K)-XMN1(K))/(X(K)-XMN1(K)))/ 2.
     IF(INDRH)1050,1045,1050
1045 HZ      = HZ / 2.
1050 I=0
     HD2     = HZ / 2.
     H       = HD2
     A1=C1*HZ
     A2=C2*HZ
     ISCNT=0
```

```
      IBETA=3
      IALP=4
      ISTEP=1
      IB1     =   1
      IB2     =   2
      IB3     =   1
      IB5=1
      IB7     =   2
      JJ      =   6
      IPT1    =   1000
      IPT2    =   1000
      IPRNT   =   1000
      RKTMF=T
      DO 1055 IMVER = 1,IPTOTL
      KMVFR   = LOCNAM( IMVER )
 1055 COMINT( KMVER ) = COMINT( KMVER + 1)
      IFIN=1
      J=-1
      IERROR = 1
      RETURN
      END
```

```
      SUBROUTINE INITUP ( IYD, CLTVAL, IDIRT  )
      COMMON / IZZZRM / IPTATL,IPTOTL, LOCNAM(50)
      COMMON /   CINT   / T, FMAX, HMIN, EMIN, EMAX, XS(8), HZ,  CUTERR,
     1                 IP, IVARH, IMTH, IPRNT, IFIN, IVAL, IAD(8),
     2              IND(8), J
      ENTRY LOC
      IPTATL = IPTATL + 1
      IF ( IPTATL .LE. 8 ) GO TO 3
      WRITE (6,1)
    1 FORMAT ( 65H ----------JOB TERMINATED, MORE THAN EIGHT CALLS TO LO
     *C----------)
      STOP
    3 CONTINUE
      IAD( IPTATL )    = IYD
      XS(  IPTATL  ) = CLTVAL
      IND( IPTATL  ) = IDIRT
      RETURN
      ENTRY INUPD
      IPTOTL = IPTOTL + 1
      IF ( IPTOTL .LE. 50 ) GO TO 4
      WRITE (6,2)
    2 FORMAT(66H----------JOB TERMINATED, MORE THAN FIFTY CALLS TO INUPD
     *----------)
      STOP
    4 LOCNAM( IPTOTL ) = IYD
      RETURN
      END
```

```
      SUBROUTINE PLOTR(X,Y,LX,A,IC,B,MQ,LL,XQELT,TITLE,IDM,TAPEN)
C PLOTR, A  POINT PLOT ROUTINE ADAPTED FROM 360 PLOT
C PLOTS A GRAPH OF ONE OR MORE CURVES FROM GIVEN SETS OF RECTANGULAR COO
C
C CALL PLOTR(X,Y,M,A,IC,B,MP,LL,XDELT,TITLE,IDM)
C
C  X      NAME OF A 2 DIMENSIONAL ARRAY CONTAINING THE X COORDINATES OF
C         ALL THE CURVES TO BE PLOTTED. THE X COORDINATES OF THE N-TH
C         CURVE,FOR EXAMPLE, ARE STORED FROM X(1,N) TO X(M,N) WHERE M IS
C         THE NUMBER OF POINTS IN THE CURVE.
C
C  Y      HAS THE SAME DIMENSIONS AS X AND CONTAINS THE Y COORDINATES.
C
C  M      NAME OF ONE DIMENSIONAL ARRAY SET UP BY THE USER.
C         M(N) IS THE NUMBER OF POINTS IN THE N-TH CURVE, WHOSE FIRST
C         POINT IS AT X(1,N) AND Y(1,N).
C
C  A      NAME OF ONE DIMENSIONAL ARRAY SET UP BY THE USER.
C         A(N) IS THE CHARACTER--LEFT ADJUSTED IN A 4 BYTE WORD--TO BE
C         USED IN PLOTTING THE N-TH CURVE.
C
C  IC     THE NUMBER OF CURVES TO BE PLOTTED.  IT MUST BE LESS THAN OR
C         EQUAL TO THE SECOND DIMENSION SPECIFIED FOR THE X AND Y ARRAYS
C         IN THEIR DIMENSION STATEMENT.
C  B      1  NO BORDER, NO AXIS
C         2  BORDER, NO AXIS
C         3  NO BORDER, X AXIS ONLY
C         4  BORDER, X AXIS ONLY
C         5  NO BORDER, Y AXIS ONLY
C         6  BORDER, Y AXIS ONLY
C         7  NO BORDER, BOTH AXES
C         8  BORDER, BOTH AXES
C
C  MP     0 SINGLE PAGE PLOT DESIRED
C         1 MULTIPLE PAGE OR FRACTION OF PAGE PLOT DESIRED
C
C  L      1 PLOT GIVEN POINTS
C         2 SEMILOG (PLOT LOGS OF Y COORDINATES)
C         3 LOG-LOG (PLOT LOGS OF X AND Y COORDINATES)
C
C XDELT  0  INDICATES DELTA X IS TO BE CALCULATED
C            OTHERWISE SPECIFY DELTA X IN FLOATING POINT.
C
C  TITLE  NAME OF THE ARRAY IN WHICH THE TITLE TO HEAD EACH PAGE IS
C         STORED.  TEN WORDS ARE ALWAYS PRINTED.
C
C  IDM    THE FIRST DIMENSION SPECIFIED FOR THE X AND Y ARRAYS IN THEIR
C         DIMENSION STATEMENT.  IT MUST BE AT LEAST EQUAL TO THE NUMBER
C         OF POINTS IN THE CURVE WHICH HAS THE GREATEST NUMBER OF POINTS
C         OF THE CURVES TO BE PLOTTED.
C
C
```

```
C   TAPEN   OUTPUT TAPE FOR PLOTR
C
C
       COMMON COMCN(4400)
       EQUIVALENCE (INDCLS,COMON( 725))
       EQUIVALENCE (TITLE1,COMON( 795))
       DIMENSION TITLE1(1C)
       DIMENSION X(1),Y(1),LX(1),A(1),PLINE(11),XAX(6),TITLE(1)
       INTEGER A,SS,PLINE,B,BLNK,PLUS,TAPEN
       DATA PLUS,MINUS,II,BLNK /1H+,1H-,1HI,10H
    11 FORMAT(1H+,60X,6A10)
    12 FORMAT(17H  SCALE/INCH    X=,1PE14.7,5H    Y=,1PE14.7,10X,25H+OR- TO
      .LERANCE/POINT    X=,1PE14.7,5H    Y=,1PE14.7)
    13 FORMAT(11X,11A10)
    14 FORMAT(1X,E9.2,1X,11A10)
    15 FORMAT(11X,5(1H+,19X),1H+/7X,5(E9.2,11X),E9.2)
    16 FORMAT(17H      MINIMUM    X=,1PE14.7,5H    Y=,1PE14.7,23X,12HMAXIMUM
      .    X=,1PE14.7,5H    Y=,1PE14.7)
       XMAX=X(1)
       YMAX=Y(1)
       XDELT=ABS(XDELT)
       MP=MQ
       IF(MP.EQ.1.AND.XDELT.EQ.0.)MP=0
C   INITIALIZATION IS NOW COMPLETE
C   NOW CURVES WILL BE SEARCHED FOR XMAX,XMIN,YMAX,YMIN,AND XDELT=MINIMUM
C   DISTANCE BETWEEN ANY TWO PCINTS ON ANY CURVE
       GO TO(3,2,1),LL
     1 XMAX=ABS(XMAX)
     2  YMAX=ABS(YMAX)
     3  XMIN=XMAX
       YMIN=YMAX
       L=2
       IF(XDELT .NE. C.) GO TO 5
     4  L=1
     5  DO 400 J=1,IC
       N=1+IDM*(J-1)
       XCMAX=X(N)
       YCMAX=Y(N)
       GO TO(22,21,2C),LL
    20 XCMAX=ABS(XCMAX)
    21 YCMAX=ABS(YCMAX)
    22 XCMIN=XCMAX
       YCMIN=YCMAX
       GO TO(23,27),L
    23 XCDELT=0
    27 NEL=LX(J)
       DO 300 I=1,NEL
       N=I+IDM*(J-1)
       XTEMP=X(N)
       YTEMP=Y(N)
       GO TO(30,29,28),LL
    28 XTEMP=ABS(XTEMP)
```

```
   29 YTEMP=ABS(YTEMP)
   30 XCMAX=AMAX1(XTEMP,XCMAX)
      XCMIN=AMIN1(XTEMP,XCMIN)
      YCMAX=AMAX1(YTEMP,YCMAX)
      YCMIN=AMIN1(YTEMP,YCMIN)
      GO TO(31,300),L
   31 IF(I .GE. NEL) GO TO 300
  311 M=I+1
      DO 100 K=M,NEL
      N=K+IDM*(J-1)
      GO TO(32,32,33),LL
   32 XDIFF=XTEMP-X(N)
      GO TO 335
   33 XDIFF=XTEMP/ABS(X(N))
      IF(XDIFF .EQ. 0.) GO TO 100
  332 XDIFF=ALOG10(XDIFF)
  335 XDIFF=ABS(XDIFF)
      IF(XDIFF .EQ. 0.) GO TO 100
  336 IF(XCDELT .NE. 0.) GO TO 338
  337 XCDELT=XDIFF
      GO TO 100
  338 XCDELT=AMIN1(XDIFF,XCDELT)
  100 CONTINUE
  300 CONTINUE
      XMAX=AMAX1(XMAX,XCMAX)
      XMIN=AMIN1(XMIN,XCMIN)
      YMAX=AMAX1(YMAX,YCMAX)
      YMIN=AMIN1(YMIN,YCMIN)
      GO TO(34,400),L
   34 IF(XDELT  .NE. 0.) GO TO 346
  345 XDELT=XCDELT
      GO TO 400
  346 XDELT=AMIN1(XDELT,XCDELT)
  400 CONTINUE
C  IF XDELT IS STILL ZERO THEN THE CURVES ARE PARALLEL TO THE Y AXIS SO
C   XMAX,XMIN AND XDELT ARE CALCULATED BY OTHER MEANS DESIGNED TO CENTER
C   CURVES ON A SINGLE PAGE
      IF(XDELT  .NE. 0.) GO TO 365
  347 IF(XMAX)354,353,352
  353 XMAX=10.+XMAX
      XMIN=XMIN-10.
      GO TO 348
  352 XMIN=XMIN-XMAX
      XMAX=2.*XMAX
      GO TO 348
  354 XMAX=XMAX-XMIN
      XMIN=2.*XMIN
      GO TO 348
  365 IF(XMAX .NE. XMIN) GO TO 3365
 3366 XMIN=XMIN-54.*XDELT
      XMAX=XMAX+54.*XDELT
  348 MP=0
```

```
C   IF THE CURVES ARE PARALLEL TO THE Y AXIS,A SIMILAR PROCEDURE IS
C     FOLLOWED FOR YMAX,YMIN,ANDYDELT
 3365 IF(YMAX .NE. YMIN) GO TO 366
  349 IF(YMAX)362,363,364
  363 YMAX=10.+YMAX
      YMIN=YMIN-10.
      GO TO 351
  364 YMAX=2.*YMAX
      YMIN=YMIN-YMAX
      GO TO 351
  362 YMIN=2.*YMIN
      YMAX=YMAX-YMIN
  351 MP=0
  366 GO TO(37,358,35),LL
   35 IF(XMAX .EQ. C.) GO TO 356
  355 XMAX=ALOG10(XMAX)
  356 IF(XMIN .EQ. C.) GO TO 358
  357 XMIN=ALOG10(XMIN)
  358 IF(YMAX.EQ.C.) GO TO 359
   36 YMAX=ALOG10(YMAX)
  359 IF(YMIN.EQ.C.) GO TO 37
  361 YMIN=ALOG10(YMIN)
   37 YRANGE=YMAX-YMIN
      XRANGE=XMAX-XMIN
      PRANGE=XDELT*10C.
C   IF THE SINGLE PAGE OPTION IS DESIRED BUT XDELT IS TOO LARGE FOR THE
C     RANGE OF X VALUES TO FIT ON ONE PAGE,THEN A NEW XDELT MUST BE FOUND
      IF(MP .NE. 0.) GO TO 376
  374 IF(XRANGE .EQ. PRANGE) GO TO 376
  375 XDELT=XRANGE/108.
      PRANGE=XRANGE
  376 NPAGE=XRANGE/PRANGE+1.
      YDELT=YRANGE/5C.
      YSF=YRANGE/8.33
      XSF=PRANGE/10.8
      XTP=XDELT/2.
      YTP=YDELT/2.
      PXMAX=XMIN+XTP
C   NOW THE PLOT IS FORMED A LINE AT A TIME, SEARCHING EACH CURVE ONCE
C     FOR EVERY LINE ON EVERY PAGE AND PRINTING OUT EACH LINE AS SOON AS
C   IT IS FORMED
 39   DO 900 K=1,NPAGE
      IF(PXMAX-XMAX .GE. XTP) GO TO 73
 40   IF(PXMAX .LE. XMAX) GO TO 401
 404  IF(K-1)375,375,73
 401   PXMIN=PXMAX-XDELT
      PXMAX=PXMAX+PRANGE
      IF(K .NE. NPAGE) GO TO 402
 403  PXMAX=XMAX+XTP
 402  GO TO(42,42,41),LL
 41   PNXMN=10.**PXMIN
      PNXMX=10.**PXMAX
```

```
42    CALL CLASS(INDCLS)
      WRITE(6,6000)TITLE1
6000  FORMAT(1H ,20X,10A6)
      WRITE(6,11) (TITLE(I),I=1,6)
      WRITE(TAPEN,16) XMIN,YMIN,XMAX,YMAX
      WRITE(TAPEN,12) XSF,YSF,XTP,YTP
      GO TO(44,43,43,43,44,43,43,43),B
43    XTEMP=PXMAX-XTP
      JJJ=(AMIN1(XTEMP,XMAX)-PXMIN)/XDELT
44    YUB=YMAX+YTP
      YLB=YMAX-YTP
      DO 700 LLINE=1,51
      LINE=52-LLINE
      DO 450 I=1,11
450   PLINE(I)=BLNK
      GO TO(46,45,45),LL
45    YNUB=10.**YUB
      YNLB=10.**YLB
46    GO TO(51,47,511,47,519,47,511,47),B
47    N=JJJ/10
      NCHAR=MOD(JJJ,10)
      IF(LINE-1)50,49,48
48    IF(LINE .NE. 51) GO TO 50
49    IF(N .LE. 0) GO TO 549
5549  DO 500 I=1,N
500   PLINE(I)=10H++++++++++
549   NCHAR=NCHAR+1
      DO 550 JJ=1,NCHAR
      J=JJ-1
550   CALL PACK(PLINE(N+1),PLUS,J)
      GO TO 5118
50    CALL PACK(PLINE(1),PLUS,0)
      CALL PACK(PLINE(N+1),PLUS,NCHAR)
5118  GO TO(51,51,511,511,519,519,511,511),B
511   IF(YUB)512,513,514
514   IF(YLB .GE. 0.) GO TO 512
513   N=JJJ/10
      NCHAR=MOD(JJJ,10)+1
      IF(N .LE. 0) GO TO 5515
5513  DO 515 I=1,N
515   PLINE(I)=10H----------
5515  DO 521 I=1,NCHAR
      J=I-1
521   CALL PACK(PLINE(N+1),MINUS,J)
512   GO TO (51,51,51,51,519,519,519,519),B
519   IF(PXMAX)51,516,517
517   IF(PXMIN .GT. 0.) GO TO 51
516   I=-PXMIN/XDELT
      NCHAR=MOD(I,10)
      I=I/10+1
      CALL PACK(PLINE(I),II,NCHAR)
51    DO 600 J=1,IC
```

```
      NEL=LX(J)
      DO 600 I=1,NEL
      M=I+IDM*(J-1)
      GO TO(54,52,52),LL
52    YTEMP=ABS(Y(M))
      IF(YTEMP-YNLB)600,58,53
53    IF(YTEMP-YNUB)58,600,600
54    IF(Y(M)-YLB)600,56,55
55    IF(Y(M).GE.YUB) GO TO 600
56    XTEMP=X(M)
      IF(XTEMP-PXMIN)600,62,57
57    IF(XTEMP-PXMAX)62,62,600
58    GO TO(56,56,59),LL
59    XTEMP=ABS(X(M))
      IF(XTEMP-PNXMN)600,61,60
60    IF(XTEMP.GT.PNXMX) GO TO 600
61    IF(XTEMP.EQ.0.) GO TO 62
611   XTEMP=ALOG10(XTEMP)
62    JJ=(XTEMP-PXMIN)/XDELT
      GO TO(66,64,63),LL
63    IF(X(M).LT.0.) GO TO 65
64    IF(Y(M).GE.0.) GO TO 66
65    SS=1HN
      GO TO 67
66    SS=A(J)
67    NCHAR=MOD(JJ,10)
      N=JJ/10+1
      CALL PACK(PLINE(N),SS,NCHAR)
600   CONTINUE
68    IF(MOD(LINE,6).NE.1) GO TO 69
901   IF(LINE .NE. 1)GO TO 71
902   WRITE(TAPEN,14) YMIN,PLINE
      GO TO 715
69    IF(LINE .EQ. 51) GO TO 71
70    WRITE(TAPEN,13) PLINE
      GO TO 715
71    YTEMP=YLB+YTP
      WRITE(TAPEN,14) YTEMP,PLINE
715   YUB=YUB-YDELT
      YLB=YLB-YDELT
700   CONTINUE
72    DO 800 I=1,6
      J=I-1
      XAX(I)=PXMIN+XTP+XDELT*FLOAT(J)*20.
800   CONTINUE
      WRITE(TAPEN,15) XAX
900   CONTINUE
73    RETURN
      END
```

```
      SUBROUTINE PACK(WORD,CHAR,IPOS)
      DIMENSION SPLIT(10)
      DATA SPLIT/10*1H /
      DECODE(10,2,WORD) (SPLIT(I),I=1,10)
      SPLIT(IPOS+1)=CHAR
      ENCODE(10,2,WORD) (SPLIT(I),I=1,10)
      RETURN
    2 FORMAT(10A1)
      END
```

```
      SUBROUTINE DISCOT2(X,Y,TABX,TABZ,TABY,NS,NZ,NY,Z,NDIM)
C
C         DISCOT2 PROVIDES FOR LINEARLY INTERPOLATICN CF FUNCTIONS
C         OF ONE OR TWO INDEPENDENT VARIABLES.
C
C         CALL DISCOT2(X,Y,TABX,TABZ,TABY,NS,NZ,NY,Z,NDIM)
C
C         X          FIRST INDEPENDENT VARIABLE
C
C         Y          SECOND INDEPENDENT VARIABLE
C                    DUMMY IF THERE IS ONLY ONE INDEPENDENT VARIABLE
C
C         TABX       ARRAY OF VALUES FOR THE FIRST INDEPENDENT VARIABLE
C                    MUST BE IN ASCENDING ORDRE
C
C         TABZ       ARRAY OF VALUES FOR THE DEPENDENT VARIABLE
C
C         TABY       ARRAY OF VALUES FOR THE SECOND INDEPENDENT VARIABLE
C                    MUST BE IN ASCENDING ORDER
C                    DUMMY IF THERE IS ONLY ONE INDEPENDENT VARIABLE
C
C         NS         NUMBER OF INDEPENDENT VARIABLES
C                    NS=1      ONE INDEPENDENT VARIABLE
C                    NS=2      TWO INDEPENDENT VARIABLES
C
C         NZ         NUMBER OF ELEMENTS IN THE ARRAY OF
C                    DEPENDENT VARIABLES      (TABZ)
C
C         NY         NUMBER OF ELEMENTS IN THE Y ARRAY,  (TABY)
C
C                    NOT USED IF THERE IS ONLY ONE INDEPENDENT
C                    VARIABLE
C         Z          INTERPOLATED DEPENDENT VARIABLE
C
C         NDIM       DIMENSION OF I IN Z(I,J) AND X(I,J)
C                    THE FIRST SUBSCRIPT OF THE ARRAYS TABX
C                    AND TABZ WHEN THEY ARE TWO DIMENSION ARRAYS
C
C
      DIMENSION TABX(1),TABZ(1), TABY(1)
      NX=NZ
      IF(NS.EQ.2)  NX=NX/NY
      IF(X.LE.TABX(1))  GO TO 4
      IF(X.GE.TABX(NX)) GO TO 5
      DO 6 I=1,NX
      IX1=I
      IF(X.LE.TABX(I)) GO TO 7
  6   CONTINUE
  7   IX2=IX1-1
      GO TO 8
  4   IX1=1
      IX2=2
```

```
      GO TO 8
5     IX1=NX
      IX2=NX-1
8     IF(NS.GT.1)  GO TO 9
      DX=X-TABX(IX1)
      DZDX=(TABZ(IX2)-TABZ(IX1))/(TABX(IX2)-TABX(IX1))
      Z=TABZ(IX1)+DZDX*DX
      RETURN
9     IF(Y.LE.TABY(1)) GO TO 11
      IF(Y.GE.TABY(NY)) GO TO 12
      DO 13 I=1,NY
      IY1=I
      IF(Y.LE.TABY(I)) GO TO 14
13    CONTINUE
14    IY2=IY1-1
      GO TO 15
11    IY1=1
      IY2=2
      GO TO 15
12    IY1=NY
      IY2=NY-1
15    CONTINUE
      I=NDIM*(IY1-1)
      I1=I+IX1
      I2=I+IX2
      I=NDIM*(IY2-1)
      I3=I+IX1
      I4=I+IX2
      Z=TABX(IX2)-TABX(IX1)
      DZDX1=(TABZ(I2)-TABZ(I1))/Z
      DZDX2=(TABZ(I4)-TABZ(I3))/Z
      Z=TABY(IY2)-TABY(IY1)
      DZDYDX=(DZDX2-DZDX1)/Z
      DZDY=(TABZ(I3)-TABZ(I1))/Z
      DX=X-TABX(IX1)
      DY=Y-TABY(IY1)
      Z=TABZ(I1)+DZDX1*DX+DZDY*DY+DZDYDX*DX*DY
      RETURN
      END
```

```
      OVERLAY(MCPILOT,5,0)
      PROGRAM MATLOC
      COMMON COMCN(4400)
      CCMMON / 5 / A(12,12,11),B(1616)
      COMMON /RPLBLK/TEMP,ANSWER,ANS,NN,NA
      CGMMON /RTLOCS/DUM1(26),PR(1CO),DUM2(200),ZR(100),DUM3(207),
     .                   NORDP,NORD2,DUM4(95),GNFCTR,
     8          KDSTOR(2,1501)
      DIMENSION NN(10),TITLE(1C),P(5)
      DIMENSION BNDY(4),BNDYI(16),SAVE(5),ANS(121),ANSWER(11),TEMP(10,5)
      DIMENSICN TEMPOR(1CC),TMPRRY(100)
      DIMENSION KGAINI(4),JLOCUS(4),DELTI(4),TOLI(4),ICROSI(4)
      EQUIVALENCE (ACRDER,COMCN(    1))
      EQUIVALENCE (NCASE ,CCMON(4344))
      EQUIVALENCE (INDCLS,CCMCN( 725))
      EQUIVALENCE (JLOCUS,COMON(4354))
      EQUIVALENCE (P      ,COMCN( 780))
      EQUIVALENCE (TITLE ,COMCN( 795))
      EQUIVALENCE (KGAINI,COMON(4350))
      EQUIVALENCE (KGAIN ,CCMCN(4340))
      EQUIVALENCE (DELTI ,CCMCN(4358))
      EQUIVALENCE (DELT  ,COMON(4341))
      EQUIVALENCE (TOLI  ,COMON(4362))
      EQUIVALENCE (TOL   ,COMCN(4342))
      EQUIVALENCE (ICROSI,COMON(4366))
      EQUIVALENCE (ICRCSS,CCMON(4343))
      EQUIVALENCE (BNDYI ,CCMCN( 449))
      EQUIVALENCE (BNCY  ,COMCN(4346))
      INTEGER ACRDER(12,12)
      CATA IZERO/C/
      SAVE(1)=P(1)
      SAVE(2)=P(2)
      SAVE(3)=P(3)
      SAVE(4)=P(4)
      SAVE(5)=P(5)
      JSAV=0
      DO 500 L=1,NCASE
      IF(JLOCUS(L).EQ.C)GO TO 5C0
      P(1)=SAVE(1)
      P(2)=SAVE(2)
      P(3)=SAVE(3)
      P(4)=SAVE(4)
      P(5)=SAVE(5)
      KGAIN = KGAINI(L)
      DELT  = CELTI(L)
      TOL   = TOLI(L)
      ICROSS= ICROSI(L)
      DO 10 KLK=1,4
      JSAV=JSAV+1
10    BNDY(KLK)=BNDYI(JSAV)
      KLIK=JLOCUS(L)
      PLOCUS=0.
```

```
      P(KLIK)=PLOCUS
      CALL BLANK
      CALL CLASS(INDCLS)
      WRITE(6,600)TITLE
600   FORMAT(1H ,10X,10A6)
      WRITE(6,601)
601   FORMAT(1H+,80X,*A MATRIX FOR OPEN LOOP POLES POLYNOMIAL*)
      DO 30 I=1,12
      IF(I.EQ.5.OR.I.EQ.9)CALL CLASS(INDCLS)
      IF(I.EQ.5.OR.I.EQ.9)WRITE(6,600)TITLE
      WRITE(6,603)(AORDER(I,J),J=1,12)
603   FORMAT(/,1X,12(4X,I2,5X))
      DO 30 K=1,11
30    WRITE(6,604)  (A(I,J,K),J=1,12)
604   FORMAT(1X,12(1PE11.4))
      CALL SKBJI
      NORDP=NA
      NORDP1=NORDP+1
      NA=NA+1
      DO 40 I=1,NA
      TEMPOR(I)=ANS(I)
40    PR(I)=ANS(I)
      PLOCUS=1.
      P(KLIK)=PLOCUS
      CALL BLANK
      CALL CLASS(INDCLS)
      WRITE(6,600)TITLE
      WRITE(6,602)
602   FORMAT(1H+,78X,*A MATRIX FOR CHARACTERISTIC EQUATION--PLOCUS = 1*)
      DO 60 I=1,12
      IF(I.EQ.5.OR.I.EQ.9)CALL CLASS(INDCLS)
      IF(I.EQ.5.OR.I.EQ.9)WRITE(6,600)TITLE
      WRITE(6,603)(AORDER(I,J),J=1,12)
      DO 60 K=1,11
60    WRITE(6,604)(A(I,J,K),J=1,12)
      CALL SKBJI
      NA=NA+1
      CALL PSUB(TMPRRY,NORDZ1,ANS,NA,TEMPOR,NORDP1)
      DO 70 I=1,NORDZ1
70    ZR(I)=TMPRRY(I)
80    IF(ABS(ZR(NORDZ1)).GT.1.E-40)GO TO 90
      NORDZ1=NORDZ1-1
      IF(NORDZ1.LE.0)GO TO 250
      GO TO 80
90    NORDZ=NORDZ1-1
      CALL CLASS(INDCLS)
      WRITE(6,600)TITLE
      WRITE(6,611)
611   FORMAT(1H0,43X,'OPEN LOOP POLES POLYNOMIAL IN ASCENDING ORDER'/)
      WRITE(6,612)IZERO,PR(1),(I,PR(I+1),I=1,NORDP)
612   FORMAT(56X,I2,2X,1PE15.7)
      WRITE(6,613)
```

```
613   FORMAT(1H0,39X,'CHARACTERISTIC EQUATION IN ASCENDING ORDER--PLOCUS
     9 = 1'/)
      NA=NA-1
      WRITE(6,612)IZERO,ANS(1),(I,ANS(I+1),I=1,NA)
      WRITE(6,614)
614   FORMAT(1H0,43X,'OPEN LOOP ZEROES POLYNOMIAL IN ASCENDING ORDER'/)
      WRITE(6,612)IZERO,ZR(1),(I,ZR(I+1),I=1,NORDZ)
      NOUR=0
95    IF((ABS(PR(1)).GT.1.E-60).OR.(ABS(ZR(1)).GT.1.E-60))GO TO 107
      KCOEFD=NORDP+1
      DO 100 I=2,KCOEFD
      J=I-1
100   PR(J)=PR(I)
      KCOEFN=NORDZ+1
      DO 105 I=2,KCOEFN
      J=I-1
105   ZR(J)=ZR(I)
      NORDP=NORDP-1
      NORDZ=NORDZ-1
      NOUR=NOUR+1
      GO TO 95
107   IF(NOUR.NE.0)WRITE(6,607)NOUR
607   FORMAT(//,I5,1X,'FREE S''S EACH IN THE NUMERATOR AND IN THE DENOMI
     .NATOR WERE CANCELLED')
      CALL FLIPPC(PR,NORDP)
      CALL FLIPPC(ZR,NORDZ)
      IF(IABS(KGAIN).EQ.1)GO TO 120
      INC=1
      KCOEFN=1
      KCOEFD=1
      GO TO 125
120   INC=-1
      KCOEFN=NORDZ+1
      KCOEFD=NORDP+1
125   IF(ZR(KCOEFN).NE.0.0)GO TO 132
      KCOEFN=KCOEFN+INC
      GO TO 125
132   IF(PR(KCOEFD).NE.0)  GO TO 135
      KCOEFD=KCOEFD+INC
      GO TO 132
135   GNFCTR=ZR(KCOEFN)/PR(KCOEFD)
110   CALL RTLCSV
500   CONTINUE
      CALL CLASSB(INDCLS)
      STOP
200   WRITE(6,610)
610   FORMAT(//,' SOMETHING IS WRONG, THERE ARE NO OPEN LOOP ZEROES')
      GO TO 90
      END
```

```
      SUBROUTINE SKBJI
      COMMON COMCN(44CC)
      COMMON / 5 / TEMP(12,12,1),B(1616)
      COMMON /RPLBLK/DUMB(1C,5),ANSWER(11),ANS,NN(10),NA
      DIMENSION NORD(12,12),A(12,12,11),ANS(121),R2(21),R3(31),R4(41),
     .          R5(51),R6(61),R7(71),R8(81),R9(91),R10(101),R11(111),
     .          RESULT(121),MORD(12,12)
      EQUIVALENCE (A      ,COMON(2591))
      EQUIVALENCE (N      ,COMON( 508))
      EQUIVALENCE (MORD   ,CCMCN(    1))
      INTEGER D,E,F,G,H,P,Q,R,S,T,W,X,PG,QG,RG,SG
C
      NX=13-N
      DO 11 I=1,12
      DO 11 J=1,12
      NORD(I,J)=-1
      DO 11 K=1,11
   11 A(I,J,K) = 0
      DO 12 I=1,12
      NORD(I,I)=C
   12 A(I,I,1) = 1.
      II = 0
      DO 13 I=NX,12
      II=II+1
      JJ=0
      DO 13 J=NX,12
      JJ=JJ+1
      NORD(I,J)=MORD(II,JJ)
      DO 13 K=1,11
   13 A(I,J,K)=TEMP(II,JJ,K)
C   ******************************************************************
C   *  ZEROES THE ANSWER AND THE ORDER OF THE ANSWER.                *
C   ******************************************************************
C
      NA=0
      DO 60 I=1,121
   60 ANS(I)=C.
  700 DO 156 I=1,12
      KSIGN=0
      IF(NORD(1,I)+1)61,156,61
C
C   ******************************************************************
C   *  THE FOLLOWING FINDS THE PROPER TERMS OF THE DETERMINANT TO    *
C   *  MULTIPLY AND KEEPS A RECORD OF THE SIGN USING ISGN1,ISGN2,    *
C   *  ISGN3,ISGN4,ISGN5,ISGN6,ISGN7,ISGN8,ISGN9,ISGN10,ISGN11, AND  *
C   *  KSIGN.
C   ******************************************************************
C
   61 DO 155 J=1,12
      ISGN1=0
      IF(NORD(2,J)+1)62,155,62
   62 IF(J-I)63,155,64
```

```
63 ISGN1=ISGN1+1
64 D=NORD(1,I)+NORD(2,J)+1
   DO 65 E=1,D
65 R2(E)=0.
   NO1=NORD(1,I)+1
   NO2=NORD(2,J)+1
   DO 66 E=1,NO1
   DO 66 F=1,NO2
   IND=E+F-1
66 R2(IND)=A(1,I,E)*A(2,J,F)+R2(IND)
   DO 154 K=1,12
   IF(NORD(3,K)+1)68,154,68
68 ISGN2=0
   IF(K-I)69,154,70
69 ISGN2=ISGN2+1
70 IF(K-J)71,154,72
71 ISGN2=ISGN2+1
72 G=D+NORD(3,K)
   DO 73 E=1,G
73 R3(E)=0.
   NO3=NORD(3,K)+1
   NO4=NO1+NO2-1
   DO 74 E=1,NO4
   DO 74 F=1,NO3
   IND=E+F-1
74 R3(IND)=R2(E)*A(3,K,F)+R3(IND)
   DO 153 L=1,12
   IF(NORD(4,L)+1)76,153,76
76 ISGN3=0
   IF(L-I)77,153,78
77 ISGN3=ISGN3+1
78 IF(L-J)79,153,80
79 ISGN3=ISGN3+1
80 IF(L-K)81,153,82
81 ISGN3=ISGN3+1
82 H=G+NORD(4,L)
   DO 83 E=1,H
83 R4(E)=0.
   NO5=NORD(4,L)+1
   NO6=NO3+NO4-1
   DO 84 E=1,NO6
   DO 84 F=1,NO5
   IND=E+F-1
84 R4(IND)=R3(E)*A(4,L,F)+R4(IND)
   DO 152 M=1,12
   IF(NORD(5,M)+1)86,152,86
86 ISGN4=0
   IF(M-I)87,152,88
87 ISGN4=ISGN4+1
88 IF(M-J)89,152,90
89 ISGN4=ISGN4+1
90 IF(M-K)91,152,92
```

```
   91 ISGN4=ISGN4+1
   92 IF(M-L)93,152,94
   93 ISGN4=ISGN4+1
   94 S=F+NORD(5,M)
      DO 95 F=1,S
95    R5(F)=0.
      NO7=NORD(5,M)+1
      NO8=NO5+NO6-1
      DO 96 E=1,NO8
      DO 96 F=1,NO7
      IND=E+F-1
   96 R5(IND)=R4(E)*A(5,M,F)+R5(IND)
      DO 151 P=1,12
      IF(NORD(6,P)+1)98,151,98
   98 ISGN5=0
      IF(P-I)99,151,100
   99 ISGN5=ISGN5+1
  100 IF(P-J)101,151,102
  101 ISGN5=ISGN5+1
  102 IF(P-K)103,151,104
  103 ISGN5=ISGN5+1
  104 IF(P-L)105,151,106
  105 ISGN5=ISGN5+1
  106 IF(P-M)107,151,108
  107 ISGN5=ISGN5+1
  108 T=S+NORD(6,P)
      DO 109 F=1,T
109   R6(F)=0.
      NO9=NORD(6,P)+1
      NO10=NO7+NO8-1
      DO 110 E=1,NO10
      DO 110 F=1,NO9
      IND=E+F-1
  110 R6(IND)=R5(E)*A(6,P,F)+R6(IND)
      DO 150 Q=1,12
      IF(NORD(7,Q)+1)112,150,112
  112 ISGN6=0
      IF(Q-I)113,150,114
  113 ISGN6=ISGN6+1
  114 IF(Q-J)115,150,116
  115 ISGN6=ISGN6+1
  116 IF(Q-K)117,150,118
  117 ISGN6=ISGN6+1
  118 IF(Q-L)119,150,120
  119 ISGN6=ISGN6+1
  120 IF(Q-M)121,150,122
  121 ISGN6=ISGN6+1
  122 IF(Q-P)123,150,124
  123 ISGN6=ISGN6+1
  124 W=T+NORD(7,Q)
      DO 125 F=1,W
125   R7(F)=0.
```

```
      NO11=NORD(7,Q)+1
      NO12=NO9+NO10-1
      DO 126 E=1,NO12
      DO 126 F=1,NO11
      IND=E+F-1
  126 R7(IND)=R6(E)*A(7,Q,F)+R7(IND)
      DO 149 R=1,12
      IF(NORD(8,R)+1)128,149,128
  128 ISGN7=0
      IF(R-I)129,149,130
  129 ISGN7=ISGN7+1
  130 IF(R-J)131,149,132
  131 ISGN7=ISGN7+1
  132 IF(R-K)133,149,134
  133 ISGN7=ISGN7+1
  134 IF(R-L)135,149,136
  135 ISGN7=ISGN7+1
  136 IF(R-M)137,149,138
  137 ISGN7=ISGN7+1
  138 IF(R-P)139,149,140
  139 ISGN7=ISGN7+1
  140 IF(R-Q)141,149,142
  141 ISGN7=ISGN7+1
  142 X=W+NORD(8,R)
      DO 143 E=1,X
 143  R8(E)=0.
      NO13=NORD(8,R)+1
      NO14=NO11+NO12-1
      DO 144 E=1,NO14
      DO 144 F=1,NO13
      IND=E+F-1
  144 R8(IND)=R7(E)*A(8,R,F)+R8(IND)
      DO 850 PG=1,12
      IF(NORD(9,PG)+1)946,850,946
  946 ISGN8=0
      IF(PG-I)947,850,948
  947 ISGN8=ISGN8+1
  948 IF(PG-J)949,850,950
  949 ISGN8=ISGN8+1
  950 IF(PG-K)951,850,952
  951 ISGN8=ISGN8+1
  952 IF(PG-L)953,850,954
  953 ISGN8=ISGN8+1
  954 IF(PG-M)955,850,956
  955 ISGN8=ISGN8+1
  956 IF(PG-P)957,850,958
  957 ISGN8=ISGN8+1
  958 IF(PG-Q)959,850,960
  959 ISGN8=ISGN8+1
  960 IF(PG-R)961,850,962
  961 ISGN8=ISGN8+1
  962 IJX=X+NORD(9,PG)
```

```
      DO 963 E=1,IJX
963   R9(E)=0.
      NO15=NORD(9,PG)+1
      NO16=NO13+NO14-1
      DO 964 E=1,NO16
      DO 964 F=1,NO15
      IND=E+F-1
964   R9(IND)=R8(E)*A(9,PG,F)+R9(IND)
      DO 851 QG=1,12
      IF(NORD(10,QG)+1)966,851,966
966   ISGN9=0
      IF(QG-I)967,851,968
967   ISGN9=ISGN9+1
968   IF(QG-J)969,851,970
969   ISGN9=ISGN9+1
970   IF(QG-K)971,851,972
971   ISGN9=ISGN9+1
972   IF(QG-L)973,851,974
973   ISGN9=ISGN9+1
974   IF(QG-M)975,851,976
975   ISGN9=ISGN9+1
976   IF(QG-P)977,851,978
977   ISGN9=ISGN9+1
978   IF(QG-O)979,851,980
979   ISGN9=ISGN9+1
980   IF(QG-R)981,851,982
981   ISGN9=ISGN9+1
982   IF(QG-PG)983,851,984
983   ISGN9=ISGN9+1
984   IKX=IJX+NORD(10,QG)
      DO 985 E=1,IKX
985   R10(E)=0.
      NO17=NORD(10,QG)+1
      NO18=NO15+NO16-1
      DO 986 E=1,NO18
      DO 986 F=1,NO17
      IND=E+F-1
986   R10(IND)=R9(E)*A(10,QG,F)+R10(IND)
      DO 852 RG=1,12
      IF(NORD(11,RG)+1)988,852,988
988   ISGN10=0
      IF(RG-I)989,852,990
989   ISGN10=ISGN10+1
990   IF(RG-J)991,852,992
991   ISGN10=ISGN10+1
992   IF(RG-K)993,852,994
993   ISGN10=ISGN10+1
994   IF(RG-L)995,852,996
995   ISGN10=ISGN10+1
996   IF(RG-M)997,852,998
997   ISGN10=ISGN10+1
998   IF(RG-P)999,852,1000
```

```
 999 ISGN10=ISGN10+1
1000 IF(RG-Q)1001,852,1002
1001 ISGN10=ISGN10+1
1002 IF(RG-R)1003,852,1004
1003 ISGN10=ISGN10+1
1004 IF(RG-PG)1005,852,1006
1005 ISGN10=ISGN10+1
1006 IF(RG-QG)1007,852,1008
1007 ISGN10=ISGN10+1
1008 ILX=IKX+NORD(11,RG)
     DO 1009 E=1,ILX
1009 R11(E)=0.
     NO19=NORD(11,RG)+1
     NO20=NO17+NO18-1
     DO 1010 E=1,NO20
     DO 1010 F=1,NO19
     IND=E+F-1
1010 R11(IND)=R10(E)*A(11,RG,F)+R11(IND)
     DO 853 SG=1,12
     IF(NORD(12,SG)+1)1112,853,1112
1112 ISGN11=0
     IF(SG-I)1113,853,1114
1113 ISGN11=ISGN11+1
1114 IF(SG-J)1115,853,1116
1115 ISGN11=ISGN11+1
1116 IF(SG-K)1117,853,1118
1117 ISGN11=ISGN11+1
1118 IF(SG-L)1119,853,1120
1119 ISGN11=ISGN11+1
1120 IF(SG-M)1121,853,1122
1121 ISGN11=ISGN11+1
1122 IF(SG-P)1123,853,1124
1123 ISGN11=ISGN11+1
1124 IF(SG-Q)1125,853,1126
1125 ISGN11=ISGN11+1
1126 IF(SG-R)1127,853,1128
1127 ISGN11=ISGN11+1
1128 IF(SG-PG)1129,853,1130
1129 ISGN11=ISGN11+1
1130 IF(SG-QG)1131,853,1132
1131 ISGN11=ISGN11+1
1132 IF(SG-RG)1133,853,1134
1133 ISGN11=ISGN11+1
1134 IMX=ILX+NORD(12,SG)
     DO 1135 E=1,IMX
1135 RESULT(E) = 0
     NO21=NORD(12,SG)+1
     NO22=NO19+NO20-1
     DO 1136 E=1,NO22
     DO 1136 F=1,NO21
     IND=E+F-1
1136 RESULT(IND)=R11(E)*A(12,SG,F)+RESULT(IND)
```

```
      NR=IMX-1
C
C   ********************************************************************
C   *  CALCULATES THE SIGN OF THE TERM USING MOD 2.                    *
C   ********************************************************************
C
      KSIGN=ISGN1+ISGN2+ISGN3+ISGN4+ISGN5+ISGN6+ISGN7+ISGN8+ISGN9+ISGN10
     1+ISGN11
      IF(KSIGN-(KSIGN/2)*2)145,147,145
  145 DO 146 IM=1,IMX
  146 RESULT(IM)=-RESULT(IM)
  147 DO 148 IM=1,IMX
  148 ANS(IM)=ANS(IM)+RESULT(IM)
      NA=MAXO(NA,NR)
  853 CONTINUE
  852 CONTINUE
  851 CONTINUE
  850 CONTINUE
  149 CONTINUE
  150 CONTINUE
  151 CONTINUE
  152 CONTINUE
  153 CONTINUE
  154 CONTINUE
  155 CONTINUE
  156 CONTINUE
 402  NR=NA+1
      ZANS=ANS(NR)
      IF(ABS(ZANS).GT.1.E-60)GO TO 400
      NA=NA-1
      IF(NA.LT.0)GO TO 500
      GO TO 402
 400  RETURN
 500  WRITE(6,501)
 501  FORMAT(//,' SOMETHING IS WRONG, KBJI CALCULATED A ZEROTH POLYNOMIA
     .L')
      GO TO 400
      END
```

```
      SUBROUTINE PSUB (Z,IDIMZ,X,IDIMX,Y,IDIMY)
C
C     THIS IBM-SSP SUBROUTINE SUBTRACTS ONE POLYNOMIAL FROM ANCTHER
C
      DIMENSION Z(1),X(1),Y(1)
      NDIM = IDIMX
      IF(IDIMX.GE.IDIMY)GO TO 20
      NDIM=IDIMY
 20   IF(NDIM.LE.0)GO TO 90
      DO 80 I=1,NDIM
      IF(I.GT.IDIMX)GO TO 6C
      IF(I.GT.IDIMY)GO TO 7C
      Z(I)=X(I)-Y(I)
      GO TO 8C
   60 Z(I)=-Y(I)
      GO TO 80
   70 Z(I) = X(I)
   80 CONTINUE
   90 ICIMZ=NDIM
      RETURN
      END
```

```
      SUBROUTINE FLIPPO (A,N)
      DIMENSION A(1)
      NUM=N+1
      NUM2=NUM/2
      DO 10 I=1,NUM2
      J=NUM+1 - I
      DUMMY = A(I)
      A(I)=A(J)
10    A(J)=DUMMY
      RETURN
      END
```

```
      SUBROUTINE RLPLOT (MODE,ISWX,X,Y,NSYM,NCHAN,SCALE,NSTORE,
     1                   STORXY,NPX,NOVFLW,NRUN,MARKT,NXINT,NHO,MSYMX,
     2                   MGRID,MSCALE,TITLE)
      COMMON COMCN(4400)
      DIMENSION IMT(2,3),SCALE(4),TITLE(10),YSCALE(13),MARK(132),
     1          KRUN(10),GRAPH(131,2),SYMBCL(32),NXINTP(2),
     2          FACTOR(2,2),SHIFT(2,2)
      EQUIVALENCE (INDCLS,CCMCN( 725))
      INTEGER STORXY(2,1500)
      DATA SYMBOL/1HA,1HB,1HC,1HD,1HE,1HF,1HG,1HH,1HI,1HJ,1HK,1HL,1HM,
     1            1HN,1HP,1HR,1HS,1HT,1HU,1HV,1H1,1H2,1H3,1H4,1H5,1H6,
     2           1H7,1H8,1H9,1HC,1HX,1HO/,BLANK,DASH,PLUS,CNE/1H ,1H-,
     3            1H+,1H1/
    1 FORMAT (1H ,T53,*ROOT LOCLS PLOT - RUN NUMBER *,I2/)
    2 FORMAT (1H ,20X,10A6)
    3 FORMAT (///56X,*PLOT IDENTIFICATICN*//
     1        48X*RUN NUMBER  PLOT SYMBOL  GAIN SYMBOL*/)
    4 FORMAT (1H ,48X,*ROOT LOCUS PLOT - CCMPARISON OF*,I3,*RUNS*/)
    5 FORMAT (//65X*IMAGINARY*)
    6 FORMAT (3X,12(1X,F9.3),F9.3)
    7 FORMAT (3X*REAL*)
    8 FORMAT (1X,131A1)
    9 FORMAT (1H ,F7.3,1X,123A1)
   10 FORMAT (////* END OF PLCT*/1H1)
   11 FORMAT (51X,I3,10X,A1,12X,A1)
   13 FORMAT (////* NOTE - PLCTTER STORAGE CAPACITY WAS EXCEEDED AND*I5,
     1        * DATA POINTS DISCARDED*)
   14 FORMAT (//54X*IMAGINARY - MLLTIPLY BY*1PE7.0)
   15 FORMAT (3X*REAL - MULTIPLY BY*1PE7.0)
      IF(ISWX .LT. 1 .OR. ISWX .GT. 4) GO TO 17
   16 GO TO (28,32,34,38),ISWX
C     INITIAL COMPUTATIONS
   17 IOW = 6
      DO 20 J=1,131
      GRAPH(J,1) = BLANK
   20 GRAPH(J,2) = BLANK
   28 NPX = NXINT + 2
      NOVFLW = 0
      NXINTP(NHO) = NXINT + 1
      FACTOR(1,NHO) = NXINT/(SCALE(2)-SCALE(1))
      FACTOR(2,NHO) = 100./(SCALE(4)-SCALE(3))
      SHIFT(1,NHO) = -SCALE(1)*FACTOR(1,NHO) + 1.5
      SHIFT(2,NHO) = -SCALE(3)*FACTOR(2,NHO) + 19.5
      DO 29 J = 1,NPX
   29 STORXY(1,J) = 0
      IF(NHO-2) 34,30,34
   30 NCRUN = 0
   32 NCRUN = NCRUN + 1
      KRUN(NCRUN) = NRLN
   34 DO 37 J=1,3
      IF(MARKT-1) 35,36,36
   35 IMT(NHO,J) = 1
```

```
      GO TO 37
36    IMT(NHO,J) = MARKT
37    CONTINUE
      ISWX = 4
38    GO TO (40,100),MODE
C     X-Y PLOT INFORMATION PROCESSING AND STORAGE ROUTINE
40    KTSW = 0
      IF(IMT(NHO,NCHAN)-MARKT) 48,46,50
46    KTSW = 1
      IMT(NHO,NCHAN) = 1
      GO TO 50
48    IMT(NHO,NCHAN) = IMT(NHO,NCHAN) + 1
50    IF(NPX-NSTORE) 51,51,511
511   NOVFLW = NOVFLW + 1
      RETURN
51    MARKX = X*FACTOR(1,NHO) + SHIFT(1,NHO)
      IF(MARKX) 90,90,52
52    IF(MARKX-NXINTP(NHO))54,54,90
54    MARKY = Y*FACTOR(2,NHO) + SHIFT(2,NHO)
      IF(MARKY) 90,90,56
56    IF(MARKY-131) 58,58,90
58    IF(KTSW) 62,62,60
60    NSYMB = NSYM + 10
      GO TO 64
62    NSYMB = NSYM
64    N1 = MARKX
      IF(STORXY(1,N1)) 73,73,66
66    N2 = STORXY(1,N1)/1000000
      N3 = STORXY(1,N1)/100
      IF(MARKY-N2) 68,74,68
68    NEXT = N3 - N2*10000
      IF(NEXT) 72,72,70
70    N1 = NEXT
      GO TO 66
72    STORXY(1,N1) = STORXY(1,N1) + NPX*100
      N1 = NPX
      NPX = NPX + 1
73    STORXY(1,N1) = MARKY*1000000 + NSYMB
      RETURN
74    ISYM  = STORXY(1,N1) - N3*100
      IF(MSYMX) 75,75,76
75    IF(ISYM-30) 82,82,90
76    DO 78 J=21,29
      IF(ISYM-J) 80,80,78
78    CONTINUE
      GO TO 90
80    NSYMB = J+1
82    STORXY(1,N1) = STORXY(1,N1) - ISYM + NSYMB
90    RETURN
C     PLOT ROUTINE
100   IF(SHIFT(2,NHO)-9) 190,160,160
160   IF(SHIFT(2,NHO)-129) 190,170,180
```

```
170   MARKX = SHIFT(2,NHO)
      MARK(1) = MARKX
      GO TO 200
180   MARKX = C
      MARK(1) = 1
200   NYAXIS = SHIFT(1,NHO)
      IF(NYAXIS-NXINTP(NHO)) 2C2,2C9,2C8
202   IF(NYAXIS-1) 2C4,2C9,2C9
204   NYAXIS = 1
      GO TO 21C
208   NYAXIS = NXINTP(NHO)
209   IF(MSCALE) 210,21C,22C
210   I1 = 1
      GO TO 26C
220   I1 = MSCALE-NYAXIS+2+((NYAXIS-2)/MSCALE)*MSCALE
222   IF(MGRID) 224,224,226
224   I2 = 1
      GO TO 260
226   I2 = MGRID-(NYAXIS-2)/MSCALE+(((NYAXIS-2)/MSCALE)/MGRID)*MGRID
260   KSW = 2
      CALL CLASS(INDCLS)
300   GO TO (3C2,3C4),NHO
302   WRITE(IOW,1) NRUN
      WRITE(IOW,2) TITLE
      GO TO 316
304   WRITE(IOW,4) NCRUN
      WRITE (IOW,2) TITLE
      WRITE (IOW,3)
      WRITE(IOW,11) (KRUN(I),SYMBOL(I),SYMBCL(I+10),I=1,NCRUN)
315   J = 3
      XMULT = 1CCC.
320   SF = 1.
      IF(ABS(SCALE(J+1))-ABS(SCALE(J))) 325,330,330
325   BASE = ABS(SCALE(J))
      GO TO 335
330   BASE = ABS(SCALE(J+1))
335   IF(BASE-XMULT) 350,34C,34C
340   SF = SF*1C.
      BASE = BASE/1C.
      GO TO 335
350   IF(BASE-1.) 355,36C,36C
355   SF = SF/10.
      BASE = BASE*1C.
      GO TO 350
360   IF(J-1) 4CC,40C,365
365   IF(SF-1.) 37C,375,37C
370   WRITE(IOW,14) SF
      GO TO 380
375   WRITE(IOW,5)
380   YSCALE(2) = SCALE(3)/SF
      BASE = (SCALE(4)/SF-YSCALE(2))/10.
      YSCALE(1) = YSCALE(2) - BASE
```

```
      DO 385 J=2,12
385   YSCALE(J+1) = YSCALE(J) + BASE
      WRITE(IOW,6) YSCALE
      J = 1
      XMULT = 100.
      GO TO 320
400   IF(SF-1.) 405,410,405
405   WRITE(IOW,15) SF
      GO TO 415
410   WRITE(IOW,7)
415   BASE = SCALE(1)/SF
      XMULT = (SCALE(2)/SF-BASE)/NXINT
500   NP = 1
      JSW = 1
      GO TO 520
505   JSW = 0
      KSW = 1
      IF(I1-MSCALE) 508,510,520
508   I1 = I1 + 1
      GO TO 520
510   JSW = 1
      I1 = 1
      IF(I2-MGRID) 512,515,520
512   I2 =I2 +1
      GO TO 520
515   KSW = 2
      I2 = 1
520   IF(NP-NYAXIS) 525,530,525
525   IF(JSW) 700,700,540
530   DO 532 J=1,131
532   GRAPH(J,2) =DASH
      KSW = 2
535   JSW = 1
540   XVAR = (NP-1)*XMULT + BASE
700   IF(MARKX) 705,705,702
702   GRAPH(MARKX,KSW) = ONE
705   IF(KSW-2) 708,706,708
706   DO 707 J=5,125,10
707   GRAPH(J,2) = PLUS
708   NMK = 1
710   IF(STORXY(1,NP)) 800,800,715
715   N1 = NP
720   MARKY = STORXY(1,N1)/1000000
      N2 = STORXY(1,N1)/100
      NEXT = N2 - MARKY*10000
      NSYMB = STORXY(1,N1) - N2*100
      GRAPH(MARKY,KSW) = SYMBOL(NSYMB)
      NMK = NMK+1
      MARK(NMK)= MARKY
      IF(NEXT) 800,800,730
730   N1 = NEXT
      GO TO 720
```

```
800   IF(JSW) 810,810,820
810   WRITE(IOW,8) (GRAPH(J,1),J=1,131)
      GO TO 835
820   WRITE(IOW,9) XVAR,(GRAPH(J,KSW),J=9,131)
830   IF(NP-NYAXIS) 835,850,835
835   DO 840 J=1,NMK
      J1 = MARK(J)
840   GRAPH(J1,KSW) = BLANK
      GO TO 900
850   DO 860 J=1,131
860   GRAPH(J,2) = BLANK
900   NP = NP + 1
      IF(NP-NXINTP(NHO)) 505,902,908
902   JSW = 1
      IF(MGRID) 520,520,904
904   KSW = 2
      GO TO 520
908   IF(NOVFLW) 910,910,905
905   WRITE(IOW,13) NOVFLW
910   WRITE(IOW,10)
999   RETURN
      END
```

```
      SUBROUTINE RTLOC(NF,NP,X,Y,ANG,STEP,TOL1,CHG,NAI,OUT,ISW,NS,XD,YD,
     1                ANGD,CLTD)
      COMMON /RTLOCS/
     1       PI1D,PI2D,TWOPID,PI1,PI2,TWOPI,IOW,NIA,NID,NSTEP,MODE,
     2       SGNLT(3),DELT2,TOLLH,NPL,NZL,KGAIN,TAU,DELT,TCL,BNDY(4),
     3       PR(100),PI(100),XMP(100),ZR(100),ZI(100),XMZ(100),MOUT,
     4       DUM1(3),NPREC,DUMB2(91),IT(4),DUM3(2),TCLANG,DUM4,GNFCTR,
     8       KOSTOR(2,1501)
 1    FORMAT (1XA2,2X1P7E14.6,4I5)
      GO TO (50,50,800,50,50,900,905,1000,1000,58,58),NF
 50   NRT= 0
      STEP2 = STEP*STEP
      IF(NP) 405,55,400
C     SINGLE PRECISION ROUTINE LINKAGES
 55   IF(NF-2) 65,65,60
 58   NRT = -1
      STEP2 = STEP*STEP
 60   TPX = X
      TPY = Y
      IF(NF-5) 200,300,62
 62   IF(NF-10) 226,226,200
 65   IF(TOL1-.99E-4) 400,100,100
C     ROOT LOCUS ADVANCE - SINGLE PRECISION
 100  ANGT = ANG
      NIA = 1
      NID = 0
 105  DX = STEP*COS(ANGT)
      DY = STEP*SIN(ANGT)
      TPX = X + DX
      TPY = Y + DY
      IF(ABS(TPY)-TOL1) 110,110,115
 110  TPY = 0.
 115  NPOINT = 1
      GO TO 300
 120  IF(ABS(SANG)-TOL1) 185,185,125
 125  IF(NIA-NAI) 130,128,128
 128  ISW = 5
      RETURN
 130  NIA = NIA + 1
C     ANGLE CORRECTION ROUTINE (NEWTON-RAPHSON)
 132  NRT = 1
      GO TO 200
 175  IF(IT(3)) 105,105,180
 180  WRITE(IOW,1) SGNLT(3),TPX,TPY,A,SANG,ANGT,TANG,STEP,NIA,NID,NF
      GO TO 105
 185  XS = TPX
      YS = TPY
      NRT = 0
C     TRIAL ANGLE COMPUTATION AND ANGLE CORRECTION - SINGLE PRECISION
 200  FX = 0.
      FY = -TAL
      NS = 0
```

```
      IF(NPL) 224,224,204
204   DO 220 I=1,NPL
      XT = TPX-PR(I)
      YT = TPY-PI(I)
      DIST = XT*XT + YT*YT
      IF(NRT) 208,206,206
206   IF(MODE) 210,210,208
208   IF(DIST-STEP2) 212,212,216
210   IF(DIST) 673,673,216
212   ISW = 3
      NS = I
      IF(DIST) 299,299,216
216   FX = FX + YT*XMP(I)/DIST
      FY = FY - XT*XMP(I)/DIST
220   CONTINUE
224   IF(NF-10) 226,247,247
226   IF(NZL) 246,246,228
228   DO 244 I=1,NZL
      XT = TPX-ZR(I)
      YT = TPY-ZI(I)
      DIST = XT*XT + YT*YT
      IF(NRT) 232,230,230
230   IF(MODE) 232,232,234
232   IF(DIST-STEP2) 236,236,240
234   IF(DIST) 673,673,240
236   ISW = 2
      NS = I
      IF(DIST) 299,299,240
240   FX = FX - YT*XMZ(I)/DIST
      FY = FY + XT*XMZ(I)/DIST
244   CONTINUE
246   IF(NRT) 2461,248,274
2461  IF(NF-10) 2462,247,247
2462  IF(NS) 1030,1030,247
247   ISW = 1
      RETURN
248   IF(MODE) 252,252,250
250   CALL ARCTAN(1,FY,-FX,TANG,D,D,D)
      GO TO 253
252   CALL ARCTAN(1,-FY,FX,TANG,D,D,D)
253   IF(NS) 255,255,254
254   IF(NF-4) 2541,272,272
2541  IF(NPREC) 256,256,299
255   IF(ABS(ABS(TANG)-PI2)-TOL) 256,256,258
256   TANG = SIGN(PI2,TANG)
258   IF( IT(1)) 262,262,260
260   WRITE(IOW,1) SGNLT(1),TPX,TPY,ANG,TANG,FX,FY,STEP,NIA,NID,NSTEP,NF
262   A = ABS(TANG-ANG)
      IF(A-CHG) 266,266,264
264   IF(ABS(A-TWOPI)-CHG) 266,266,270
266   IF(ABS(FX)+ABS(FY)-1.E-5) 270,268,268
268   ISW = 1
```

```
      IF(NF-2) 269,269,272
 269  A = ABS(TANG-ANGT)
      IF(A-CHG) 2692,2692,2691
 2691 IF(ABS(A-TWOPI)-CHG) 2692,2692,270
 2692 IF(NF-2) 286,296,296
 270  ISW = 4
 272  OUT = TANG
      RETURN
 274  IF(NS) 275,275,2742
 2742 IF(NPREC) 275,275,299
 275  A = FY*DX-FX*DY
      IF(A) 276,270,276
 276  TANG = -SANG/A
      IF(ABS(TANG)-CHG) 278,278,277
 277  TANG = SIGN(.2/NIA,TANG)
 278  ANGT = ANGT + TANG
      GO TO 175
C     CHECK THAT A ZERO OF 'SANG' IS WITHIN 'TOL' PERPENDICULAR DISTANCE
 286  DX1 = TOLLF*DX
      DY1 = TOLLF*DY
      TPX = XS + DY1
      TPY = YS - DX1
      NPOINT = 2
      GO TO 300
 288  TPX = XS -DY1
      TPY = YS + DX1
      NPOINT = 3
      GO TO 300
 290  XD = X
      YD = Y
      ANGD = ANG
      ANGTD = ANGT
      TOL2 = TOLANG
      NID = 1
      GO TO 505
 296  X = XS
      Y = YS
      ANG = TANG
      OUT = SANG
 299  RETURN
C     ANGLE SUMMATION ROUTINE (EVALUATE 'SANG') - SINGLE PRECISION
 300  IF(TAU) 302,304,302
 302  A = -TAU*TPY
      A = A - IFIX(A/TWOPI)*TWOPI
      GO TO 305
 304  A = 0.
 305  IF(NPL) 315,315,306
 306  DO 310 I=1,NPL
      CALL ARCTAN(1,TPX-PR(I),TPY-PI(I),ANGLE,D,D,D)
 310  A = A -ANGLE*XMP(I)
 315  IF(NZL) 330,330,320
 320  DO 325 I=1,NZL
```

```
      CALL ARCTAN(1,TPX-ZR(I),TPY-ZI(I),ANGLE,D,D,D)
 325  A = A + ANGLE*XMZ(I)
 330  IF(KGAIN) 340,335,335
 335  A = A + PII
 340  IF(A+PII) 345,350,350
 345  A = A + TWOPI
      GO TO 340
 350  IF(A-PII) 360,360,355
 355  A = A-TWOPI
      GO TO 350
 360  OUT = A
      IF(NF-5) 365,299,1020
 365  IF(NPOINT-1) 370,370,375
 370  SANG = A
      GO TO 120
 375  IF(A*SANG) 296,296,380
 380  IF(NPOINT-2) 288,288,290
C     DOUBLE PRECISION ROUTINE LINKAGES
 400  XD = X
      YD = Y
      ANGD = ANG
 405  IF(NF-2) 500,500,410
 410  TPXD = XD
      TPYD = YD
      IF(NF-4) 600,600,700
C     ROOT LOCUS ADVANCE - DOUBLE PRECISION
 500  ANGTD= ANGD
      TOL2 = TOL1
      NIA = 1
      NID = 0
 505   DXD = STEP*COS(ANGTD)
      DYD  = STEP*SIN(ANGTD)
      TPXD = XD + DXD
      TPYD = YD + DYD
      IF(ABS(TPYD)-TOL1) 510,510,515
 510  TPYD = 0.
 515  NPOINT = 1
      GO TO 700
 520   IF(ABS(SANGD)-TOL2) 585,585,525
 525  IF(NIA-NAI) 530,528,528
 528  ISW = 5
      RETURN
 530  NIA = NIA+1
C     ANGLE CORRECTION ROUTINE (NEWTON-RAPHSON)
 532  NRT = 1
      GO TO 600
 575  IF( IT(3)) 505,505,580
 580  WRITE(IOW,1) SGNLT(3),TPXD,TPYD,AD,SANGD,ANGTD,TANGD,STEP,NIA,NID,
     1            NF
      GO TO 505
 585  XSD = TPXD
      YSD = TPYD
```

```
      NRT = 0
C     TRIAL ANGLE COMPUTATION AND ANGLE CORRECTION - DOUBLE PRECISION
 600  FXD = 0.
      FYD = -TAU
      NS = 0
      IF(NPL) 624,624,604
 604  DO 620 I=1,NPL
      XTD = TPXD-PR(I)
      YTD = TPYD-PI(I)
      DISTD = XTD*XTD + YTD*YTD
      IF(NRT) 608,606,606
 606  IF(MODE) 610,610,608
 608  IF(DISTD-STEP2) 612,612,616
 610  IF(DISTD) 673,673,616
 612  ISW = 3
      NS = I
      IF(DISTD) 699,699,616
 616  FXD = FXD +YTD*XMP(I)/DISTD
 620  FYD = FYD - XTD*XMP(I)/DISTD
 624  IF(NZL) 646,646,628
 628  DO 644 I=1,NZL
      XTD = TPXD - ZR(I)
      YTD = TPYD - ZI(I)
      DISTD = XTD*XTD + YTD*YTD
      IF(NRT) 632,630,630
 630  IF(MODE) 632,632,634
 632  IF(DISTD-STEP2) 636,636,640
 634  IF(DISTD) 673,673,640
 636  ISW = 2
      NS = I
      IF(DISTD) 699,699,640
 640  FXD = FXD - YTD*XMZ(I)/DISTD
 644  FYD = FYD + XTD*XMZ(I)/DISTD
 646  IF(NRT) 647,648,674
 647  IF(NS) 1060,1060,247
 648  IF(MODE) 652,652,650
 650  CALL ARCTAN(2,E,E,E,FYD,-FXD,TANGD)
      GO TO 653
 652  CALL ARCTAN(2,E,E,E,-FYD,FXD,TANGD)
 653  IF(NS) 655,655,654
 654  IF(NF-4) 6541,672,672
 6541 IF(NPREC) 656,656,655
 655  IF(ABS(ABS(TANGD)-PI2D)-TCL) 656,656,658
 656  TANGD = SIGN(PI2D,TANGD)
 658  IF(IT(1)) 662,662,660
 660  WRITE(IOW,1) SGNLT(1),TPXD,TPYD,ANGD,TANGD,FXD,FYD,STEP,NIA,NID,
     1            NSTEP,NF
 662  AD=ABS(TANGD-ANGD)
      IF(AD-CHG) 666,666,664
 664  IF(ABS(AD-TWOPID)-CHG) 666,666,670
 666  IF(ABS(FXD)+ABS(FYD)-1.E-14) 670,668,668
 668  ISW = 1
```

```
      IF(NF-2) 669,669,672
 669  AD = ABS(TANGD-ANGTD)
      IF(AD-CHG) 6692,6692,6691
 6691 IF(ABS(AD-TWOPID)-CHG) 6692,6692,670
 6692 IF(NF-2) 686,696,696
 670  ISW = 4
 672  OUT = TANGC
      OUTD = TANGD
      RETURN
 673  ISW =4
      RETURN
 674  IF(NS) 675,675,6742
 6742 IF(NPREC) 675,675,699
 675  AD = FYD*DXD-FXD*DYD
      IF(AD) 676,670,676
 676  TANGD = -SANGD/AD
      IF(ABS(TANGD)-CHG) 678,678,677
 677  TANGD = SIGN(.2/NIA,TANGD)
 678  ANGTD = ANGTD + TANGD
      GO TO 575
C     CHECK THAT A ZERO OF'SANGD' IS WITHIN'TOL' PERPENDICULAR DISTANCE
 686  DX1D = TOLLH*DXD
      DY1D = TOLLH*DYD
      TPXD = XSD + DY1D
      TPYD = YSD - DX1D
      NPOINT = 2
      GO TO 700
 688  TPXD = XSD - DY1D
      TPYD = YSD + DX1D
      NPOINT = 3
      GO TO 700
 690  IF(NID-3) 692,696,696
 692  NID = NID + 1
      TPXD = XSD
      TPYD = YSD
      TOL2 = TOLANG
      GO TO 532
 696  IF(NP) 698,697,697
 697  X = XSD
      Y = YSD
      ANG = TANGD
      OUT = SANGD
 698  XD = XSD
      YD = YSD
      ANGD = TANGD
      OUTD = SANGD
 699  RETURN
C     ANGLE SUMMATION ROUTINE (EVALUATE 'SANGD') - DOUBLE PRECISION
 700  IF(TAU) 702,704,702
 702  AD = -TAU*TPYD
      AD = AD-IFIX(SNGL(AD/TWOPID))*TWOPID
      GO TO 705
```

```
704   AD = 0.
      IF(NPL) 715,715,705
705   DO 710 I=1,NPL
      CALL ARCTAN(2,E,E,E,TPXD-PR(I),TPYD-PI(I),ANGLED)
710   AD = AD - ANGLED*XMP(I)
715   IF(NZL) 730,730,720
720   DO 725 I=1,NZL
      CALL ARCTAN(2,E,E,E,TPXD-ZR(I),TPYD-ZI(I),ANGLED)
725   AD = AD + ANGLED*XMZ(I)
730   IF(KGAIN) 740,735,735
735   AD = AD + PI1D
740   IF(AD+ PI1D) 745,750,750
745   AD = AD + TWOPID
      GO TO 740
750   IF(AD-PI1D) 760,760,755
755   AD = AD - TWOPID
      GO TO 750
760   OUT = AD
      OUTD = AD
      IF(NF-5) 765,699,1050
765   IF(NPOINT-1) 770,770,775
770   SANGD = AD
      GO TO 520
775   IF(AD*SANGD) 696,696,780
780   IF(NPOINT-2) 688,688,690
C     BREAKAWAY POINT LOCATION ROUTINE (FINDS ZEROES OF (FX,FY))
800   TPXD = X
      TPYD = Y
      DYD = 0.
      YSD = 1.E15
      NRT = 0
      J = 1
802   FXD = 0.
      FYD = -TAU
      FXX = 0.
      FXY = 0.
      IF(NPL) 820,820,805
805   DO 815 I=1,NPL
      XTD = TPXD-PR(I)
      YTD = TPYD-PI(I)
      DX1D = XTD*XTD
      DY1D = YTD*YTD
      DISTD = DX1D + DY1D
      DIST2 = DISTD*DISTD
      IF(DISTD-1.E-25) 890,890,808
808   FYD = FYD - XTD*XMP(I)/DISTD
      FXY = FXY +(DX1D-DY1D)*XMP(I)/DIST2
      IF(Y) 810,815,810
810   FXD = FXD + YTD*XMP(I)/DISTD
      FXX = FXX - XTD*YTD*XMP(I)/DIST2
815   CONTINUE
820   IF(NZL) 840,840,825
```

```
 825   DO 835 I=1,NZL
       XTD = TPXD - ZR(I)
       YTD = TPYD - ZI(I)
       DX1D = XTD*XTD
       DY1D = YTD*YTD
       DISTD = DX1D + DY1D
       DIST2 = DISTD*DISTD
       IF(DISTD-1.E-25) 890,890,828
 828   FYD = FYD + XTD*XMZ(I)/DISTD
       FXY = FXY - (DX1D-DY1D)*XMZ(I)/DIST2
       IF(Y) 830,835,830
 830   FXD = FXD - YTD*XMZ(I)/DISTD
       FXX = FXX + XTD*YTD*XMZ(I)/DIST2
 835   CONTINUE
 840   IF(Y) 845,852,845
 845   AD = 4.*FXX*FXX + FXY*FXY
       IF(AD) 850,885,850
 850   DXD = (-2.*FXD*FXX-FYD*FXY)/AD
       DYD = (2.*FYD*FXX-FXD*FXY)/AD
       XSD = ABS(DXD) + ABS(DYD)
       GO TO 862
 852   IF(FXY) 855,885,855
 855   DXD = -FYD/FXY
       XSD = ABS(DXD)
 862   IF(IT(2)) 870,870,865
 865   WRITE(IOW,1) SGNLT(2),TPXD,TPYD,DXD,DYD,FXD,FYD,STEP,J
 870   IF(ABS(FXD)+ABS(FYD)-TOL1) 871,871,8721
 871   IF(XSD-YSD-YSD) 8711,885,885
 8711  IF(XSD-YSD) 872,8712,8712
 8712  IF(NRT) 8713,8713,885
 8713  NRT = 1
 872   YSD = XSD
 8721  TPXD = TPXD + DXD
       TPYD = TPYD + DYD
       IF(ABS(TYPD)-TOL1) 873,873,874
 873   TPYD = 0.
 874   IF(ABS(TPXD)+ABS(TYPD)-1.E15) 875,890,890
 875   IF(J-NAI) 880,885,885
 880   J = J + 1
       GO TO 802
 885   IF((TPXD-X)*(TPXD-X)+(TPYD-Y)*(TPYD-Y)-CHG*CHG) 888,890,890
 888   XD = TPXD
       YD = TPYD
       ISW = 0
       RETURN
 890   ISW = 1
       RETURN
C      GAIN COMPUTATION ROUTINE
 900   GAIN = 1.
       NS = 0
       GO TO 910
 905   GAIN = BODE
```

```
      NS = NBODE
  910 IF(TAU) 912,920,912
  912 A = -TAU*X
  914 IF(ABS(A)-100.) 918,918,916
  916 E = SIGN(100.,A)
      DIST = EXP(E)
      NTR = -1
      N = 1
      A = A-E
      GO TO 942
  918 GAIN = EXP(A)*GAIN
  920 NTR = 0
  922 I = 1
  924 IF(NTR) 914,930,926
  926 IF(I-NPL) 928,928,960
  928 N = XMP(I)
      DIST = (X-PR(I))*(X-PR(I))+(Y-PI(I))*(Y-PI(I))
      GO TO 936
  930 IF(I-NZL) 934,934,932
  932 NTR = 1
      NS = -NS
C  THE VARIABLE NAI IS USED TO DETECT IF THIS IS A RUN TO FIND THE
C  NORMALIZING GAIN CONSTANT. IF SO, NAI=30,AND THE PLANT GAIN IS NOT
C  DIVIDED OUT OF THE GAIN.
      GAIN=1./GAIN
      IF(NAI.EQ.30)GO TO 922
      GAIN=GAIN/(GNFCTR*GNFCTR)
      GO TO 922
  934 N = XMZ(I)
      DIST = (X-ZR(I))*(X-ZR(I))+(Y-ZI(I))*(Y-ZI(I))
  936 IF(DIST) 938,938,942
  938 IF(NP) 940,956,940
  940 IF(NTR) 958,958,996
  942 DO 952 J=1,N
  944 IF(GAIN-1.E26) 948,948,946
  946 GAIN = GAIN/1.E10
      NS = NS +5
      GO TO 944
  948 IF(GAIN-1.E-26) 950,952,952
  950 GAIN = GAIN*1.E10
      NS = NS - 5
      IF(GAIN) 952,952,948
  952 GAIN = GAIN*DIST
  956 I = I+1
      GO TO 924
  960 IF(NP) 962,983,962
  962 IF(ISW) 984,964,984
  964 GAIN = SQRT(GAIN)
  966 IF(GAIN-1.E6) 970,968,968
  968 GAIN = GAIN/10.
      NS = NS + 1
      GO TO 966
```

```
 970  IF(GAIN-1.E-3) 972,974,974
 972  GAIN = GAIN*10.
      NS = NS-1
      GO TO 970
 974  IF(NS) 980,990,976
 976  IF(GAIN-1.E5) 978,990,990
 978  GAIN = GAIN*10.
      NS= NS-1
      GO TO 974
 980  IF(GAIN-1.E-2) 989,982,982
 982  GAIN = GAIN/10.
      NS = NS + 1
      GO TO 974
 983  BODE = GAIN
      NBODE = NS
 984  IF(GAIN-100.) 986,985,985
 985  GAIN = GAIN/100.
      NS = NS + 1
      GO TO 984
 986  IF(GAIN-1.) 987,988,988
 987  GAIN = GAIN*100.
      NS = NS-1
      GO TO 986
 988  OUT = SQRT(GAIN)
      RETURN
 989  GAIN = GAIN*1000.
      NS = NS - 3
 990  OUT = GAIN
      RETURN
 992  IF(NP) 998,994,998
 994  BODE = 1.
      OUT = 1.
      GO TO 997
 996  OUT = 0.
 997  NS = 0.
      RETURN
 998  OUT = 1.000000001D6
      NS = 999
      RETURN
C     NEWTON-RAPHSON ROUTINE TO FIND ZEROES OF'SANG' WITH FIXED X OR Y
 1000 NIA = 0
      NRT = -1
      STEP2 = STEP*STEP
      NTR = NF-8
      TANG = 1.5*CHG
      IF(TOL1-.99E-4) 1048,1002,1002
 1002 TPX = X
      TPY = Y
      GO TO 300
 1020 IF(ABS(A)-TOL1) 1046,1046,200
 1030 IF(NTR) 1032,1032,1038
 1032 IF(FX) 1034,247,1034
```

```
1034 DX = -A/FX
     IF(ABS(DX)-TANG) 1036,1036,247
1036 TPX = TPX+DX
     GO TO 1044
1038 IF(FY) 1040,247,1040
1040 DY = -A/FY
     IF(ABS(DY)-TANG) 1042,1042,247
1042 TPY = TPY+DY
1044 NIA = NIA + 1
     IF(NIA-NAI) 300,247,247
1046 ISW = 0
     X = TPX
     Y = TPY
     RETURN
1048 TPXD = X
     TPYD = Y
     GO TO 700
1050 IF(ABS(AD)-TOL1) 1076,1076,600
1060 IF(NTR) 1062,1062,1068
1062 IF(FXD) 1064,247,1064
1064 DXD = -AD/FXD
     IF(ABS(DXD)-TANG) 1066,1066,247
1066 TPXD = TPXD + DXD
     GO TO 1074
1068 IF(FYD) 1070,247,1070
1070 DYD = -AD/FYD
     IF(ABS(DYD)-TANG) 1072,1072,247
1072 TPYD = TPYD + DYD
1074 NIA = NIA + 1
     IF(NIA-NAI) 700,247,247
1076 X = TPXD
     Y = TPYD
     GO TO 888
     END
```

```
      BLOCK DATA
      COMMON /RTLOCS/
     1        PI1D,PI2D,TWOPID,PI1,PI2,TWOPI,ICW,NIA,NID,NSTEP,MODE,
     2        SGNLT(3),DUM1(2),NPL,NZL,DUM2,TAU,DUM3(6),
     3        PR(100),PI(100),XMP(100),ZR(100),ZI(100),XMZ(100),MOUT,
     4        MGAIN,DUM4,ICUTSW,NPREC,IDCHK,NCUTS,NORDP,NORDZ,NRUN,
     5        MPLOT,SCALE(4),NXINT,MSCALE,MGRID,MARKT,MSYMX,KPLSW,MPLOTC,
     6        SCALEC(4),NXINTC,MSCALC,MGRIDC,MARKTC,MSYMXC,DUM5(65),
     7        IT(4),ITSW,TOLBRK,TOLANG,CHGI,DUM6,
     8        KDSTOR(2,1501)
      DATA PI1D/3.14159265358979/
      DATA CHGI/0.40/,TOLBRK,TOLANG/2*1.0E-12/,PI1/3.141593/
      DATA TWOPI/6.283185/,XMP,XMZ/200*1.0/,NXINT,NXINTC/2*96/
      DATA MSCALE,MSCALC/2*6/,MSYMX/-2/,NCUTS/25/
      DATA SGNLT/2HTA,2HBT,2HAI/
      DATA KGAIN,NPREC,MOUT,MGAIN,MPLOT,MGRID,MGRIDC,MSYMXC,IDCHK
     1     / 9*1/
      DATA PR,PI,ZR,ZI/400*0.0/
      DATA SCALE,SCALEC/8*0.0/
      DATA IT/4*0/
      DATA NPL,NZL,ICROSS,ICUTSW,NORDP,NORDZ,MARKT,KPLSW,MPLOTC,
     1     MARKTC,ITSW,NRUN/12*0/
      DATA MODE/0/
      DATA NIA,NID,NSTEP/3*0/
      DATA TAU /0.0/
      END
```

```
      SUBROUTINE ARCTAN(NF,X,Y,OUT,XD,YD,OUTD)
      GO TO (100,300),NF
100   IF(X) 110,200,110
110   ARG = Y/X
      IF(X) 140,200,130
130   OUT = ATAN(ARG)
      RETURN
140   IF(Y) 160,160,150
150   OUT = ATAN(ARG) + 3.141593
      RETURN
160   OUT = ATAN(ARG) - 3.141593
      RETURN
200   IF(Y) 210,220,230
210   OUT = -1.570796
      RETURN
220   OUT = 0.
      RETURN
230   OUT = 1.570796
      RETURN
300   IF(XD) 310,400,310
310   ARGD = YD/XD
      IF(XD) 340,400,330
330   OUTD = ATAN(ARGD)
      RETURN
340   IF(YD) 360,360,350
350   OUTD = ATAN(ARGD) + 3.14159265358979
      RETURN
360   OUTD = ATAN(ARGD) - 3.14159265358979
      RETURN
400   IF(YD) 410,420,430
410   OUTD = -1.57079632679490
      RETURN
420   OUTD = 0.
      RETURN
430   OUTD =  1.57079632679490
      RETURN
      END
```

```
      SUBROUTINE POLYRT(M,C,TOL1,RX,RY,RMULT,NR,ISW)
C
C     THIS SUBROUTINE CALCULATES THE REAL AND COMPLEX ROOTS
C     OF A REAL POLYNOMIAL
C
C     DESCRIPTION OF PARAMETERS
C           M     -ORDER OF POLYNOMIAL(MAXIMUM ORDER=99)
C           C     -VECTOR OF M+1 COEFFICIENTS OF POLYNOMIAL
C                  ORDERED FROM LARGEST TO SMALLEST
C           TOL1 -TOLERANCE OF ROOTS
C           RX    -REAL ROOTS
C           RY    -RESPECTIVE IMAGINARY ROOTS
C           RMULT-MULTIPLICITY OF ROOTS
C           NR    -NUMBER OF UNIQUE ROOTS
C           ISW   -ERROR CODE WHERE
C                  ISW=1  NO ERROR
C                  ISW=2  UNABLE TO DETERMINE ROOT
C                  ISW=3  M GREATER THAN 99
C
      DIMENSION C(100),RX(100),RY(100),RMULT(100),RTX(2),RTY(2),IRX(2),
     1          IRY(2),FRX(2),FRY(2),L(4),NRF(4)
      DIMENSION A(100,4),RF(4),P(2),Q(2)
      EQUIVALENCE (X,FRX(1),IRX(1)),(Y,FRY(1),IRY(1))
      FCTR = 4294967296.
      TOLM = TOL1*TOL1
      N = M + 1
      NR = 0
      DO 2000 I=1,N
      IF(C(I)) 2002,2000,2002
 2000 CONTINUE
 2002 N = N-I+1
      IF(N-100) 2010,2010,2005
 2005 ISW = 3
 2008 NR = -1
      RETURN
 2010 L(3) =N
      L(4) =N-1
      K= I-1
      DO 2015 I=1,N
      J = I+K
 2015 A(I,1) = C(J)
      NZR = 0
 2020 IF(N-1) 2060,2044,2022
 2022 IF(A(N,1)) 2024,2023,2024
 2023 NZR = NZR +1
      N = N - 1
      GO TO 2020
 2024 T1 = ABS(A(N,1))
      T2 = 16.*ABS(A(1,1))
      I = 0
 2028 IF(T1-T2)2032,2032,2030
 2030 T1 = T1/16.
```

```
      I = I+1
      GO TO 2028
 2032 NF = (4*I)/(N-1)
      RFCTR = 1.
      IF(NF) 2044,2C44,2034
 2034 DO 2035 I=1,NF
 2035 RFCTR = RFCTR*2.
      T1 = 1.
      K = 0
      DO 2042 I=2,N
      IF(T1-1.E5C) 204C,204C,2C36
 2036 K = K+1
      DO 2038 J=1,K
 2038 A(I,1) = A(I,1)/RFCTR
      GO TO 2042
 2040 T1 = T1*RFCTR
 2042 A(I,1) = A(I,1)/T1
 2044 K = L(3)
      DO 2045 I=1,K
      A(I,3) =A(I,1)
 2045 A(I,4) = A(I,3)*(N-I)
      IF(NZR) 205C,2C5C,2C48
 2048 RX(1) = C.
      RY(1) = C.
      RMULT(1) = NZR
      NR = 1
 2050 NT = N-1
      L(1) = N
      L(2) = NT
      DO 2056 I=1,N
 2056 A(I,2) = A(I,1)*(N-I)
      IF(N-1) 206C,2C6C,21CC
 2060 ISW = 1
      RETURN
 2100 XO = .5CC1C1E-2
      YO = .1CC1C1E-1
      NSG = 0
      ACFG = 1.
 2110 DO 2111 I=1,4
      NRF(I) = C
 2111 RF(I) = 1.
      IF(NSG-NSG/2) 2112,2112,2113
 2112 XO = -1C.*XC
      YO = -10.*YO
 2113 T = YO
      YC =XO
      XO =T
      NSG = NSG+1
      ACFG =ACFG*10CC.
      XR = XO
      YR = YC
      K1 = 0
```

```
      NI = 1
      NFI = 0
2114  IVSW = -2
      ARL = 1.E50
2118  K = 0
2120  K = K+1
      K1 = K1+1
      P(K) = A(1,K1)*RF(K1)
      Q(K) = 0.
      NRT = L(K1)
      IF(NRT-2) 2170,2121,2121
2121  DO 2165 I=2,NRT
      T = P(K)*XR - Q(K)*YR + A(I,K1)*RF(K1)
      Q(K) = P(K)*YR + Q(K)*XR
      P(K) = T
2122  ARS = ABS(P(K)) + ABS(Q(K))
      IF(ARS-1.E30) 2130,2130,2124
2124  P(K) = P(K)/FCTR
      Q(K) = Q(K)/FCTR
      IF(NRF(K1)-7) 2126,2128,2128
2126  RF(K1) = RF(K1)/FCTR
2128  NRF(K1) = NRF(K1) + 1
      GO TO 2122
2130  IF(ARS-1.E-20) 2132,2165,2165
2132  IF(NRF(K1)) 2165,2165,2134
2134  P(K) = P(K)*FCTR
      Q(K) = Q(K)*FCTR
      IF(NRF(K1)-7)2136,2136,2138
2136  RF(K1) = RF(K1)*FCTR
2138  NRF(K1) = NRF(K1) - 1
      ARS = ABS(P(K)) + ABS(Q(K))
      GO TO 2130
2165  CONTINUE
2170  IF(K-2) 2120,2190,2190
2190  DIV = P(2)*P(2) + Q(2)*Q(2)
      IF(DIV) 2196,2207,2196
2196  DX = -(P(1)*P(2) +Q(1)*Q(2))/DIV
      DY = (P(1)*Q(2)-Q(1)*P(2))/DIV
      NRT = NRF(K1) - NRF(K1-1)
      IF(NRT) 2198,2205,2200
2198  NRT = -NRT
      T2 = 1./FCTR
      GO TO 2202
2200  T2 = FCTR
2202  DO 2204 I=1,NRT
      DX = DX/T2
2204  DY = DY/T2
2205  IF(ABS(DX)+ABS(DY) - ACHG) 2207,2207,2206
2206  DX = SIGN(ACHG/1000.,DX)
      DY = SIGN(ACHG/1000.,DY)
      IVSW = -2
2207  XR = XR + DX
```

```
      YR = YR + DY
      ARS = ABS(DX)+ABS(DY)
      IF(ARS-(ABS(XR)+ABS(YR))*1.E-6) 2216,2216,2228
2216  IF(ARS-ARL) 2227,2218,2218
2218  IVSW = IVSW+1
      IF(IVSW) 2227,2227,2224
2224  IF(NFI) 2226,2226,2250
2226  NFI = 1
      XT =XR
      YT =YR
      K1 =2
      GO TO 2114
2227  ARL = ARS
2228  IF(NFI) 2230,2230,2240
2230  NI = NI + 1
      K1 = 0
      IF(NI-500) 2118,2118,2232
2232  IF(NSG-8) 2110,2234,2234
2234  ISW = 2
      GO TO 2008
2240  NFI = NFI+1
      K1 = 2
      IF(NFI-100) 2118,2118,2250
2250  IF(ABS(XR-XT)+ABS(YR-YT)-.1)2295,2295,2260
2260  IF(NFI-100) 2262,2262,2280
2262  IF(ARS-1.E-6) 2264,2264,2280
2264  RTX(1) = XR*RFCTR
      RTY(1) = YR*RFCTR
      DO 2265 I=1,NR
      IF(ABS(RTX(1)-RX(I))+ABS(RTY(1)-RY(I))-TOL1) 2280,2280,2265
2265  CONTINUE
      GO TO 2295
2280  IF(NSG-4) 2110,2290,2290
2290  X = XT
      Y = YT
      GO TO 2300
2295  X = XR
      Y = YR
2300  IF(ABS(Y)-1.E-4) 2305,2315,2315
2305  Y = 0.
      DO 2310 I=2,NT
2310  A(I,1) = A(I,1)+X*A(I-1,1)
      GO TO 2325
2315  T = X+X
      DIV = -(X*X + Y*Y)
      A(2,1) = A(2,1) + T*A(1,1)
      DO 2320 I=3,NT
2320  A(I,1) = A(I,1) + T*A(I-1,1) + DIV*A(I-2,1)
2325  X = X*RFCTR
      Y = Y*RFCTR
      IF(IRX(2)) 2330,2340,2340
2330  IRX(1) = IRX(1) + 1
```

```
      IF(FRX(1)) 2340,2335,2340
 2335 IRX(1) = IRX(1) + 1048576
 2340 IF(IRY(2)) 2345,2355,2355
 2345 IRY(1) = IRY(1) + 1
      IF(FRY(1)) 2355,2350,2355
 2350 IRY(1) = IRY(1) +1048576
 2355 RTX(1) = X
      RTY(1) = Y
      IF(Y) 2365,2360,2365
 2360 NRT = 1
      GO TO 2400
 2365 NRT = 2
      RTX(2) = X
      RTY(2) = -Y
 2400 N = N-NRT
      DO 2470 I=1,NRT
      J = 1
      JFL = 0
 2405 IF(J-NR) 2410,2410,2455
 2410 IF((RX(J)-RTX(I))*(RX(J)-RTX(I))+(RY(J)-RTY(I))*(RY(J)-RTY(I))
     1   -TOLM) 2415,2415,2420
 2415 RMULT(J) = RMULT(J) + 1.
      GO TO 2462
 2420 IF(RX(J)-RTX(I)) 2440,2425,2435
 2425 IF(ABS(RY(J))-ABS(SNGL(RTY(I)))) 2435,2430,2436
 2430 IF(RY(J)) 2436,2436,2435
 2435 J = J+1
      GO TO 2405
 2436 IF(JFL) 2438,2438,2435
 2438 JFL = J
      GO TO 2435
 2440 IF(RTX(I)-RX(J)-TOL1) 2436,2441,2441
 2441 IF(JFL) 2444,2444,2442
 2442 J = JFL
 2444 K = NR+1
 2445 IF(K-J) 2460,2460,2450
 2450 K1 = K-1
      RX(K) = RX(K1)
      RY(K) = RY(K1)
      RMULT(K) = RMULT(K1)
      K = K1
      GO TO 2445
 2455 IF(JFL) 2460,2460,2442
 2460 RMULT(J) = 1.
      NR = NR+1
 2462 RX(J)= RTX(I)
      NR1 = NR1 + 1
 2470 RY(J) = RTY(I)
      GO TO 2050
      END
```

```
      SUBROUTINE RTLCSV
C     MAIN OF RTLOCV SUBROUTININZED IN ITS ENTIRITY.
C     IT IS USED IN CONJUNCTION WITH  SKBJI WITH LOGIC TO SET
C     SPECIFIED ELEMENTS TO ZERO AND ONE THUS FINDING THE OPEN LOOP
C     POLES AND ZEROES WHICH ARE PASSED TO THE ROOT LOCUS PROGRAM.
      COMMON COMCN(14CC)
      COMMON /RTLOCS/
     1       PI1C,PI2D,TWOPID,PI1,PI2,TWOPI,ICW,NIA,NIC,NSTEP,MODE,
     2       SGNLT(3),CELT2,TOLLH,NPL,NZL,DUM1,TAU,DUM2(6),
     3       PR(100),PI(1CC),XMP(100),ZR(100),ZI(100),XMZ(100),MOUT,
     4       MGAIN,CUM3,IOLTSW,NPREC,IOCHK,NCUTS,NORDP,NORDZ,NRUN,
     5       MPLCT,SCALE(4),NXINT,MSCALE,MGRID,MARKT,MSYMX,KPLSW,MPLOTC,
     6       SCALEC(4),NXINTC,MSCALC,MGRIDC,MARKTC,MSYMXC,EATE(5),
     7       CUM4(6C),IT(4),ITSW,TOLBRK,TCLANG,CHGI,GNFCTR,
     8       KDSTOR(2,1501)
      DIMENSION TITLE(10),BNDY(4)
      DIMENSION SGNL(6),NLOC(99),NTYPE(99),DANG(10),IBLK(3),SC(4),NE(3),
     1          SCC(4),NSYMB(3),RLX(3),RLY(3),ANG(3),ISGNL(3),IOUT(3),
     2          IGAIN(3),IPLOT(3),IPLOTC(3),NL(3),FLAG(3),APX(3),APY(3),
     3          SIGNAL(3),IOFLAG(3),IGFLAG(3),IGFL(3),GAIN(3),RL(2),
     4          RB(2),DELTA(2),SX(1CC),SY(100),KSTOR(2,750),
     5          KSTORC(2,75C)
      EQUIVALENCE (BNDY   ,CCMON(4346))
      EQUIVALENCE (INDCLS,CCMON( 725))
      EQUIVALENCE (KGAIN  ,CCMCN(434C))
      EQUIVALENCE (DELT   ,COMCN(4341))
      EQUIVALENCE (TOL    ,COMON(4342))
      EQUIVALENCE (ICROSS,CCMCN(4343))
      EQUIVALENCE (TITLE  ,COMCN( 795))
      EQUIVALENCE(KSTOR(1,1),KDSTOR(1,1)),(KSTCRC(1,1),KDSTOR(2,1))
      DATA M/O/
      DATA E/1.0/
      DATA SGNL  /1H  ,1HB,1HE,1HZ,1HP,1HS/
      DATA NSYM  /1/,NSTOR,NSTCRC   /2*1500/
      DATA NTYPE /3*-2,4*-1,-1,-2,6*-1,16*-2,-1,9*-2,-1,7*-2,5,15,30*-2,
     1    3*-1,15*-2/
      DATA NLOC/19,20,21,22,23,24,25,733,-1,29,129,229,329,429,529,
     1    -1,-1,-1,-1,629,630,631,632,633,634,635,636,637,638,-1,639,
     2    640,644,645,646,647,648,-1,-1,649,650,651,655,656,657,658,
     3    659,-1,-1,660,665,28*-1,725,729,730,731,732,15*-1/
      DATA TRUN/C.C/
      DATA FLAG /3*C.C/
      DATA IOFLAG,IGFLAG,IGFL/9*C/
 1    FORMAT (1H ,T51,30HROOT LOCUS CALCULATION PROGRAM/
     .       T59,10HRUN NUMBER,I3//(20X,10A6))
 2    FORMAT (//6X'SYSTEM ORDER'11X'-'I4/6X'NO. OF OPEN LOOP POLES -'I4/
     1    6X'NO. OF OPEN LOOP ZEROES -'I4/6X'TIME DELAY',T30,'-'1PE13.4
     2       //6X'OPEN LOOP'6X'LOCATIONS')
 3    FORMAT (6X,2(F13.5,2X),I7,T67,2(F13.5,2X),I7)
 4    FORMAT (6X,2(F13.5,2X),I7)
 5    FORMAT (T67,2(F13.5,2X),I7)
 6    FORMAT(//6X'STEP SIZE',T21,'-'F9.5,T67,'PRINT INTERVAL',T92,'-'I4/
```

```
     1     6X'ACCURACY',T21,'-'1PE9.1,T67,'GAIN INTERVAL',T92,'-'I4/
     2     6X'GAIN MODE',T21,'-',T67,'PLOT INTERVAL',T92,'-'I4)
7     FORMAT (1H+,T24,'POSITIVE')
8     FORMAT (1H+,T24,'NEGATIVE')
9     FORMAT (  11X'REAL'8X'IMAGINARY'4X,'MULTIPLICITY'/)
10    FORMAT (6X'BYPASSES ON - STEP SIZE ADJUSTMENT IN CRITICAL AREAS')
11    FORMAT (1H+,65X,*ROOT LOCUS - RUN NUMBER*,I3//)
12    FORMAT (3(14X'BRANCH NUMBER'I3,14X)/3(4X'LOCUS-REAL'3X'LOCUS-IMAG'
     1         6X'GAIN'7X)/)
13    FORMAT(1H+,T17,'POLE',T67,'OPEN LOOP ZERO LOCATIONS'/2(11X'REAL'8X
     1         'IMAGINARY'4X'MULTIPLICITY'12X)/)
14    FORMAT (2(14X'BRANCH NUMBER'I3,14X)/2(4X'LOCUS-REAL'3X'LOCUS-IMAG'
     1         6X'GAIN'7X)/)
15    FORMAT (14X'BRANCH NUMBER'I3,14X/4X'LOCUS-REAL'3X'LOCUS IMAG'5X
     1         ' GAIN'7X/)
17    FORMAT (/6X,'TIME',I2,' =',F7.2,' SEC.')
18    FORMAT (1H+,T17,'POLE')
19    FORMAT (1H+,T17,'ZERO')
20    FORMAT (3(1XA1,F12.5,F13.5,F14.5,3X))
21    FORMAT (1XA1,F12.5,F13.5,F14.5,48XA1,F12.5,F13.5,F14.5)
22    FORMAT (  44X2(1XA1,F12.5,F13.5,F14.5,3X))
23    FORMAT (89XA1,F12.5,F13.5,F14.5)
24    FORMAT (3(1XA1,F12.5,F13.5,17X))
25    FORMAT (1XA1,F12.5,F13.5,62XA1,F12.5,F13.5)
26    FORMAT (44X2(1XA1,F12.5,F13.5,17X))
27    FORMAT (89XA1,F12.5,F13.5)
28    FORMAT (1H+26XF14.5,30XF14.5)
29    FORMAT(1H+26XF14.5,74XF14.5)
30    FORMAT (1H+70XF14.5,30XF14.5)
31    FORMAT (1H+114XF14.5)
32    FORMAT (6X,'TIME FOR RUN NO.',I3,' =',F7.2,' SEC.')
33    FORMAT (6X'ELAPSED JOB TIME =',F8.2,' SEC.')
34    FORMAT (1H+,26X,F11.2,'E'I3,2(29X,F11.2,'E'I3))
35    FORMAT (1H+,26X,F11.2,'E'I3,73X,F11.2,'E'I3)
36    FORMAT (1H+,41X,2(29X,F11.2,'E'I3))
37    FORMAT (1H+,114X,F11.2,'E'I3)
38    FORMAT (1H+,T33,'(NORMALIZED)')
39    FORMAT (1X,A1,1P4(5X,E14.6,4X),0PF15.6,'E'I3,3X,3I4)
40    FORMAT (T61,'BRANCH NUMBER'I3///11X'LOCUS-REAL'12X'LOCUS-IMAG'13X
     1         'ANGLE ERROR'12X'SLOPE ANGLE'15X'GAIN'8X'NIA NID NSTEP')
41    FORMAT (' (AI LINES- TPX'9X'TPY'10X'DENOM'10X'SANG'10X'ANGT'10X
     1         'DTHETA'8X'STEP'7X'NIA   NID    NF)')
42    FORMAT ('  (BT LINES- TPX'9X'TPY'10X'DX'13X'DY'12X'FX'12X'FY'12X
     1         'STEP'7X'J)')
43    FORMAT (' (TA LINES- TPX'9X'TPY'10X'ANG'12X'TANG'10X'FX'12X'FY'12X
     1         'STEP'7X'NIA   NID NSTEP   NF)')
44    FORMAT (1H )
45    FORMAT(6X'GAIN CONSTANT -'F9.5,T67,'COMPARISON PLOT INTERVAL -'I4)
46    FORMAT (1H+,T31,'E',I3)
47    FORMAT (//6X'REGION OF CALCULATION - REAL -'F11.4,3X'TO'F11.4/T31
     1       ,'IMAG -'F11.4,3X'TO'F11.4//6X'CRITICAL POINT INDICATORS - '
     2  ,'P - OPEN LOOP POLE'/T35,'Z - OPEN LOOP ZERO'/T35,'E - BOUND'
```

```
     3    .'ARY POINT'/T35,'B - BREAKAWAY POINT'/T35,'S - PROGRAM STUCK'
     4       .' - BRANCH TERMINATED'/)
  54  FORMAT (1X,A1,1PE23.13,0PF15.6,'E'I3,3X,3I4)
  55  FORMAT (1H ,T39,'BOUNDARY SEARCH FOR ADDITIONAL STARTING POINTS -'
     1    .' RESULTS'/T39,20X,T81,'RUN NUMBER'I3///17X'POINT NO.'8X'REAL'
     2         13X'IMAGINARY'10X'ANGLE ERROR'8X'STARTING ANGLE'7X'NI'/)
  56  FORMAT (19XI3,1P4E20.6,8XI2)
  57  FORMAT (22X1P4E20.6,8XI2,3X'(NOT START PT.)')
  58  FORMAT (////17X'NO ADDITIONAL STARTING POINTS FOUND')
  59  FORMAT (1H0,T44,*IMAGINARY AXIS CROSSING POINTS - RUN NUMBER*,I3//
     ./T44,*IMAG. VALUE*,27X,*GAIN*/)
  60  FORMAT (T40,1PE21.15)
  61  FORMAT (T35,F20.6)
  62  FORMAT (T44,'ROOT LOCUS DOES NOT CROSS IMAGINARY AXIS'/T46,'BETWE'
     1      ,'EN',F11.4,3X'AND',F11.4)
  63  FORMAT (1H+,T12,'PAGE'I3)
  64  FORMAT (1H+,'RUN NUMBER',I3,' NOT EXECUTED - ')
  65  FORMAT (1H ,T31,'POLE POLYNOMIAL UNSOLVABLE')
  66  FORMAT (1H ,T31,'POLE POLYNOMIAL ORDER EXCEEDS 99')
  67  FORMAT (1H ,T31,'ZERO POLYNOMIAL UNSOLVABLE')
  68  FORMAT (1H ,T31,'ZERO POLYNOMIAL ORDER EXCEEDS 99')
  69  FORMAT (1H+,T58,'ON PREVIOUS RUN - NO NEW INFORMATION GIVEN')
  70  FORMAT (1H ,T31,'ZERO ORDER SYSTEM - NPL,NZL = 0')
  71  FORMAT(1H+,T75,F11.2,'E'I3)
  72  FORMAT (1H+,T75,F15.6)
  73  FORMAT(1XA1,1P4(5XE14.6,4X),22X3I4)
  74  FORMAT (1XA1,1P4E23.13,22X3I4)
  75  FORMAT (/6X'BRANCH STARTING POINTS ON BOUNDARY'/11X'REAL'8X'IMAGI'
     1        ,'NARY'//(6XF13.5,2XF13.5))
  76  FORMAT (//' COMPARISON PLOT CANNOT BE MADE-NO CURVES ACCUMULATED')
  77  FORMAT (6X'BYPASSES ON - LOCUS DISTANCE CHECK')
  78  FORMAT (T21,'LOCUS DISTANCE CHECK')
  79  FORMAT (6X'PLANT GAIN',T21,'-'1PE13.5)
C
C     SETUP COMPUTATIONS
C
C     ERRSET SUBROUTINE STOPS TRACEBACK MESSAGES FOR UNDERFLOWS WHICH
C     COMMONLY OCCUR WHEN FINDING ROOTS OF HIGH ORDER POLYNOMIALS
      CALL ERRSET(208,256,-1,1,0,208)
      DUM2(3)=BNDY(1)
      DUM2(4)=BNDY(2)
      DUM2(5)=BNDY(3)
      DUM2(6)=BNDY(4)
      DUM1=KGAIN
      DUM2(1)=DELT
      DUM2(2)=TOL
      IF(NRUN.NE.0)GO TO 5101
      ISWXC=10
      T0=0
      IOW=6
      IDCHK = 1
      PI2D = PI10/2
```

```
      TWOPID = PI1D*2
      PI2 = PI1/2
      DANG(1) = PI2
      DANG(2) = .01
      DO 90 I=3,10
 90   DANG(I) = PI1*(1.-1./I)
C     INITIAL ROUTINE
 100  NPOINT = 10
      ISWX=10
      GO TO 993
 101  IF(NRUN.NE.0)RETURN
 5101 NRL=NRUN
      NRUN = NRUN +1
C     CALL INPUT(NLOC,NTYPE,KWIT)
      IF(KWIT.EQ.1)RETURN
      IF(KPLSW) 9101,9101,6101
C     ROUTINE FOR COMPARISON PLOT ONLY
 6101 NRUN = NRL
      IF(ISWXC) 7101,7101,8101
 7101 WRITE(IOW,76)
      GO TO 990
 8101 KSW = 2
      ISWXC = 4
      GO TO 970
 9101 TOLLH = TOL/DELT
      DELT2 = (DELT+TOL)*(DELT+TOL)
      IF(NPREC) 102,102,103
 102  ACHG = 1.
      BCHG = 1.
      DELTR = DELT
      DTOL = DELT
      GO TO 104
 103  ACHG = CHGI
      BCHG = .7
      DELTR = DELT/100.
      DTOL = 2.*DELTR
 104  DO 105 I=1,3
 105  IBLK(I) = 0
      DO 110 I=1,3,2
      IF(SCALE(I+1)-SCALE(I)) 106,106,108
  106 SC(I) = BNDY(I)
      SC(I+1) = BNDY(I+1)
      GO TO 110
  108 SC(I) = SCALE(I)
      SC(I+1) = SCALE(I+1)
  110 CONTINUE
      IF(NSYM-1) 115,115,122
  115 DO 120 I=1,3,2
      IF(SCALEC(I+1)-SCALEC(I)) 116,116,118
 116  SCC(I) = BNDY(I)
      SCC(I+1) = BNDY(I+1)
      GO TO 120
```

```
118   SCC(I) = SCALEC(I)
      SCC(I+1) = SCALEC(I+1)
120   CONTINUE
122   K3 = 0
      DO 125 I=1,3
125   NSYMB(I) = 1
      IF(IOUTSW-1) 134,136,130
130   IF(TOL-1.E-4) 132,136,136
132   IOSW = -1
      GO TO 138
134   IOSW = 0
      IF(IABS(IT(1))+IABS(IT(2))+IABS(IT(3))) 140,140,136
136   IOSW = 1
138   NE(2) = 1
140   IF(NORDP) 142,160,146
142   IF(NPL) 144,158,158
144   CONTINUE
      CALL CLASS(INDCLS)
      WRITE(6,80)TITLE
80    FORMAT(1H ,10X,10A6)
      WRITE(6,65)
      WRITE(IOW,69)
      GO TO 184
146   CALL POLYRT(NORDP,PR,TOL,SX,SY,XMP,NPL,ISW)
      NPOINT = 8
      GO TO 992
147   IF(ISW-2) 154,148,150
148   CONTINUE
      CALL CLASS(INDCLS)
      WRITE(6,80)TITLE
      WRITE(6,65)
      GO TO 152
150   CONTINUE
      CALL CLASS(INDCLS)
      WRITE(6,80)TITLE
      WRITE(6,66)
152   NORDP = -1
      GO TO 184
154   IF(NPL) 158,158,155
155   DO 156 I=1,NPL
      PR(I) = SX(I)
156   PI(I) = SY(I)
158   NORDP = 0
160   IF(NORDZ) 162,180,166
162   IF(NZL) 164,178,178
164   CONTINUE
      CALL CLASS(INDCLS)
      WRITE(6,80)TITLE
      WRITE(6,67)
      WRITE(IOW,69)
      GO TO 184
166   CALL POLYRT(NORDZ,ZR,TOL,SX,SY,XMZ,NZL,ISW)
```

```
      NPOINT =9
      GO TO 992
167   IF(ISW-2) 174,168,17C
168   CONTINUE
      CALL CLASS(INDCLS)
      WRITE(6,8C)TITLE
      WRITE(6,67)
      GO TO 172
170   CONTINUE
      CALL CLASS(INDCLS)
      WRITE(6,8C)TITLE
      WRITE(6,68)
172   NORDZ = -1
      GO TO 184
174   IF(NZL) 178,178,175
175   DO 176 I=1,NZL
      ZR(I) = SX(I)
176   ZI(I) = SY(I)
178   NORDZ = 0
180   IF(NPL+NZL) 182,182,2CC
182   CONTINUE
      CALL CLASS(INDCLS)
      WRITE(6,8C)TITLE
      WRITE(6,7C)
184   WRITE(IOW,64)NRUN
      GO TO 10C
200   NP = 0
      IF(NPL) 210,21C,202
202   DO 204 I=1,NPL
204   NP = NP + XMP(I) .
210   NZ = 0
      IF(NZL) 225,225,212
212   DO 214 I=1,NZL
214   NZ = NZ + XMZ(I)
225   IF(NP-NZ) 236,236,238
236   NBR =NZ
      GO TO 24C
238   NBR = NP
240   NORD = NBR
245   IF(NP-NZ) 265,26C,26C
260   MODE = C
      GO TO 275
265   MODE = 1
275   IF(IABS(KGAIN)-1) 282,28C,282
280   NGAIN = 7
      CALL RTLOC(6,0,0.,0.,E,E,E,E,3C,GAIN(1),M,NE(1),D,D,D,D)
      GO TO 285
282   NGAIN = 6
      GAIN(1) = 1
      NE(1) = C
285   IF(IDCHK) 290,29C,288
288   NLC = 1
```

```
      GO TO 300
290   NLC = 2
300   NSP = 0
      IF(NCUTS) 380,380,310
310   NPOINT = 6
      GO TO 992
311   DO 312 I=1,2
      DELTA(I) = (BNDY(2*I)-BNDY(2*I-1))/NCUTS
312   RL(I) = BNDY(2*I-1)
      BPX = BNDY(4)
      SGN = 1.
      I = 2
      K = 2
      IF(IT(4)) 314,315,314
314   CONTINUE
      CALL CLASS(INDCLS)
      WRITE(6,80)TITLE
      WRITE(6,55) NRUN
315   RB(1) = RL(1)
      RB(2) = RL(2)
      RL1 = RL(I) - SIGN(DELTA(I),SGN)
      CALL RTLOC(5,0,RL(1),RL(2),E,E,E,E,M,SANGL,M,M,D,D,D,D)
      IF(SANGL) 320,350,320
320   RL(I) = RL(I) + SIGN(DELTA(I),SGN)
      RB(I) = RL(I)
      CALL RTLOC(5,0,RL(1),RL(2),E,E,E,E,M,SANG,M,M,D,D,D,D)
      IF(SANG*SANGL) 350,350,325
325   SANGL = SANG
      IF(SGN*(RL(I)-BPX)) 320,330,330
330   I = I - 1
      IF(I) 340,340,335
335   RL(2) = BNDY(K+2)
      BPX = BNDY(K)
      K = K - 1
      GO TO 315
340   IF(SGN) 376,376,345
345   SGN = -1.
      I = 2
      RL(1) = BNDY(2)
      BPX = BNDY(3)
      GO TO 315
350   CALL RTLOC(I+7,0,RB(1),RB(2),E,DELTR,1.E-12,DELTA(I),15,TANG,ISW,
     1          M,D,D,D,D)
      IF(ISW) 354,354,325
354   IF(ABS(RB(I)-RL1)-TOL) 325,325,355
355   IF(SGN*(RB(I)-BPX)) 356,330,330
356   RL(I) = RB(I) + SIGN(DELT,SGN)
      RL1 = RB(I)
      CALL RTLOC(4,0,RB(1),RB(2),C.,DELTR,E,10.,M,ANGLE,ISW,M,D,D,D,D)
      IF(ISW-1) 358,358,325
358   IF(I-1) 360,360,365
360   IF(SGN*ANGLE) 370,367,367
```

```
365   IF(SGN*(ABS(ANGLE)-PI2)) 370,367,367
367   IF(IT(4)) 368,315,315
368   WRITE(IOW,57) RB(1),RB(2),TANG,ANGLE,NIA
      GO TO 315
370   NSP = NSP + 1
      SX(NSP) = RB(1)
      IF(ABS(RB(2))-TOL) 371,371,372
371   SY(NSP) = C.
      GO TO 373
372   SY(NSP) = RB(2)
373   IF(IT(4)) 374,375,374
374   WRITE(IOW,56) NSP,RB(1),RB(2),TANG,ANGLE,NIA
375   IF(NSP-100) 315,377,377
376   IF(NSP) 377,377,379
377   IF(IT(4)) 378,379,378
378   WRITE(IOW,58)
379   NPOINT = 7
      GO TO 992
380   NB = NBR + NSP
C     WRITE HEADING
      CALL CLASS(INDCLS)
400   WRITE(IOW,1) NRUN,TITLE
      WRITE(IOW,2) NORD,NP,NZ,TAU
      IF(NPL-NZL) 405,425,430
405   IF(NPL) 415,415,410
410   JSW = 2
      GO TO 420
415   WRITE(IOW,19)
      WRITE(IOW,9)
      DO 418 I=1,NZL
      N = XMZ(I)
418   WRITE(IOW,4) ZR(I),ZI(I),N
      GO TO 480
420   MB1 =NPL
      MB2 = NPL+1
      MB3 = NZL
      GO TO 450
425   JSW = 1
      MB1 = NZL
      GO TO 450
430   IF(NZL) 440,440,435
435   JSW = 3
      GO TO 445
440   JSW = 4
445   MB1 = NZL
      MB2 = NZL + 1
      MB3 = NPL
450   IF(JSW-2) 455,455,468
455   WRITE(IOW,13)
      DO 458 I=1,MB1
      M = XMP(I)
      N = XMZ(I)
```

```
 458    WRITE(IOW,3) PR(I),PI(I),M,ZR(I),ZI(I),N
        GO TO (480,460,470),JSW
 460    DO 465 I=MB2,MB3
        N =XMZ(I)
 465    WRITE(IOW,5) ZR(I),ZI(I),N
        GO TO 480
 468    WRITE(IOW,18)
        WRITE(IOW,9)
 470    DO 475 I=MB2,MB3
        M = XMP(I)
 475    WRITE(IOW,4) PR(I),PI(I),M
 480    IF(NSP) 484,484,482
 482    WRITE(IOW,75) (SX(I),SY(I),I=1,NSP)
 484    WRITE(IOW,6) DELT,MOLT,TOL,MGAIN,MPLOT
        IF(KGAIN) 486,488,488
 486    WRITE(IOW,8)
        GO TO 490
 488    WRITE(IOW,7)
 490    IF(NGAIN-7) 492,491,491
 491    WRITE(IOW,38)
 492    WRITE(IOW,45) GAIN(1),MPLOTC
        IF(NE(1)) 493,494,493
 493    WRITE(IOW,46) NE(1)
 494    WRITE(IOW,79) GNFCTR
        WRITE(IOW,47) BNDY
        IF(NPREC) 497,497,495
 495    IF(IDCHK) 496,496,498
 496    WRITE(IOW,77)
        GO TO 498
 497    WRITE(IOW,10)
        IF(IDCHK) 4971,4971,498
 4971   WRITE(IOW,78)
 498    K = 1
        K1 = 0
        L = 1
        NBI= 1
C       STARTING POINT CALCULATION ROUTINE
 500    NAB = K-NBR
        IF(NAB) 505,505,502
 502    RLX(L) = SX(NAB)
        RLY(L) = SY(NAB)
        MP = 1
        ISGNL(L) = 1
        CALL RTLOC(4,0,RLX(L),RLY(L),E,0.,E,E,M,ANG(L),ISW,NT,D,D,D,D)
        GO TO (550,546,547,550),ISW
 505    J= 1
        XJ = 0.
        XM = 0.
        IF(MODE) 510,510,506
 506    DO 509 I=1,NZL
        IF(ABS(ZR(I)-ZR(NBI))+ABS(ZI(I)-ZI(NBI))) 507,507,509
 507    XM = XM + XMZ(I)
```

```
      IF(I-NBI) 508,509,509
  508 XJ = XM
  509 CONTINUE
      GO TO 515
  510 DO 514 I=1,NPL
      IF(ABS(PR(I)-PR(NBI))+ABS(PI(I)-PI(NBI))) 511,511,514
  511 XM = XM + XMP(I)
      IF(I-NBI) 512,514,514
  512 XJ =XM
  514 CONTINUE
  515 IF(MODE) 518,518,516
  516 RLX(L) = ZR(NBI)
      RLY(L) = ZI(NBI)
      MP = XMZ(NBI)
      ISGNL(L) = 4
      GO TO 520
  518 RLX(L) = PR(NBI)
      RLY(L) = PI(NBI)
      MP = XMP(NBI)
      ISGNL(L) = 5
  520 IF(RLX(L)-BNDY(1)) 556,522,522
  522 IF(RLX(L)-BNDY(2)) 524,524,556
  524 IF(RLY(L)-BNDY(3)) 556,526,526
  526 IF(RLY(L)-BNDY(4)) 530,530,556
  530 CALL RTLOC(10+MODE,0,RLX(L),RLY(L),E,0.,E,E,M,E,ISW,NT,D,D,D,D)
      IF(ISW-2) 540,546,547
  540 CALL RTLOC(5,0,RLX(L),RLY(L),E,E,E,E,M,SANG,M,M,D,C,D,D)
      IF(MODE) 544,544,542
  542 ANG(L) = (-SANG + TWOPI*XJ)/XM
      GO TO 550
  544 ANG(L) = (SANG + TWOPI*XJ)/XM
      GO TO 550
  546 APX(L) = ZR(NT)
      APY(L) = ZI(NT)
      GO TO 548
  547 APX(L) = PR(NT)
      APY(L) = PI(NT)
  548 FLAG(L) =ISW
  550 L = L + 1
      K1 = K1 + 1
      IF(IOSW) 552,554,552
  552 NC = 1
      GO TO 575
  554 IF(L-3) 556,556,570
  556 IF(K-NB) 805,560,560
  560 IF(L-1) 850,850,565
  565 NC = L - 1
      GO TO 575
  570 NC = 3
  575 L = 1
C     BRANCH CALCULATION ROUTINE (3 BRANCHES)
  580 NPOINT = 1
```

```
      GO TO 992
  581 NW = NC
      NPAGE = 1
      NROUT = 0
      NFP = 1
      NGFMT = 1
      NGFMTL = 1
      DO 600 I=1,NC
      IF(MOUT) 582,582,584
  582 IOUT(I) = 1
      GO TO 590
  584 IOUT(I) = MOUT
      IF(MGAIN) 586,586,588
  586 IGAIN(I)= 1
      GO TO 590
  588 IGAIN(I) = MGAIN
  590 IF(MPLOT) 592,592,594
  592 IPLOT(I) = 1
      GO TO 596
  594 IPLOT(I) = MPLOT
  596 IF(MPLOTC) 597,597,598
  597 IPLOTC(I) = 1
      GO TO 600
  598 IPLOTC(I) = MPLOTC
  600 IBLK(I) = -1
      IF(MPLOT) 614,614,604
  604 IF(IABS(MSYMX)-2) 614,606,606
  606 DO 610 I=1,NC
      KB =KB+1
      IF(KB-10) 610,610,608
  608 KB = 1
  610 NSYMB(I) = KB
  614 IF(MOUT) 6247,6247,615
C     WRITE BRANCH HEADING
  615 KM = K1 + 1-NC
      NLINE = 1
      CALL CLASS(INDCLS)
      WRITE(6,80)TITLE
      WRITE(6,11) NRUN
      IF(NROUT) 616,616,6151
 6151 NPAGE = NPAGE + 1
      WRITE(IOW,63) NPAGE
  616 IF(IOSW) 624,617,624
  617 GO TO (622,620,618),NC
  618 WRITE(IOW,12) (I,I=KM,K1)
      GO TO 6247
  620 WRITE(IOW,14) (I,I=KM,K1)
      GO TO 6247
  622 WRITE(IOW,15) K1
      GO TO 6247
  624 WRITE(IOW,40) K1
      IF(IT(1)) 6242,6242,6241
```

```
6241 WRITE(IOW,43)
6242 IF(IT(2)) 6244,6244,6243
6243 WRITE(IOW,42)
6244 IF(IT(3)) 6246,6246,6245
6245 WRITE(IOW,41)
6246 WRITE(IOW,44)
6247 IF(NROUT) 625,625,716
625  I=1
630  IF(IBLK(I)) 631,700,632
631  IBLK(I) = 1
     NSGNL = ISGNL(I)
     IF(IOSW) 6311,635,6312
6311 XD =RLX(1)
     YD =RLY(1)
     SANGD = 0.
     ANGD = ANG(1)
     GO TO 6313
6312 SANG = 0.
     ANGLE = ANG(1)
6313 NIA = 1
     NID = 0
     NSTEP = 1
     GO TO 635
632  IF(IOFLAG(I)) 633,633,700
C    ROOT LOCUS CALCULATION ROUTINE
633  NSGNL = 1
     NSTEP = 1
     IF(FLAG(I)) 1015,1020,1000
1000 IF(FLAG(I)-10.) 1006,1002,1006
1002 IF(IOSW) 1003,1008,1004
1003 XD = XT
     YD = YT
     ANGD = AT
     SANGD = D
     GO TO 1008
1004 ANGLE = ANG(I)
     SANG = E
     GO TO 1008
1006 NSGNL = FLAG(I) + 2
1008 FLAG(I)=0.
     IF(NSGNL-6) 1010,1999,1999
1010 RLX(I) = APX(I)
     RLY(I) = APY(I)
     IF(NSGNL-3) 1999,1999,1012
1012 XD = RLX(I)
     YD = RLY(I)
     GO TO 1999
1015 STEP = -FLAG(I)
     FLAG(I) = 0.
     X = APX(I)
     Y = APY(I)
     GO TO 1025
```

```
 1020 STEP = DELT
      X = RLX(I)
      Y = RLY(I)
 1025 ANGLE = ANG(I)
      CALL RTLOC(NLC,0,X,Y,ANGLE,STEP,TOL,ACHG,15,SANG,ISW,NS,XD,YD,
     1           ANGD,SANGD)
      GO TO (1100,1055,1050,1028,1028),ISW
 1028 XD= X
      YD = Y
      ANGD = ANGLE
      IF(NPREC) 1030,1030,1200
 1030 GO TO (1100,1055,1050,1250,1065),ISW
 1050 APX(I) = PR(NS)
      APY(I) = PI(NS)
      GO TO 1060
 1055 APX(I) = ZR(NS)
      APY(I) = ZI(NS)
 1060 FLAG(I) = ISW
      GO TO 1130
 1065 NSGNL = 6
      GO TO 1999
C     BOUNDARY CHECK
 1100 IF(X-BNDY(1)) 1120,1120,1105
 1105 IF(X-BNDY(2)) 1110,1120,1120
 1110 IF(Y-BNDY(3)) 1120,1120,1115
 1115 IF(Y-BNDY(4)) 1125,1120,1120
 1120 NSGNL = 3
      GO TO 1130
 1125 ANG(I) = ANGLE
 1130 RLX(I) = X
      RLY(I) = Y
      GO TO 1999
C     ADAPTIVE LOOP (STEP SIZE ADJUSTMENT IN CRITICAL AREAS)
 1200 CALL RTLOC(4,-1,E,E,E,0.,E,E,M,E,ISW,NS,XD,YD,D,ANGD)
      STEP = STEP/4
      JR = 1
 1205 KR = 1
 1210 NSTEP = NSTEP + 1
      CALL RTLOC(2,-1,E,E,E,STEP,TOLANG,ACHG,15,SANG,ISW,NS,XD,YD,
     1           ANGD,SANGD)
      IF(ISW-1) 1211,1211,1240
 1211 IF(IT(1)) 1212,1215,1212
 1212 CALL RTLOC(NGAIN,1,SNGL(XD),SNGL(YD),E,E,E,E,M,GAIN(1),ICSW,NE(1),
     1           D,D,D,D)
      IF(ICSW) 1213,1213,1214
 1213 WRITE(IOW,54) SGNL(1),XD,YD,SANGD,ANGD,GAIN(1),NE(1),NIA,NID,NE(2)
      GO TO 1215
 1214 WRITE(IOW,39)SGNL(1),XD,YD,SANGD,ANGD,GAIN(1),NE(1),NIA,NID,NE(2)
 1215 IF(KR-4) 1220,1225,1225
 1220 KR = KR +1
      GO TO 1210
 1225 IF(JR-1) 1230,1230,1235
```

```
 1230 X = XD
      Y = YD
      ANGLE = ANGD
      GO TO 11CC
 1235 JR = JR-1
      STEP = STEP*4.
      KR = NL(JR)
      GO TO 1215
 1240 IF(JR-3) 1245,1245,1246
 1245 NL(JR) = KR
      JR = JR+1
      STEP = STEP/4.
      GO TO 1205
 1246 GO TO (110C,1248,1247,1250,1065),ISW
 1247 RLX(I) = PR(NS)
      RLY(I) = PI(NS)
      GO TO 1249
 1248 RLX(I) = ZR(NS)
      RLY(I) = ZI(NS)
 1249 NSGNL = ISW+2
      XD = RLX(I)
      YD = RLY(I)
      GO TO 1959
C     BREAKAWAY POINT LOCATION ROUTINE
 1250 BPX = XD
      BPY = YD
      IR = 0
 1251 CALL RTLOC(3,1,BPX,BPY,E,E,TOLBRK,DTOL,20,E,ISW,NS,XT,YT,D,D)
      IF(ISW) 1255,1255,1252
 1252 IF(IR) 1253,1253,1256
 1253 IR = 1
      BPX = XD+.5*STEP*COS(ANG(I))
      IF(BPY) 1254,1251,1254
 1254 BPY = YD+.5*STEP*SIN(ANG(I))
      GO TO 1251
 1255 XD = XT
      YD = YT
 1256 X = XD
      Y = YD
      IF(ABS(Y)-TOLBRK) 1258,1258,1260
 1258 Y = 0.
 1260 BPX = X
      BPY = Y
      SANGD = C.
      SANG = 0.
      ANGT = ANGD
C     FIND TAKEOFF ANGLE FROM BREAKAWAY POINT
 1275 IR = 0
      NSGNL = 2
      STEP = DELTR
 1280 FLAG(I) = STEP-DELT
      DO 1290 NR=1,1C
```

```
      TANG = ANGT - DANG(NR)
      CALL RTLOC(4,1,X+STEP*COS(TANG),Y+STEP*SIN(TANG),TANG,STEP,E,1.,
     1            M,E,ISW,NS,XT,YT,AT,D)
      GO TO (1282,1C55,1C5C,129C,1290),ISW
 1282 ANGLE = TANG
      ANGD = TANG
 1285 CALL RTLOC(2,1,BPX,BPY,TANG,STEP,TOLANG,BCHG,15,E,ISW,NS,XT,YT,
     1            AT,D)
      GO TO (13CC,1C55,1C5C,129C,1290),ISW
 1290 CONTINUE
      IF(IR) 1294,1294,1298
 1294 STEP = DELT/1C.
      IR = 1
      GO TO 1280
 1298 FLAG(I) = 4
      GO TO 1130
 1300 APX(I) = BPX
      APY(I)=BPY
      ANG(I) = TANG
 1310 IF(FLAG(I)) 1130,1320,1130
 1320 FLAG(I) = 1C.
      GO TO 113C
 1999 IF(NSGNL-3) 635,634,634
 634  IBLK(I) = C
      ICHK = 1
 635  SIGNAL(I) = SGNL(NSGNL)
      IF(IPLOT(I)-MPLOT) 636,638,640
 636  IPLOT(I) = IPLOT(I) + 1
      GO TO 64C
638   IPLOT(I) = 1
      CALL RLPLOT(1,ISWX,RLX(I),RLY(I),NSYMB(I),I,SC,NSTOR,KSTOR,NPX,
     .NOF,NRUN,MARKT,NXINT,1,MSYMX,MGRID,MSCALE,TITLE)
640   IF(IPLOTC(I)-MPLOTC) 642,644,660
 642  IPLOTC(I) = IPLOTC(I) + 1
      GO TO 66C
 644  IPLOTC(I) = 1
      CALL RLPLOT(1,ISWXC,RLX(I),RLY(I),NSYM,I,SCC,NSTORC,KSTORC,NPXC,
     .     NOFC,NRUN,MARKTC,NXINTC,2,MSYMXC,MGRIDC,MSCALC,TITLE)
 660  IF(IOUT(I)-MOUT) 662,664,7CC
 662  IOUT(I) = IOUT(I) + 1
      GO TO 67C
 664  IOUT(I) = 1
      IOFLAG(I) = 1
      IF(IGAIN(I)-MGAIN) 666,668,670
 666  IGAIN(I) = IGAIN(I) + 1
      GO TO 67C
 668  IGAIN(I) = 1
      IGFLAG(I) = 1
 670  GO TO (68C,672,672,674,672,674),NSGNL
 672  IGFLAG(I) = 1
      GO TO 675
 674  IGFLAG(I) = 0
```

```
675   IOFLAG(I) = 1
680   IF(IGFLAG(I)) 700,700,685
685   CALL RTLOC(NGAIN,I,RLX(I),RLY(I),E,E,E,E,M,GAIN(I),IOSW,NE(I),
     1            D,D,D,D)
      IF(NE(I)) 686,687,686
686   IGFLAG(I) = 0
      IGFL(I) = 1
687   IF(KGAIN) 688,7001,7001
688   GAIN(I) = -GAIN(I)
7001  IF(GNFCTR.LT.0.0) GAIN(I)=-GAIN(I)
700   IF(I-NC) 705,710,710
705   I = I + 1
      GO TO 630
710   IF(MOUT) 760,760,715
715   IF(IOFLAG(1)+IOFLAG(2)+IOFLAG(3)-NW) 625,716,716
716   NLINE = NLINE + 1
      IF(NLINE-50) 7162,7162,7161
7161  NROUT = 1
      GO TO 615
7162  IF(IOSW) 776,717,773
717   NGN = IGFLAG(1) + IGFLAG(2) +IGFLAG(3)
      NGR = NGN+1
      IF(NGN-NW) 718,729,729
718   NFMT = NFP + 4
      IF(NGN) 725,725,720
720   IF(IGFLAG(1)) 721,721,724
721   NGFMT = 4 + IGFLAG(2)
      GO TO 725
724   NGFMT = 3-IGFLAG(2)
725   NGL = IGFL(1)+IGFL(2)+IGFL(3)
      NGLR = NGL+1
      IF(NGL) 730,730,726
726   IF(IGFL(1)) 728,728,727
727   NGFMTL = 3-IGFL(2)
      GO TO 730
728   NGFMTL = 4 + IGFL(2)
      GO TO 730
729   NFMT = NFP
730   GO TO (731,732,734,733,735,736,738,737),NFMT
731   WRITE(IOW,20) (SIGNAL(I),RLX(I),RLY(I),GAIN(I),I=1,NW)
      GO TO 752
732   WRITE(IOW,21) (SIGNAL(I),RLX(I),RLY(I),GAIN(I),I=1,NWR,2)
      GO TO 752
733   WRITE(IOW,22) (SIGNAL(I),RLX(I),RLY(I),GAIN(I),I=2,NWR)
      GO TO 752
734   WRITE(IOW,23) SIGNAL(3),RLX(3),RLY(3),GAIN(3)
      GO TO 752
735   WRITE(IOW,24) (SIGNAL(I),RLX(I),RLY(I),I=1,NW)
      GO TO 739
736   WRITE(IOW,25) (SIGNAL(I),RLX(I),RLY(I),I=1,NWR,2)
      GO TO 739
737   WRITE(IOW,26) (SIGNAL(I),RLX(I),RLY(I),I=2,NWR)
```

```
      GO TO 739
738   WRITE(IOW,27) SIGNAL(3),RLX(3),RLY(3)
739   GO TO (745,740,741,743,742),NGFMT
740   WRITE(IOW,28) (GAIN(I),I=1,NGN)
      GO TO 744
741   WRITE(IOW,29) (GAIN(I),I=1,NGR,2)
      GO TO 744
742   WRITE(IOW,30) (GAIN(I),I=2,NGR)
      GO TO 744
743   WRITE(IOW,31) GAIN(3)
744   NGFMT = 1
745   GO TO (752,746,747,749,748),NGFMTL
746   WRITE(IOW,34) (GAIN(I),NE(I),I=1,NGL)
      GO TO 750
747   WRITE(IOW,35) (GAIN(I),NE(I),I=1,NGLR,2)
      GO TO 750
748   WRITE(IOW,36) (GAIN(I),NE(I),I=2,NGLR)
      GO TO 750
749   WRITE(IOW,37) GAIN(3),NE(3)
750   NGFMTL = 1
752   DO 754 I=1,3
      IOFLAG(I)= 0
      IGFLAG(I)= 0
754   IGFL(I) = 0
760   IF(ICHK) 625,625,770
770   IF(IBLK(1)) 772,772,780
772   NFP=3+IBLK(2)
      GO TO 785
773   IF(IGFLAG(1)+IGFL(1)) 775,775,774
774   WRITE(IOW,39) SIGNAL(1),RLX(1),RLY(1),SANG,ANGLE,GAIN(1),NE(1),
     1                NIA,NID,NSTEP
      GO TO 752
775   WRITE(IOW,73) SIGNAL(1),RLX(1),RLY(1),SANG,ANGLE,NIA,NID,NSTEP
      GO TO 752
776   IF(IGFLAG(1)+IGFL(1)) 778,778,777
777   WRITE(IOW,54) SIGNAL(1),XD,YD,SANGD,ANGD,GAIN(1),NE(1),NIA,NID,
     1                NSTEP
      GO TO 752
778   WRITE(IOW,74) SIGNAL(1),XD,YD,SANGD,ANGD,NIA,NID,NSTEP
      GO TO 752
780   NFP = 2 - IBLK(2)
785   NW = IBLK(1) + IBLK(2) + IBLK(3)
      NWR = NW + 1
      ICHK = 0
      IF(NW) 790,790,625
790   IF(ISWX-3) 794,794,792
792   ISWX = 3
794   IF(ISWXC-3) 800,800,796
796   ISWXC = 3
800   IF(K-NB) 805,850,850
805   K = K+1
      IF(J-MP) 810,815,815
```

```
  810 J=J+1
      XJ = XJ +1.
      GO TO 515
  815 NBI = NBI + 1
      GO TO 500
C     FINAL ROUTINE
  850 NPOINT =2
      GO TO 992
  852 IF(ICROSS) 900,900,854
  854 WRITE(6,59) NRUN
      RL2 = BNDY(3)
      DELTA(2) = (BNDY(4)-BNDY(3))/50
      RL(1) = RL2-DELTA(2)
      I = 0
  856 RL1 = RL2
      CALL RTLOC(5,0,0.,RL1,E,E,E,E,M,SANGL,M,M,D,D,D,D)
      IF(SANGL) 858,865,858
  858 RL2 = RL2 + DELTA(2)
      RL1 = RL2
      CALL RTLOC(5,0,0.,RL2,E,E,E,E,M,SANG,M,M,D,D,D,D)
      IF(SANG*SANGL) 865,865,860
  860 SANGL = SANG
      IF(RL2-BNDY(4)) 858,890,890
  865 CALL RTLOC(9,0,0.,RL1,E,DELTR,1.E-12,DELTA(2),15,E,ISW,M,D,YD,D,D)
      IF(ISW) 868,868,860
  868 IF(ABS(RL1-RL(1))-TOL) 860,860,870
  870 I = I+1
      RL(1) = RL1
      CALL RTLOC(NGAIN,1,0.,RL1,E,E,E,E,M,GAIN(1),0,NE(1),D,D,D,D)
  874 IF(KGAIN) 875,876,876
  875 GAIN(1) = -GAIN(1)
  876 IF(GNFCTR.LT.0.0) GAIN(I)=-GAIN(I)
      IF(IOSW)878,880,880
  878 WRITE(IOW,60) YD
      GO TO 882
  880 WRITE(IOW,61) RL1
  882 IF(NE(1)) 884,886,884
  884 WRITE(IOW,71) GAIN(1),NE(1)
      GO TO 856
  886 WRITE(IOW,72) GAIN(1)
      GO TO 856
  890 IF(I) 892,892,894
  892 WRITE(IOW,62) BNDY(3),BNDY(4)
  894 NPOINT  = 3
      GO TO 992
  900 IPLSW = 1
  902 IF(MPLOT) 950,950,905
  905 I = 1
      KSW = 1
  910 IF(I-NPL) 915,915,920
  915 RL1 = PR(I)
      RL2 = PI(I)
```

```
      ISYM = 31
      GO TO 930
920   IZ = I-NPL
      IF(IZ-NZL) 925,925,935
925   RL1 = ZR(IZ)
      RL2 = ZI(IZ)
      ISYM = 32
930   IF(IPLSW-1) 975,975,980
935   KSW = 2
940   GO TO (945,948,960,986),IPLSW
945   IPLSW = 2
947   GO TO 975
948   ISWX = 1
      NPOINT = 4
      GO TO 992
950   IF(MPLOTC.LE.0)GO TO 100
955   IPLSW = 3
      GO TO 905
960   IF(KPLSW) 970,965,970
965   NSYM = NSYM + 1
      IF(NSYM-10) 989,989,970
970   IPLSW = 4
      KPLOTC = 0
      GO TO 980
975   CALL RLPLOT(KSW,ISWX,RL1,RL2,ISYM,1,SC,NSTOR,KSTOR,NPX,NOF,
     .NRUN,0,NXINT,1,C,MGRID,MSCALE,TITLE)
      GO TO 985
980   CALL RLPLOT(KSW,ISWXC,RL1,RL2,ISYM,1,SCC,NSTORC,KSTORC,NPXC,
     .   NOFC,NRUN,C,NXINTC,2,C,MGRIDC,MSCALC,TITLE)
985   I = I+1
      GO TO 910
986   NPOINT = 5
      GO TO 992
987   IF(IABS(KPLSW)-2) 988,989,989
988   ISWXC = 1
      NSYM = 1
      GO TO 990
989   ISWXC = 2
990   KPLSW = 0
      GO TO 101
C     TIMER ROUTINE
992   IF(ITSW) 993,998,998
993   CONTINUE
      NTIME=100.0
      T= NTIME/100.
      TIME = T-T0
      T0 = T
      TRUN = TRUN + TIME
      IF(NPOINT-10) 997,994,994
994   IF(ITSW) 995,996,995
995   WRITE(IOW,32) NRUN,TRUN
      WRITE(IOW,33) T
```

```
      TRUN = 0
      GO TO 101
997   WRITE(IOW,17) NPOINT,TIME
998   GO TO (581,852,900,950,987,311,380,147,167),NPOINT
      END
```

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: AFFDL-TR-73-142, Volume II / 2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): Design Methods for Specifying Handling Qualities for Control Configured Vehicles Volume II - McPilot Program User's Manual	5. TYPE OF REPORT & PERIOD COVERED: Technical Report 15 Jan 73 - 15 Nov 73 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Donald W. Vogt	8. CONTRACT OR GRANT NUMBER(s): F33615-73-C-3064
9. PERFORMING ORGANIZATION NAME AND ADDRESS: McDonnell Aircraft Company P.O. Box 516 St. Louis, Missouri 63166	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: Air Force Flight Dynamics Laboratory AFFDL/FGL/(FY1456) Wright-Patterson Air Force Base, Ohio 45433	12. REPORT DATE: 15 November 1973 13. NUMBER OF PAGES: 242
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)	15. SECURITY CLASS. (of this report): Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

CCV Handling Qualities Analysis
Control Configured Vehicles
Pilot Ratings
Pilot Models
Pilot Vehicle Analysis
Dynamic Handling Qualities

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A method for predicting aircraft pilot ratings which is applicable to control configured vehicles (CCV's) is developed based on aircraft performance and pilot workload. A discussion of this method of handling quality analysis, and a description of four random tracking pilot rating correlations developed using the method are presented in Volume I of the report. A literature survey classifying the most recent reports applicable to flying qualities is tabulated. A user's manual for McPilot, a general purpose digital computer

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

program developed to implement the method and correlate pilot rating data, is presented in Volume II. The correlations that were developed are used to evaluate the pilot ratings of two conventional contemporary fighter aircraft configurations. The correlations are also applied to two CCV fighter configurations for which limited amounts of handling quality data had been obtained through manned aircraft simulations.

☆U.S.Government Printing Office: 1974 — 758-434/544