

AD-A053 446

AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/G 12/1
INTERAC - AN INTERACTIVE SOFTWARE PACKAGE FOR DIRECT DIGITAL CO--ETC(U)
DEC 77 J A COLGATE

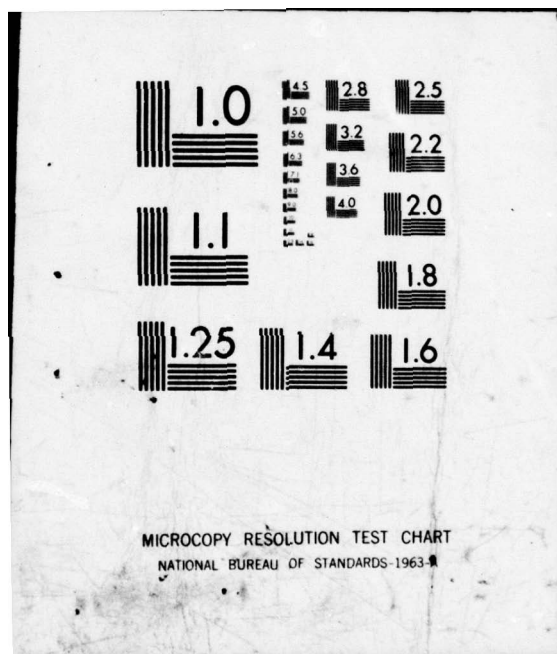
UNCLASSIFIED

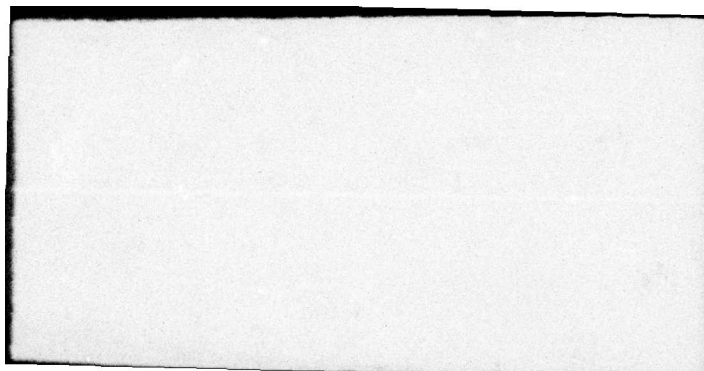
AFIT/66C/EE/77D-1

NL

1 OF 2
AD
A053446





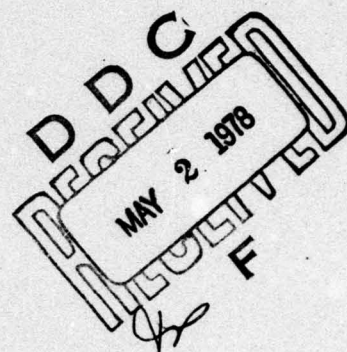


AFIT/GGC/EE/77D-1 ✓

(1)

AD A 053446

AD No.
DDC FILE COPY



INTERAC - AN INTERACTIVE SOFTWARE PACKAGE ✓
FOR DIRECT DIGITAL CONTROL DESIGN

THESIS

AFIT/GGC/EE/77D-1 ✓

James A. Colgate
Capt USAF

**INTERAC - AN INTERACTIVE SOFTWARE PACKAGE
FOR DIRECT DIGITAL CONTROL DESIGN**

THESIS

**Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science**

by

James A. Colgate, B.S.E.E.

Capt

USAF

Graduate Guidance and Control

December 1977

ACCESSION for	
NTIS	Wide Section ☒
DDC	Buff Section ☐
UNANNOUNCED	☐
JUSTIFICATION _____	
BY _____	
DISTRIBUTION/AVAILABILITY CODES	
Dist.	SP CIAL
	

Approved for public release; distribution unlimited

Preface

This report is the result of my attempt to develop an interactive computer program that a user can use to get exactly what he needs (no more and no less) with the minimum amount of work and expertise in the program operation. The program that I started with is FORTRAC, which is a package of the latest state-of-the-art algorithms for direct digital control system design. Through this effort, I have gained a healthy respect for the computer programmers who can produce a simple easy to use program for solving complex problems.

I wish to express my gratitude to Professor Brian Porter from the University of Salford, England, for developing the FORTRAC package and to Dr. Constantine H. Houpis for suggesting the use of this package for a control system design. I also wish to thank Mr. Duane Robertus of the Flight Dynamics Lab for sponsoring this thesis.

Finally, I wish to thank my wife, Julie, for her constant encouragement while I worked on this project.

James A. Colgate

Contents

	<u>Page</u>
Preface	ii
List of Figures	iv
List of Tables.	v
Abstract.	vi
I. Introduction.	1
The Problem.	2
Scope and Objectives	2
Development.	3
II. FORTRAC Computer Package.	4
Sampled Data Transformation.	6
Augmentation	8
Control Law Synthesis.	9
Matrix Manipulation Routines	10
III. Specialized INTERAC Software.	12
Input Subroutines.	13
Output Subroutines	23
Program Sequence Control	24
IV. C-141 and F-4E Mathematical Modeling.	26
C-141 Model.	26
F-4E Model	29
Results of F-4E Control Law Synthesis.	32
V. Conclusions and Recommendations	34
Conclusions.	34
Recommendations.	34
Bibliography.	36
Appendix A: Listing of Specialized INTERAC Software.	37
Appendix B: INTERAC User's Guide	83
Appendix C: Programmer's Guide for Using the Specialized INTERAC Software	98
Appendix D: Card Sequence Required to Utilize FORTRAC.	110

List of Figures

<u>Figure</u>		<u>Page</u>
1	Continuous Time System	5
2	READNUM Flow Chart, Part I	15
3	READNUM Flow Chart, Part II.	16
4	READNUM Flow Chart, Part III	17
5	READNUM Flow Chart, Part IV.	18
6	READNUM Flow Chart, Part V	19
7	READCOM Flow Chart	22

List of Tables

<u>Table</u>	<u>Page</u>
I FORTRAC Subroutines	6
II Matrix Manipulation Routines.	11
III READNUM Legal Entry Table	14
IV Legal Commands For The INTERAC Package.	24
V Option Catagories	25
VI C-141 Longitudinal Dimensional Derivatives.	27
VII F-4E Longitudinal Non-dimensional Derivatives	30
VIII Simulation Data for the F-4E with a Discrete-Time Flight Path Angle Tracking Control System.	32

Abstract

The purpose of this investigation is to develop an interactive, user oriented software package for direct digital control system design. The batch mode program, FORTRAC, developed by Professor Brian Porter of the University of Salford, England, is the source of the computational algorithms used. This report details the input, output, and program sequence control software developed to produce an efficient, user oriented interactive package.

The package is very forgiving of user errors and gives the user complete control over what data to input, what data will be output, and which parts of the program are executed. The package is quite useful in the design process for discrete-time time-optimal control systems, where many possible control parameter variations must be examined.

INTERAC - AN INTERACTIVE SOFTWARE PACKAGE
FOR DIRECT DIGITAL CONTROL DESIGN

I. Introduction

In the past few years the digital computer has become a valuable asset in the control system design process. With increased interest in modern control theories, the computer is fast becoming a necessity. A portion of a computer program for control system design is going to be the coding dealing with input, output, and program sequence control.

Many computer programs in classical and a few in modern control theory now exist (Ref 1,2,3,4, and 5), but most can be quite cumbersome for the user. These programs are written in terms of computational and computer resource efficiency, often at the expense of user efficiency. User efficiency is defined as the capability of a user to get exactly what he needs from a program execution (no more and no less) with the minimum amount of work and expertise in the program operation.

There is no unique solution to a given control problem, and thus an iterative procedure with one or more design parameters may be required. This can best be achieved in an interactive environment where the user can make selective changes in the design parameters based upon intermediate outputs. The problem then is to develop an interactive, user oriented, program for the I-O (input-output) and program sequence control functions which can be put around any set of control system design theory algorithms.

The Problem

Professor Brian Porter of the University of Salford, England, has been developing a control system design theory based on discrete state variable feedback to achieve a time optimal response at any given sampling rate. The algorithms for this design process have been furnished by Professor Porter and the Air Force Flight Dynamics Lab in the computer program FORTRAC. This batch mode program which was written for the University of Salford computer and adapted to run on the Wright-Patterson AFB computer system along with the SCOPE operating system can be cumbersome for the user.

The problem focused on in this investigation is to develop the interactive I-O and program sequence control software for the design algorithms of FORTRAC which will give a user oriented design package that will operate efficiently on the CDC-6600/Cyber-74 computer system at Wright-Patterson AFB, Ohio. The capability of this package, INTERAC, is then demonstrated by synthesizing control laws for a fly-by-wire system for the longitudinal axis of the C-141 and F-4E aircraft.

Scope and Objectives

The software was designed to meet the following characteristics:

- (1) Input timing and format will be under user control.
- (2) Program termination will not be caused by any format errors in an input sequence. In case of such an error, the user will be notified and be asked to reenter that item.
- (3) Output content and format will be user defined.
- (4) Only user desired elements of the program need be executed and in user defined order.
- (5) Program directions will be printed for the beginning user or they can be suppressed for the experienced user.

(6) Additional algorithms can be added to the package with a minimum of additional programming. (7) The I-O subroutines can be used directly in any program requiring a user oriented, interactive environment. (8) The package will operate within the 60,000 word field length available to the interactive users on the CDC computers at Wright-Patterson AFB.

The control laws for the C-141 and F-4E fly-by-wire systems are obtained for the longitudinal axis and at only one flight condition.

Development

Chapter 2 gives an overview of the FORTRAC package with specific emphasis on the algorithms incorporated into INTERAC. The specialized INTERAC software is discussed in chapter 3. The mathematical models for the C-141 and the F-4E are developed in Chapter 4. The chapter concludes with a discussion of the control laws obtained for the two aircraft. Chapter 5 presents the conclusions and recommendations for improving the software package and for continued control system development for the C-141 and F-4E.

The actual FORTRAN Extended source code for the specialized INTERAC software is given in Appendix A, and a user's guide for the INTERAC package is contained in Appendix B. Appendix C is a programmer's guide for using the specialized INTERAC subroutines in other applications. Card sequence for the batch mode program, FORTRAC, is shown in Appendix D.

II. FORTRAC Computer Package

FORTRAC, a software package for the design of multivariable digital control systems, (Ref 6) was developed by Professor Brian Porter and furnished under contract to the Air Force Flight Dynamics Lab. This package is a collection of 42 subroutines tied together by a master program written for the particular design problem attempted. The master program furnished to the Flight Dynamics Lab will design a discrete control law for a sixth order or smaller system, design an observer, and run a time simulation of the system for the resulting controller. The theory is good for any order system, but a larger dimensioned master program would be required for higher order systems.

Table I shows the subroutines from FORTRAC grouped into four general functional categories. Most of the subroutines listed under control law synthesis and utility are included in INTERAC, and their specific function will be discussed in this chapter. A description of the other subroutines can be obtained from Reference 6.

FORTRAC is designed to take the continuous time description of a linear system (Fig 1), and to synthesize a control law for any of the following classes of problems: (1) discrete-time time-optimal regulator, (2) discrete-time time-optimal disturbance rejector, and (3) discrete-time time-optimal tracker.

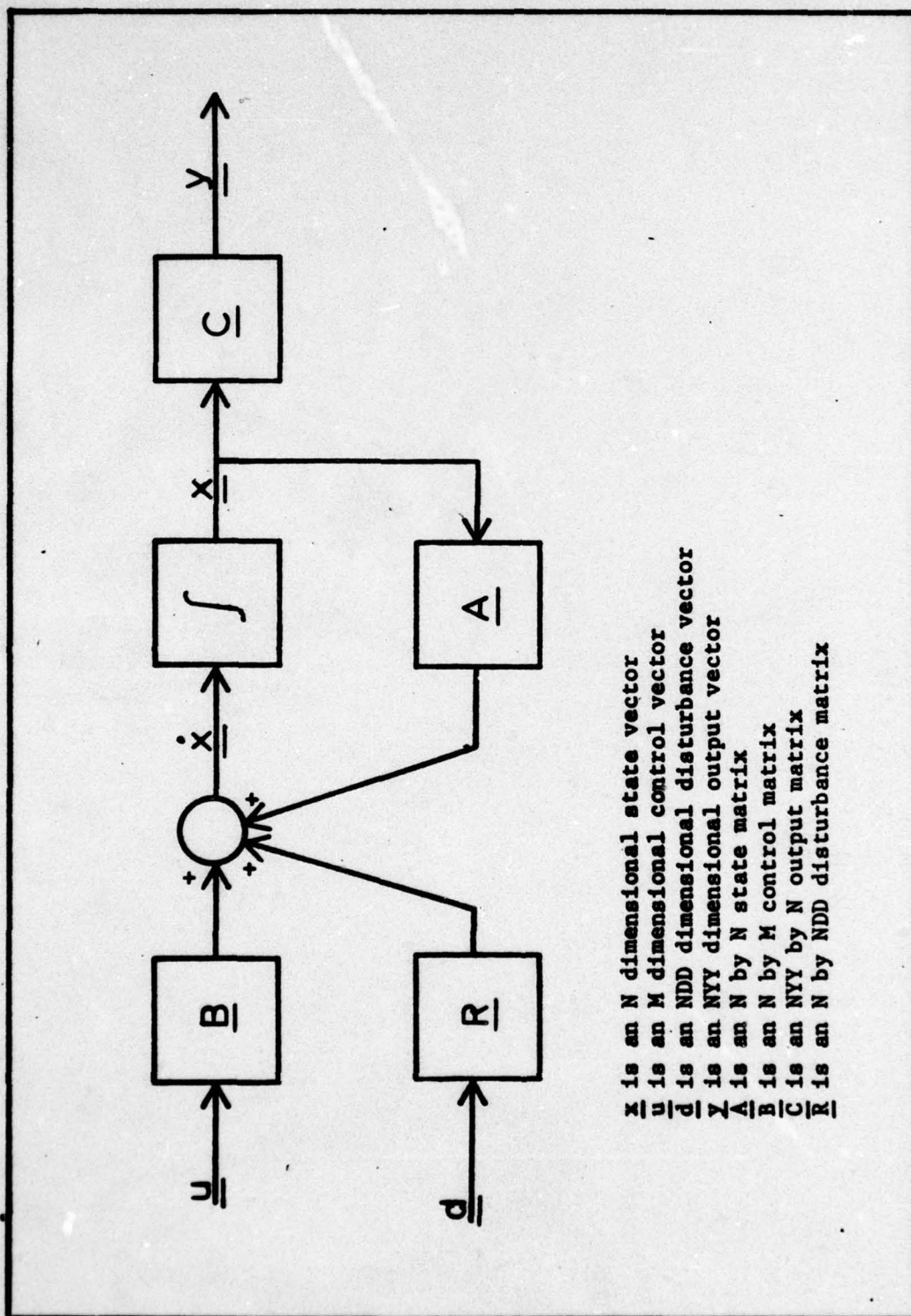


Figure 1. Continuous Time System

Table I
FORTRAC Subroutines

Control Law Synthesis	Observer Design	Simulation	Utility
RPDATA	TRANFAC	DISCLS	ZEROELE
EIGSTR	FULLOB1	*EIGNVAL	DF01CKF
EIGVAV	REDOB1	DISSIMU	MATRE10
SAMPLE	REDOB2	GENCOM	MATPR10
AUGMAT		GENDIST	SETQUP
TRANFAB		*STSTATE	UREFORM
*CONGAIN		STIMEST	*ACHOROW
		SAMSIMU	SUB1000
		*KUTAMER	SUB2000
		DIF	SUB3000
		*STCONT	*APLFORM
		STIMECO	*BRNFORM
		PRIRES	*STQDOWN
		SAMRES	CINDS
			CALCBAR
			CONLAW

* These names had to be shortened from the original names in Reference 6 in adapting the package to operate on the CDC computer system at Wright-Patterson AFB.

The type of problem to be solved and the dimensions of the input system are handled by RPDATA. The I-O of the matrix data are handled by MATRE10 and MATPR10 respectively. These functions will be handled in INTERAC by the specialized INTERAC software.

Sampled Data Transformation

The first step in any class of problem is to transform the continuous system of Figure 1, described by Eq (1), to a sampled-data system description, Eq (2).

$$\begin{aligned}\dot{\underline{x}}(t) &= \underline{A}\underline{x}(t) + \underline{B}\underline{u}(t) + \underline{R}\underline{d}(t) \\ \underline{y}(t) &= \underline{C}\underline{x}(t)\end{aligned}\tag{1}$$

$$\begin{aligned}\underline{x}(k+1) &= \underline{F}(T)\underline{x}(k) + \underline{G}(T)\underline{u}(k) + \underline{DR}(T)\underline{d}(k) \\ \underline{y}(k) &= \underline{C}\underline{x}(k)\end{aligned}\tag{2}$$

All discrete functions of k are implied to be functions of kT . Two major subroutines are used in this operation. The first is EIGSTR, which also calls EIGVAV. The eigenvalues and eigenvectors of the continuous system are determined with the aid of matrix manipulation routines from a common user mathematics library. The description of the matrix manipulation routines required for FORTRAC and INTERAC are given later in this chapter.

The second major subroutine used is SAMPLE, which generates the discrete matrices $\underline{F}(T)$, $\underline{G}(T)$, and $\underline{DR}(T)$ from $\underline{A}$, $\underline{B}$, and $\underline{R}$. The transformation can be accomplished using Eqs (3), (4), and (5).

$$\underline{F}(T) = e^{\underline{A}T}\tag{3}$$

$$\underline{G}(T) = \int_0^T e^{\underline{A}t} \underline{B} dt\tag{4}$$

$$\underline{DR}(T) = \int_0^T e^{\underline{A}t} \underline{R} dt\tag{5}$$

In general, an exact solution to the above equations cannot be easily determined. To facilitate the solution the diagonal matrix $\underline{\Lambda}$ which satisfies Eq (6) is produced,

$$\underline{A} = \underline{U}\underline{\Lambda}\underline{U}^{-1}\tag{6}$$

where $\underline{U}$ is the matrix of eigenvectors obtained from EIGSTR. At this

point the subroutine SAMPLE is limited to systems which have non-repeated eigenvalues. Multiple eigenvalues would produce off diagonal terms in $\underline{\Lambda}$, thus this simplification could not be used. The equations for the discrete transformation reduce to (Ref 6:27)

$$\underline{F}(T) = \underline{U} e^{\underline{\Delta}T} \underline{U}^{-1} \quad (7)$$

$$\underline{G}(T) = \underline{U} \left[\int_0^T e^{\underline{\Delta}t} dt \right] \underline{U}^{-1} \underline{B} \quad (8)$$

$$\underline{DR}(T) = \underline{U} \left[\int_0^T e^{\underline{\Delta}t} dt \right] \underline{U}^{-1} \underline{R} \quad (9)$$

The computation of $e^{\underline{\Delta}T}$ becomes a set of scalar problems $e^{\lambda_i T}$, where the λ_i 's are the eigenvalues of $\underline{A}$ which appear on the diagonal of $\underline{\Lambda}$. The constant matrices $\underline{U}$, $\underline{U}^{-1}$, $\underline{B}$, and $\underline{R}$ are taken outside the the integrals, and the integrals become sets of scalar integrations. The $\underline{C}$ matrix remains the same in the transformation from continuous to sampled-data form.

Augmentation

In the disturbance rejection and tracking problems, the next step is to augment the system with discrete-time integrators shown in Eq (10).

$$\underline{z}(k+1) = \underline{z}(k) + T[\underline{e}'(k)] \quad (10)$$

where the error, $\underline{e}'(k)$, for the disturbance rejector is $\underline{y}(k)$ and for

the tracker is $\underline{y}(k) - \underline{v}(k)$. The rationale and specific requirements for the use of integral feedback is given in Reference 7 (papers 7 and 9). The command input matrix, $\underline{DE} = \underline{IT}$, is established for the input command, $\underline{v}(k)$, in the augmented system equation. The augmented discrete system is then represented by Eq (11).

$$\underline{x}'(k+1) = \underline{AF}\underline{x}'(k) + \underline{AG}\underline{u}(k) + \underline{ADR}\underline{d}(k) + \underline{ADE}\underline{v}(k) \quad (11)$$

where

$$\underline{AF} = \begin{bmatrix} \underline{F} & \underline{\emptyset} \\ \underline{CT} & \underline{I} \end{bmatrix} \quad \underline{AG} = \begin{bmatrix} \underline{G} \\ \underline{\emptyset} \end{bmatrix}$$

$$\underline{ADR} = \begin{bmatrix} \underline{DR} \\ \underline{\emptyset} \end{bmatrix} \quad \underline{ADE} = \begin{bmatrix} \underline{\emptyset} \\ -\underline{DE} \end{bmatrix}$$

$$\underline{x}'(k) = \begin{bmatrix} \underline{x}(k) \\ \underline{z}(k) \end{bmatrix} \quad (12)$$

Control Law Synthesis

The control law to be synthesized is given by Eq (13).

$$\underline{u}(k) = \underline{K}\underline{x}'(k) \quad (13)$$

The first step uses subroutine TRANFAB to transform the matrices

AF and AG into Brunovsky controllable canonical form. This is done using the first method of Aplevich (Ref 8:124-126). The utility subroutines SETQUP through STQDOWN (Table I) are used to achieve the transformation and to generate the transformation matrix $\underline{T}^{-1}$ to satisfy Eq 14.

$$\underline{x}''(k) = \underline{T}^{-1} \underline{x}'(k) \quad (14)$$

The control law synthesized to meet Eq (13) is as follows.

$$\underline{u}(k) = \underline{L}'[\underline{H} + \underline{E}] \underline{T}^{-1} \underline{x}'(k) \quad (15)$$

The $\underline{H}$ and $\underline{L}$ matrices are formed from the Brunovsky controllable canonical form of AF and AG respectively (Ref 6:44-46). The $\underline{E}$ matrix is used to specify the desired closed-loop eigenvalues. For the time-optimal control law the closed-loop eigenvalues are placed at the origin in the z-plane and the $\underline{E}$ matrix is the zero matrix. Refer to Reference 6, page 45, for the algorithm used for calculating $\underline{E}$ for non-zero eigenvalues.

Matrix Manipulation Routines

Table II shows the matrix manipulation routines called in the original FORTRAC package and the corresponding routines used in the version adapted to run on the CDC computers at Wright-Patterson AFB. The eigenstructure analysis routines were taken from the EISPACK Library (Ref 9) available on the CDC computers. The other routines were written by the author and the source code is shown in Appendix A.

Table II
Matrix Manipulation Routines

Routines Called In The Original FORTRAC Package	Routines Used In The Wright-Patterson Adaptation
F01CAF F01CBF F01CCF F01CDF F01CEF F01CJF F04AEF	PRESET IDENT COPYAB ADDMAT SUBMAT TRANPOS INVERT
F01ATF F01AKF F02AGF F02APF	BALANC ELMHES ELTRAN HQR HQR2 BALBAK
Eigenstructure Routines	

Appendix C gives the syntax required to use the author's matrix manipulation routines and gives a brief description of the algorithms used.

The above routines along with DISCLS are included in the INTERAC package. The subroutine DISCLS calculates the closed-loop equations which can then be used in a discrete simulation.

III. Specialized INTERAC Software

Before adapting FORTRAC to an interactive program, several interactive programs (Ref 1,2,3,4, and 5) available on the Wright-Patterson AFB computer were examined to see how efficient they were for the user. The first item of note was that each program had different input format, output content and format, and control structure. This greatly increases the workload on the user who has to use several programs in a design effort. With the exception of OPTCON, there are no provisions for correcting an erroneous data point entry without reentering the entire data set. All programs are terminated by the SCOPE operating system for a simple input error by the user. All programs allowed the user to select the options he wished to execute, but there are no provisions to stop between options and allow the user to decide if the sequence should be continued. In general each program has some good and many bad points in regard to user efficiency.

Three methods of developing FORTRAC into an efficient, user oriented, interactive, package were considered. The first was to redefine the operating system for interactive computer use. Although this would provide the greatest improvement for all classes of interactive programs, the time and resources required would be beyond the capability of this investigation. The second method would be to utilize a higher order computer language other than FORTRAN. Of the languages available on the Wright-Patterson CDC-6600/Cyber-74 computer system, FORTRAN Extended seemed to be the best suited for the job. This left direct FORTRAN programming, possibly sacrificing

a little computer resource efficiency in order to achieve the desired user efficiency.

The INTERAC software described in this chapter was developed to meet the specifications given in Chapter 1 using the maximum capabilities of the FORTRAN Extended language (Ref 10) and the SCOPE operating system (Ref 11) available on the Wright-Patterson AFB computers. The three areas in a typical program operation cycle addressed are: (1) input, (2) output, and (3) program sequence control.

Input includes both numeric data and alphanumeric commands. The two subroutines developed for this are READNUM (read numeric) and READCOM (read command). The data output is controlled in content by an entry in a variable output table and in format by the subroutines PRINTR and LISTER. The program sequence control is based on entries in a valid command table and is achieved in this program with the subroutine COMND.

Input Subroutines

The subroutine READNUM was first conceptualized due to the deficiencies in the SCOPE operating system on the Wright-Patterson AFB computer. When a format error is made in a numeric input operation, an error message is printed and the program is terminated. This can be quite frustrating to a user who is just entering the last number of a 10 by 10 matrix. The READNUM subroutine reads all inputs as alphanumeric characters and checks for format errors. If any errors are detected the user is asked to reenter those items. The error checking is based on a legal entry table to be discussed later.

Table III
READNUM, Legal Entry Table

♦A ♦B ♦C			LEGAL ENTRY TABLE	♦A ♦B ♦C			LEGAL ENTRY TABLE
♦♦	♦♦	01	66666666666666666666				
:	00	02	11111111111153555555	5	40	34	11111111111113111112
A	01	03	11111111111131555553	6	41	35	11111111111113111112
B	02	04	11111111111151555555	7	42	36	11111111111113111112
C	03	05	11111111111131555553	8	43	37	11111111111113111112
D	04	06	11111111111151555555	9	44	38	11111111111113111112
E	05	07	11111111111151555225	+	45	39	11111111111113551552
F	06	08	11111111111151555555	-	46	40	11111111111113551552
G	07	09	11111111111151555555	♦	47	41	11111111111113555552
H	10	10	11111111111151555555	/	50	42	111111111111142455554
I	11	11	11111111111151555555	(	51	43	111111111111153555555
J	12	12	11111111111151555555	)	52	44	111111111111153555555
K	13	13	11111111111151555555	\$	53	45	111111111111135777773
L	14	14	11111111111131555553	=	54	46	111111111111153555555
M	15	15	11111111111151555555		55	47	111111111111112244441
N	16	16	11111111111151555555	,	56	48	111111111111112344441
O	17	17	11111111111151555555	.	57	49	111111111111113555532
P	20	18	11111111111151555555	#	60	50	111111111111153555555
Q	21	19	11111111111151555555	[	61	51	111111111111153555555
R	22	20	11111111111151555553	]	62	52	111111111111153555555
S	23	21	11111111111151555555	%	63	53	111111111111153555555
T	24	22	11111111111151555555	"	64	54	111111111111153555555
U	25	23	11111111111151555555	_	65	55	111111111111153555555
V	26	24	11111111111151555555	!	66	56	111111111111153555555
W	27	25	11111111111151555555	&	67	57	111111111111153555555
X	30	26	11111111111151555555	<	70	58	111111111111153555555
Y	31	27	11111111111151555555	?	71	59	111111111111153555555
Z	32	28	11111111111131555553	<	72	60	111111111111153555555
0	33	29	11111111111113111112	>	73	61	111111111111153555555
1	34	30	11111111111113111112	@	74	62	111111111111153555555
2	35	31	11111111111113111112	\	75	63	111111111111153555555
3	36	32	11111111111113111112	^	76	64	111111111111153555555
4	37	33	11111111111113111112	;	77	65	111111111111153555555

♦A -- CHARACTER
 ♦B -- OCTAL VALUE OF CHARACTER REPRESENTATION
 ♦C -- DECIMAL INDEX FOR TABLE
 ♦♦ -- THIS ENTRY IN THE TABLE PROVIDES FOR AN
 END OF LINE INDICATOR.

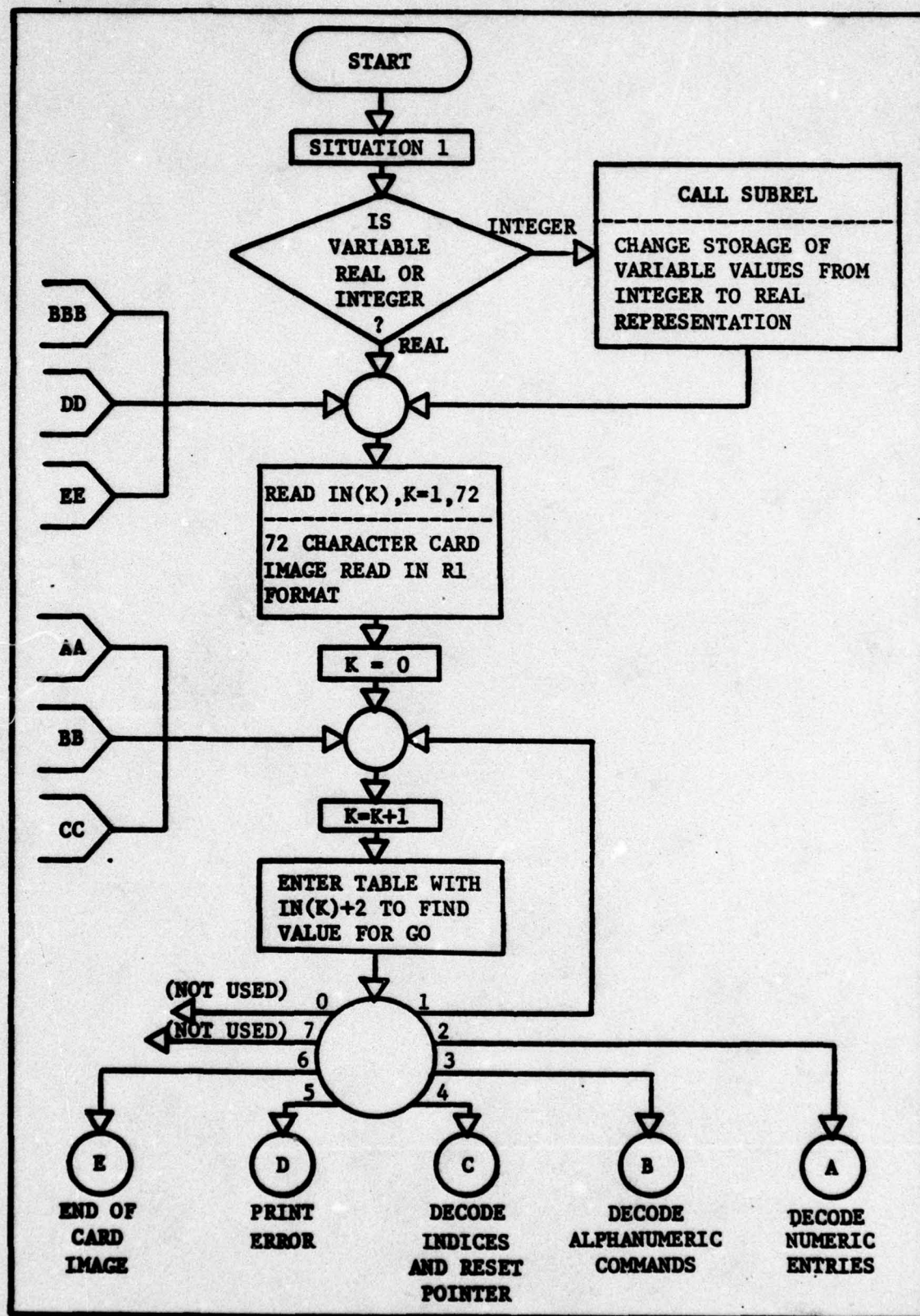
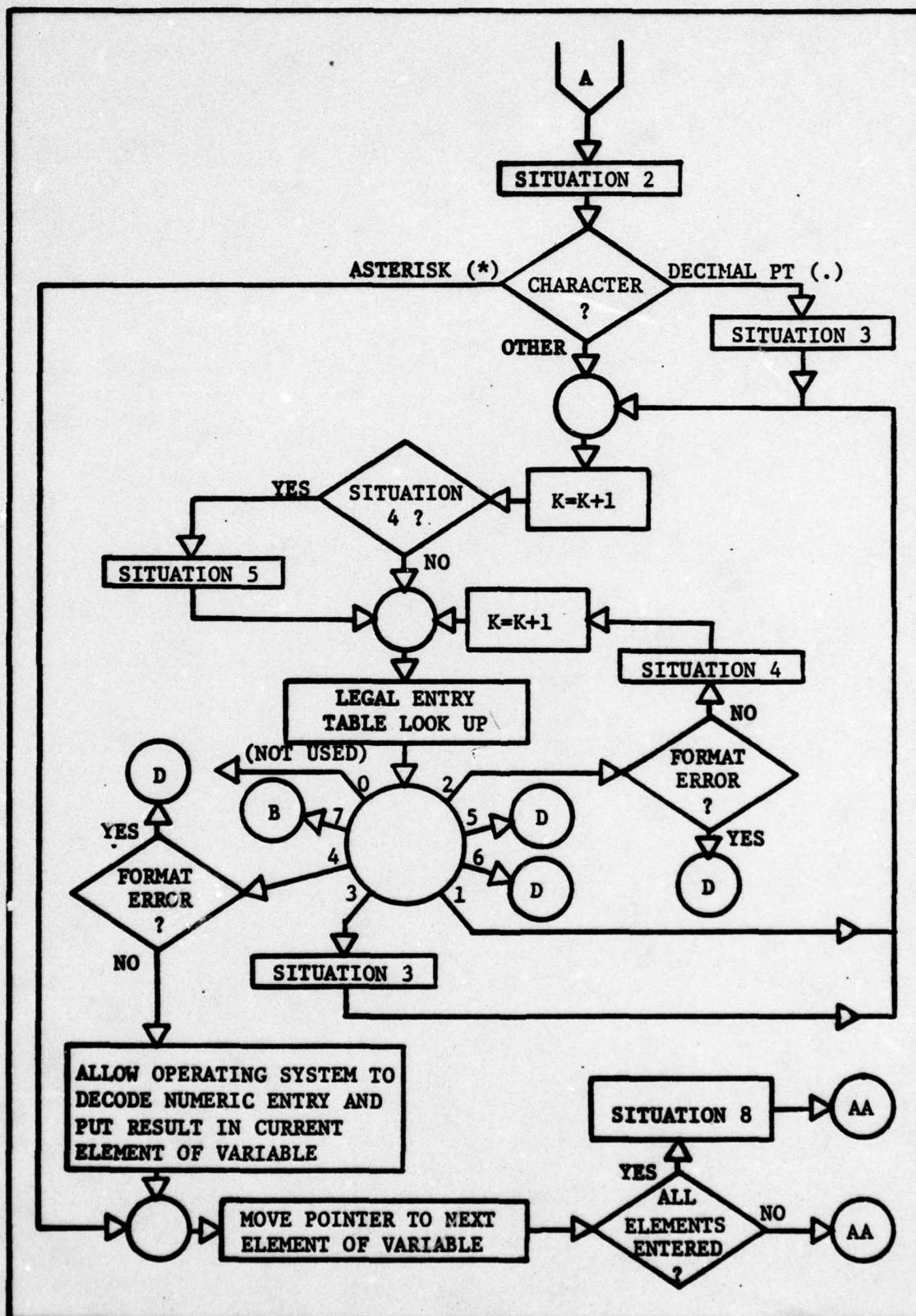
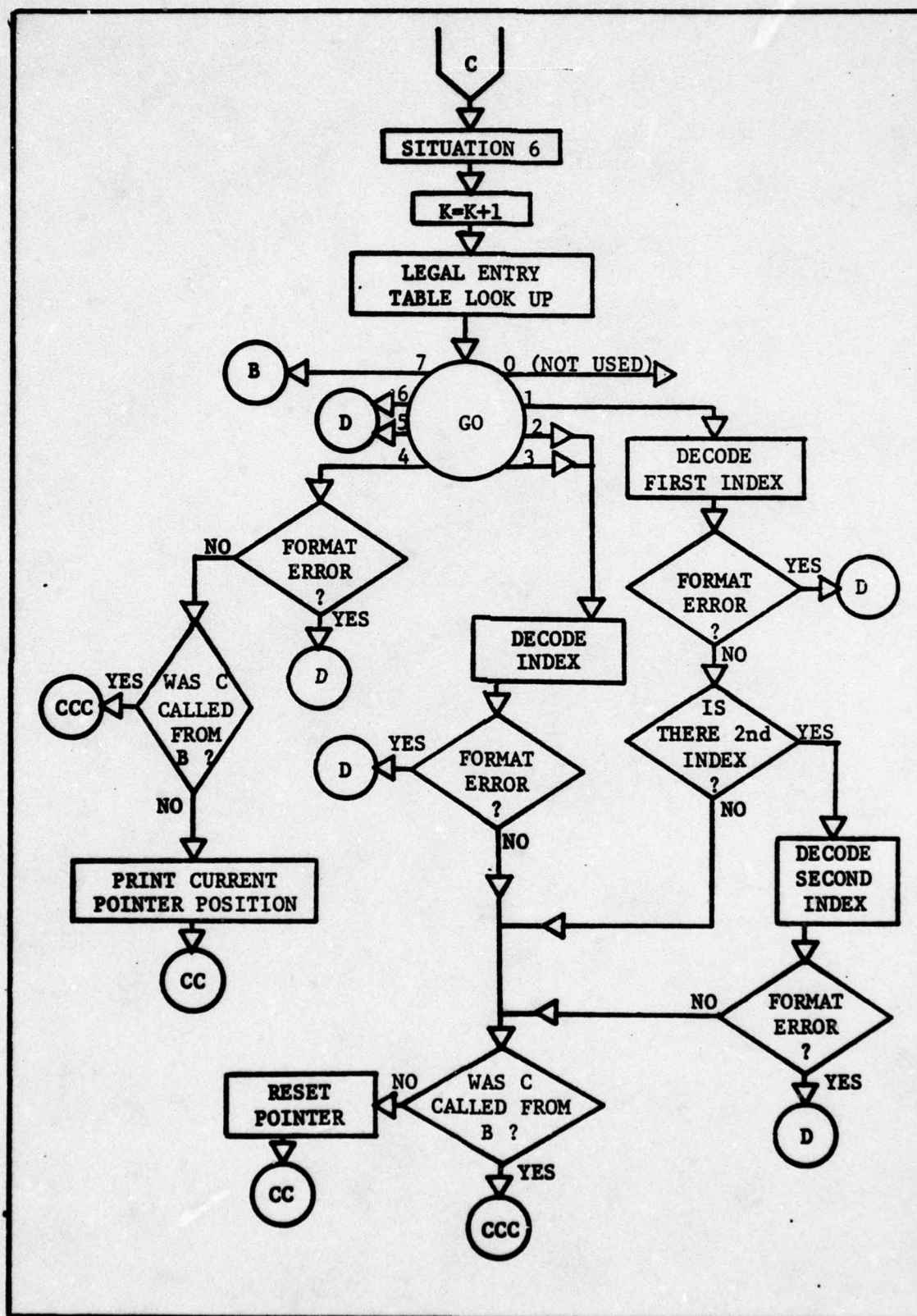


Figure 2. READNUM Flow Chart, Part I





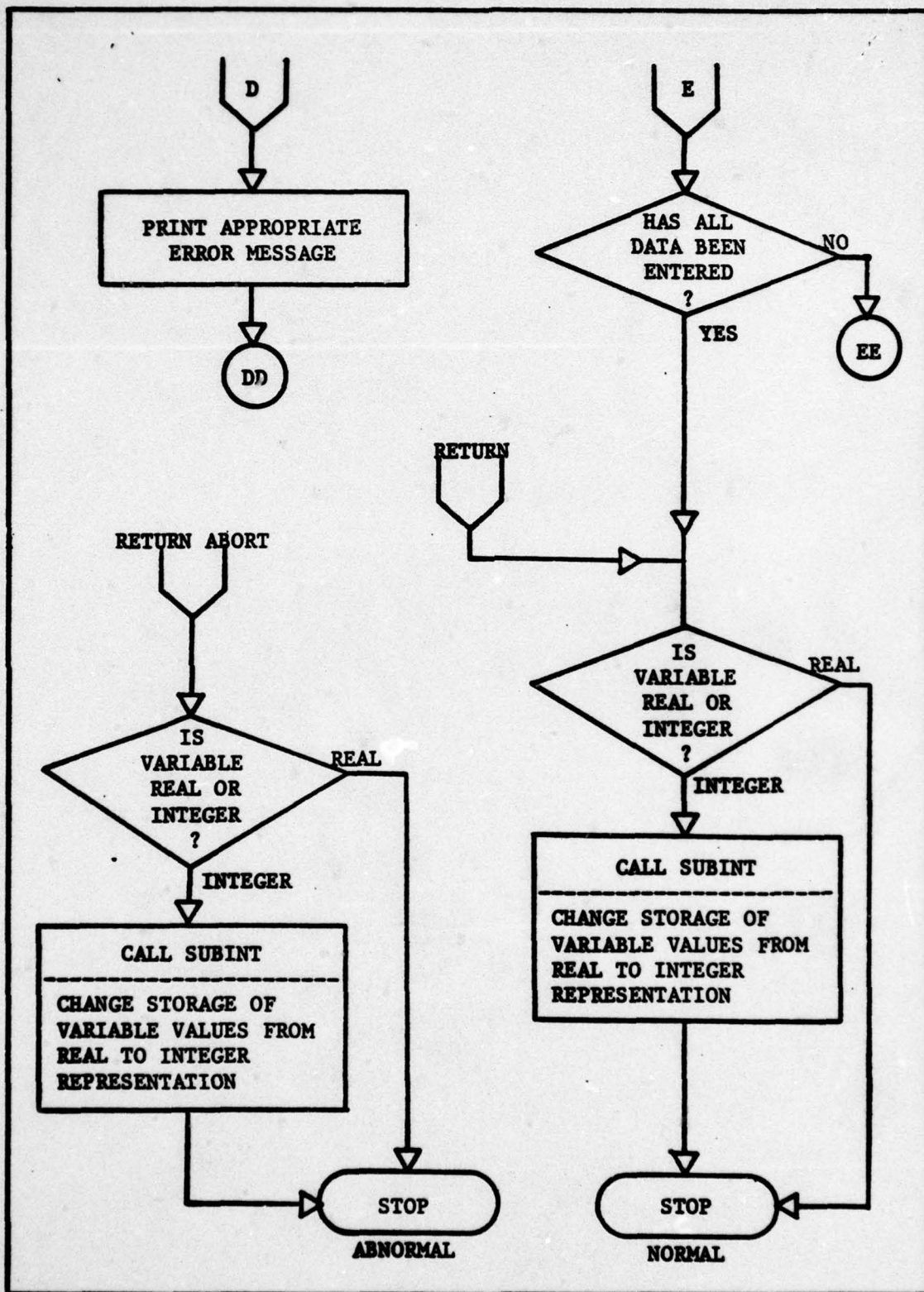


Figure 6. READNUM Flow Chart, Part V

The power in READNUM is particularly evident in matrix data entry. Each element of the matrix is constantly available for data entry or change. A data entry error noted after a carriage return can still be corrected on the next line. Six special commands give the user complete control over the data input process.

The basic flow chart for READNUM is shown in Figure 2. As can be seen, this subroutine can handle variables in real or integer representation. All data within the subroutine are represented as real numbers, and just prior to termination the subroutine converts them back to integer form if necessary.

All input characters are checked against the legal entry table, Table III. This table is stored as an array of 65 computer words. Each octal bit of the word is used in a different situation. Thus each character can cause one of 8 actions in up to 20 different situations. Situation 1 (the table is read from right to left) is used at the first decision point shown in Figure 2. The remainder of the flow chart for the READNUM subroutine is shown in Figures 3 thru 6.

All matrix elements are entered by rows in free format until all elements have been filled, unless the sequence is altered with one of the special commands. The first special character of interest is the asterisk (*), shown in Figure 3. The entry of the * causes the data entry pointer to move to the next element of the matrix with no change to the value of the current element. Figure 4 shows the section for decoding the six special alphanumeric commands and the special character dollar sign (\$). Any time the \$ is encountered a return abort is executed from the subroutine.

The six alphanumeric commands are "abort", "zero", "continue",

"read", "list", and "/n,m/". The "abort" command has the same result as the \$. "Zero" causes all elements of the variable to be set to zero and is useful for entering sparse matrices. "Continue" results in a normal termination of the subroutine regardless of the data entry pointer position. "Read" allows the input data to be read from the file TAPE1, TAPE2, or TAPE3. An optional parameter following the command "read n" is "rewind", which causes TAPEn to be rewound prior to reading data from it. The command "list" causes the current contents of the variable elements to be printed at the terminal. The optional parameter "/n,m/" following "list" is used to print only one element, one row, or one column of the variable. All of the above commands may be abbreviated to as little as the first letter. The command "/n,m/" is used to reset the data entry pointer to element (n,m). This allows previous errors to be corrected or several elements that need not be entered to be skipped.

The technical aspects for using this and other INTERAC subroutines for different programs can be found in the Programmer's Guide, Appendix C.

The second input subroutine developed is READCOM. This subroutine reads alphanumeric commands of 9 characters or less and checks them against a list of valid commands. Commands may have any characters, including single embedded blanks, except the slash (/), comma (,), and dollar sign (\$). Any abbreviation that is unique in the command list may be used. The flow chart for READCOM is shown in Figure 7. Multiple commands may be entered if they are separated by "/" or at least two blanks. A command string is also recognized and identified. A command string is a command followed by one or more other commands

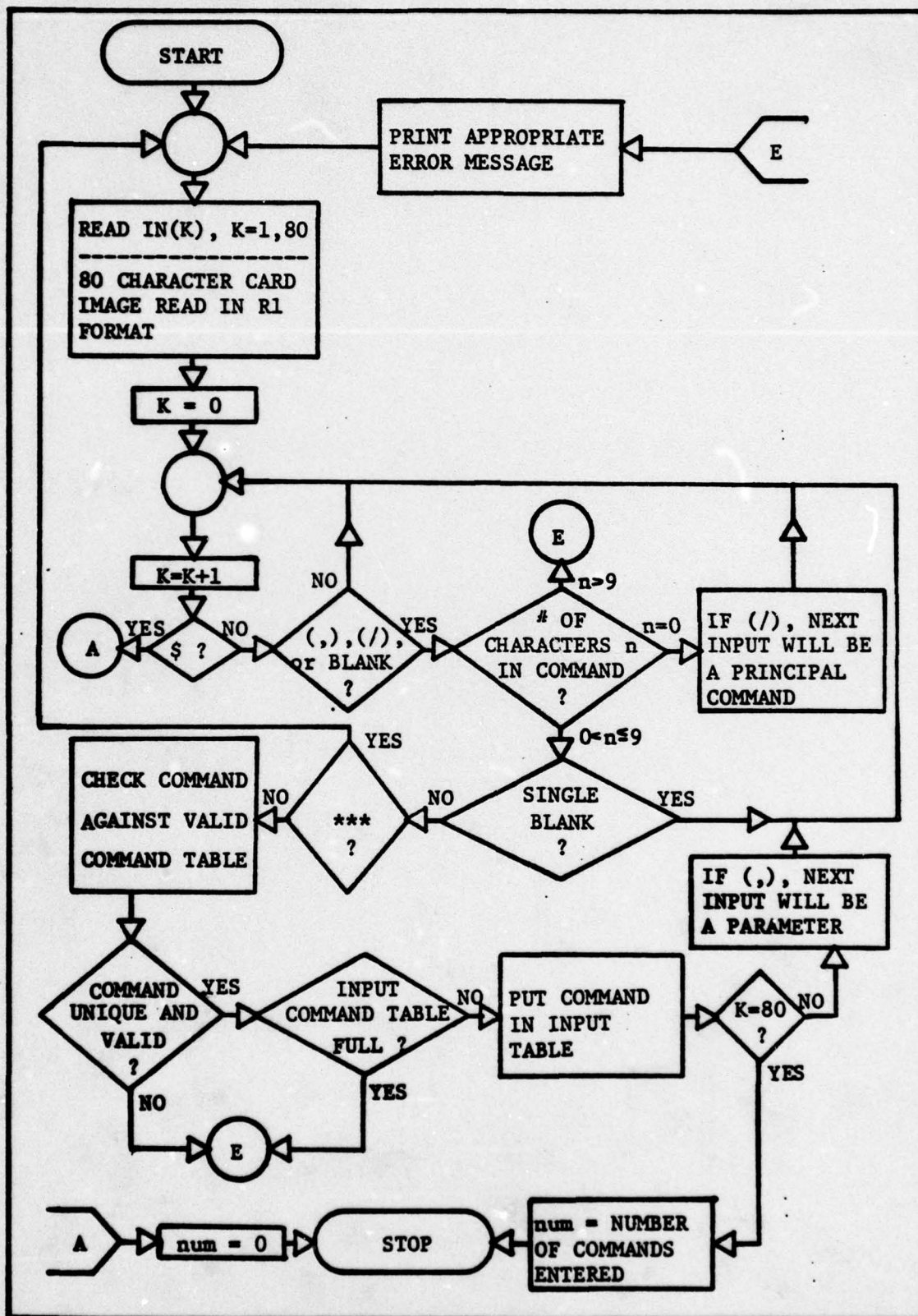


Figure 7. READCOM Flow Chart

acting as modifying parameters. The parameters are separated from the command and from each other by commas.

There are two special commands that are used internally in the subroutine. The \$ causes a return abort. Anytime the \$ is encountered the subroutine returns with an indication that no commands have been entered. The second special command is "***". READCOM normally reads only one 80 character input line. But when "***" is encountered as a command or parameter, another line is read. There are four error conditions recognized by READCOM. After the error message is printed the user is asked to reenter and is allowed another 80 character line.

Output Subroutines

Five of the legal commands interpreted by READCOM are used to change parameters in the output subroutines. Three of these commands are concerned with changing entries in the variable output table. Data output is sent to two possible locations as prescribed by the user with these commands. Data can be displayed at the terminal or can be written on a local file (to be disposed to a line printer after the program is terminated), or both.

The subroutine PRINTR is called by the computational subroutines whenever the value of a variable is changed. In this subroutine the variable output table is checked to see where the variable values should be printed. If display at the terminal or write to file are indicated the subroutine LISTER is called. Within LISTER the data is formatted to appear on a 72 character line for the terminal or a 132 character line for the line printer output as appropriate. The number of significant digits used in each case can be specified by the user.

Program Sequence Control

The output format and content as well as all program operations are controlled by the user's alphanumeric inputs. Table IV lists all legal commands available to the user.

Table IV

Legal Commands For The INTERAC Package

Utility Options	I-O Options	Design Options	Variables			
OPTIONS	ENTER	RUN	AMATRIX	AFMATRIX	CEIGEN	INTEGRATE
VARIABLES	CHANGE	FORTRAC	BMATRIX	AGMATRIX	DEIGEN	MODAL
SAVE	DFORMAT	SAMPLE	RMATRIX	AEMATRIX	DESIGEN	INVMODAL
STOP	OFORMAT	AUGMAT	CMATRIX	ARMATRIX	CLEIGEN	UREFORM
END	DISPLAY	TRANSFORM	FMATRIX	ACMATRIX	TSAMPLE	APLFORM
RESTART	OUTPUT	CONLAW	GMATRIX	BFMATRIX	DIMENSION	BRNFORM
REWIND	DELETE	CLOSELOOP	DEMATRIX	BGMATRIX	STATES	CINDICES
	PRINT	SETDIMEN	DRMATRIX	TINVERSE	CONTROLS	COFMATRIX
				KMATRIX	COMMANDS	
				CLMATRIX	DISTURBS	

Refer to Appendix B for the description and use of each command. The table of legal commands is stored in the array LIST, dimensioned n by 7. Column 1 of LIST contains the Hollerith representation of each option or variable name.

The other columns of LIST contain all the information needed to decide what action to take when the command is input by the user.

Column 2 divides the commands into four types: (1) option names, (2) real variable names, (3) integer variable names, and (4) output variable names. The output variables are intermediate results in a computational algorithm. These results are stored in temporary scratch locations in memory, and the user may not enter or change data within them. The user may indicate if and where the variable results should be printed when they are calculated. All real and integer variables are stored in contiguous locations in a common block. The other five columns of LIST indicate the starting location for the variable within the common block, the maximum row and column dimensions allowed, and the currently used row and column dimensions. Option names are divided into five categories by an entry in the last column of LIST. These categories are shown in Table V.

Table V
Option Categories

Entry In Column 7 Of LIST	Description
+n	Principal command only, with n required parameters
0	Principal command only, parameters optional
-1	Principal command only, no parameters allowed
-2	Principal command or parameter, no parameters allowed
-3	Parameter only

Subroutine COMND uses the entries in the array LIST to initiate the action requested by the user's command input. Refer to Appendix A for the listing of this subroutine.

IV. C-141 and F-4E Mathematical Modeling

An analog fly-by-wire control system for the C-141 has been designed and built by Honeywell, Inc. under contract to the Air Force Flight Dynamics Lab (Ref 12). The results of this project could provide data against which to check a digital control system designed using INTERAC. The second plant chosen to test the INTERAC package is an F-4E (Ref 13). This chapter describes the mathematical modeling process for the two plants. After several computer runs, the C-141 was dropped from consideration due to modeling problems to be discussed later. The chapter ends with a discussion of the results obtained for the F-4E.

C-141 Model

The model for the C-141 was derived using data from Reference 12. The longitudinal equations of motion are given as

$$\ddot{\theta} = M_{\dot{w}} \dot{w} + M_w w + M_q \dot{\theta} + M_{\delta e} \delta e + M_{\delta sp} \delta sp \quad (16)$$

$$\dot{w} = Z_u u + Z_w w + U_0 \dot{\theta} + Z_{\delta e} \delta e + Z_{\delta sp} \delta sp \quad (17)$$

$$\dot{u} = X_u u + X_w w - g\theta + X_{\delta sp} \delta sp \quad (18)$$

$$\dot{h} = -w + U_0 \theta \quad (19)$$

$$a_{z_{pilot}} = \dot{w} + U_0 \dot{\theta} - l_x \ddot{\theta} \quad (20)$$

The coefficients for the heavyweight landing configuration are given in Table VI (Ref 12:28,31).

Table VI
C-141 Longitudinal Dimensional Derivatives

257,000 lb Weight, Gear Down, 119 KCAS at Sea Level

Parameter	Value	Parameter	Value	Parameter	Value
M_w	-.00102	Z_u	-.321	X_u	-.044
$M_{\dot{w}}$	-.0051	Z_w	-.558	X_w	.083
M_q	-.645	U_e	201	g	32.2
$M_{\delta e}$	-.743	$Z_{\delta e}$	-5.36	$X_{\delta sp}$	-4.25
$M_{\delta sp}$	.240	$Z_{\delta sp}$	33.0	l_x	47
				U_{co}	350

The transfer functions for the elevator and spoiler actuator-servo combinations were modeled as given in Eq 21 and 22 respectively.

$$\frac{\delta e}{e_i} = \frac{10}{s + 10} \quad (21)$$

$$\frac{\delta sp}{sp_i} = \frac{5}{s + 5} \quad (22)$$

These transfer functions and the above equations of motion were modeled as state equations in the form of Eq (1).

$$\begin{aligned} \underline{\dot{x}}(t) &= \underline{A} \underline{x}(t) + \underline{B} \underline{u}(t) + \underline{R} \underline{d}(t) \\ \underline{y}(t) &= \underline{C} \underline{x}(t) \end{aligned} \quad (1)$$

where

$$\underline{x}(t) = \begin{bmatrix} \dot{\theta}(t) \\ \theta(t) \\ w(t) \\ u(t) \\ h(t) \\ \delta e(t) \\ \delta sp(t) \end{bmatrix} \quad \underline{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 10 & 0 \\ 0 & 5 \end{bmatrix} \quad \underline{u}(t) = \begin{bmatrix} e_i \\ sp_i \end{bmatrix} \quad \underline{R} = \emptyset$$

$$\underline{A} = \begin{bmatrix} -.85 & 0 & -.00453 & .00033 & 0 & -.736 & .206 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 201 & 0 & -.558 & -.321 & 0 & -5.36 & 33 \\ 0 & -32.2 & .083 & -.044 & 0 & 0 & -4.25 \\ 0 & 201 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -10 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -5 \end{bmatrix} \quad (23)$$

The $\underline{C}$ matrix was then chosen to give a C^* output. C^* is described in Ref 14 and the equation used is (Ref 12:55)

$$C^* = a_{z_{pilot}} + K_q \dot{\theta} \quad (24)$$

$$K_q = .217 \text{ for } U_{co} = 350$$

This was the first hint that the data provided by Ref 12 may not be valid. Using the definition of C^* from Ref 14, the value of K_q should have been 10.8. The $\underline{C}$ matrix then became.

$$\underline{C} = \begin{bmatrix} 1 & 0 & -.345 & -.336 & 0 & 29.3 & 23.3 \end{bmatrix} \quad (25)$$

where β is 442.2 or 452.8. The control law for a discrete-time time-optimal tracker was synthesized for both $\underline{C}$ matrices with similar results. The gains were excessive (greater than 10^4) and a time simulation showed huge overshoots and impossible control requirements (elevator deflections of 400°). A modeling problem was suspected when it was noted that the calculated Brunovsky transformation matrix $\underline{T}^{-1}$ was incorrect. This was due to computer numerical problems caused by the fact that the plant as modeled was almost uncontrollable. In an attempt to verify the modeling, the coefficients taken from Page 28 and 31 of Ref 12 were checked against the analog flow chart on Page 33 of Ref 12. There were numerous numerical differences and a sign difference in the $\int_{sp}$ term in the $\dot{u}$ equation (Eq 18). Thus the validity of any comparison of a discrete fly-by-wire system developed using INTERAC/FORTRAC with the analog system described in Ref 12 was in serious doubt. At this time it was decided to attempt a control system design for another aircraft.

F-4E Model

The F-4E was modeled using data from a lab project from the EE 6.41 course at AFIT (Ref 13). The coefficients of Table VII were used in the following equations of motion:

$$\ddot{\theta} = \frac{1}{\left(\frac{I_y}{S q c}\right)} \left[\frac{c}{2U_0} C_{m_{\dot{\alpha}}} \dot{\alpha}' + C_{m_{\alpha}} \alpha' + \frac{c}{2U_0} C_{m_{\dot{q}}} \dot{\theta}' + C_{m_{\delta e}} \delta e \right] \quad (26)$$

$$\dot{\alpha}' = \frac{1}{\left(\frac{mU_0}{Sq} - \frac{c}{2U_0} C_{Z\dot{\alpha}}\right)} \left[C_{Zu} u' + C_{Z\alpha} \alpha' + \left(\frac{mU_0}{Sq} + \frac{c}{2U_0} C_{Zq}\right) \dot{\theta}' + C_{Z\delta_e} \delta_e \right] \quad (27)$$

$$\dot{u}' = \frac{1}{\left(\frac{mU_0}{Sq}\right)} \left[C_{Xu} u' + C_{X\alpha} \alpha' + C_{W} \theta' + C_{X\delta_T} \delta_T \right] \quad (28)$$

Table VII
F-4E Longitudinal Non-dimensional Derivatives

33,439 lb Weight, Gear Down, 138 KCAS at Sea Level

Parameter	Value	Parameter	Value	Parameter	Value
I_y	137,710	$C_{m\dot{\alpha}}$	-.9801	C_{Zu}	-2.0
U_0	230.5	$C_{m\alpha}$	-.07634	$C_{Z\alpha}$	-2.967
q	63.14	C_{mq}	-2.066	C_{Zq}	-2.276
c	16.04	$C_{m\delta_e}$	-.5776	$C_{Z\delta_e}$	-.3887
g	32.174	C_{Xu}	-.339	$C_{X\delta_T}$	.2
s	530	$C_{X\alpha}$	.630		

Dimensionalizing and other relationships

$$n = w/g \quad C = -ng/Sq \quad u' = u/U_0 \quad \alpha = 57.3\alpha' \quad \theta = 57.3\theta'$$

The elevator actuator and servo were modeled as the first order lag

$$\frac{\delta_e}{e_i} = \frac{20}{s + 20} \quad (29)$$

and the engine and throttle actuator were modeled as

$$\frac{\delta_T}{T_i} = \frac{2}{s + 2} \quad (30)$$

The statespace equation for the F-4E then became

$$\begin{aligned} \dot{\underline{x}}(t) &= \underline{A}\underline{x}(t) + \underline{B}\underline{u}(t) + \underline{R}\underline{d}(t) \\ \underline{y}(t) &= \underline{C}\underline{x}(t) \end{aligned} \quad (1)$$

where

$$\underline{x}(t) = \begin{bmatrix} \dot{\theta}(t) \\ \theta(t) \\ \alpha(t) \\ u(t) \\ e(t) \\ T(t) \end{bmatrix} \quad \underline{B} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 20 & 0 \\ 0 & 2 \end{bmatrix} \quad \underline{u}(t) = \begin{bmatrix} e(t) \\ T(t) \end{bmatrix} \quad \underline{R} = \underline{\emptyset}$$

$$\underline{A} = \begin{bmatrix} -4115 & 0 & -.2424 & .009231 & -2.244 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ .989 & 0 & -.4146 & -.06946 & -.05431 & 0 \\ 0 & -.5717 & .3541 & -.04737 & 0 & 6.442 \\ 0 & 0 & 0 & 0 & 20 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 \end{bmatrix} \quad (31)$$

The $\underline{C}$ matrix was chosen to allow tracking of flight path angle, $\theta - \alpha$, as would be desirable in an automatic approach.

$$\underline{C} = \begin{bmatrix} 0 & 1 & -1 & 0 & 0 & 0 \end{bmatrix} \quad (32)$$

A control law was synthesized to track flight path angle for various sampling times.

Results of F-4E Control Law Synthesis

At all sampling times a simulation showed the F-4E to be on track within 4 sample periods. This requires higher gains and higher control power as the sampling interval decreases. Table VIII shows some of the representative data obtained for sampling intervals of .05 to 5.0 seconds, with a unit step command of 1 degree.

Table VIII
Simulation Data for the F-4E with a Discrete-Time
Flight Path Angle Tracking Control System

Sampling Interval Seconds	----- Maximum -----				Final Change in Forward Velocity ft/sec
	Elevator Deflection	Change in Forward Velocity ft/sec	Negative Peak in Commanded Output	Magnitude of Feedback Gain Required	
.05	-8010 °	-255	-15.6°	2.5×10^5	-255
.1	-1245 °	-154	- 4.9°	1.4×10^4	-154
.25	- 87.6°	- 62	- .7°	353	- 62
.5	- 12 °	- 29	- .1°	24	- 29
.6	- 7.3°	- 24	- .01°	15	- 24
.7	.4°	16	0.0	17.5	3.6
.8	.4°	14	0.0	12.5	3.2
.9	.3°	13	0.0	9.3	3.1
1.0	.3°	11	0.0	7.2	2.9
2.0	.2°	5	0.0	1.2	.08
5.0	.3°	13	0.0	.7	13.4

At sample times below .5 seconds, the elevator deflection required is beyond the capability of the aircraft. A sample time of 2 seconds requires the least control power and produces the smoothest response. But 8 seconds to track a 1 degree flight path change is excessive for aircraft control. Even the lowest practical sampling time of .5 seconds, may be too high for this fighter type aircraft.

The discrete-time time-optimal control seems to work quite well, if a relatively long sampling interval can be tolerated. If a higher sampling time is required, a sub-optimal control must be used. One possibility is to use deadbeat control, which in this case would allow 7 sample periods to track the command. The deadbeat algorithm is not presently in the INTERAC package, but it could be easily added. The second possibility is to place the desired eigenvalues at some place other than the origin in the z-plane. Since there are an infinite number of possible eigenvalues to use, the INTERAC package is more desirable than the batch mode program for quickly trying the most promising choices of eigenvalues.

V. Conclusions and Recommendations

Conclusions

The INTERAC package meets all of the specifications given in Chapter 1. There is still a little room for improvement in the I-O versatility, but what is now presented is a vast improvement over existing programs. In a current investigation, Lt. Stanley J. Larimer is incorporating many of these user oriented ideas into a classical control design package.

The use of discrete-time time-optimal control systems for total aircraft control is not generally possible due to the long sampling interval required to keep the magnitude of the control within practical limits. There are applications, with low frequency disturbances and commands, where the time-optimal control could be used, such as cruise control and possibly automatic approach control. The total aircraft control system will probably require a sub-optimal control law.

The theory behind the FORTRAC and INTERAC packages does not provide a cut and dry control system design process. There are too many possible parameter changes in the multi-input, multi-output system for a one best answer to be achieved in one iteration. The INTERAC package provides a basis for quickly investigating the various parameter changes.

Recommendations

(1) A graphical simulation capability should be added to the INTERAC package. The magnitude of the feedback gains does give a general idea as to the response of the closed loop system. But, as

shown for the sampling interval of .6 and .7 seconds in Table VIII, there can be a drastic change in the total system response without large changes in the feedback gains.

(2) The DEADBEAT subroutine for achieving the sub-optimal control law should be added to the package. This subroutine is in the FORTRAC package, but it is not called by the master program furnished.

(3) The investigation of the discrete-time fly-by-wire control system for the C-141 should be continued with basic flight test data from Lockheed. The heavy transport type aircraft does not normally require high frequency control and response, and thus it would be ideal for further investigating the time-optimal control laws.

Bibliography

1. ROOTL - User's Manual for a Digital Computer Program to Calculate and Plot the Root Locus, Unpublished user's guide to computer program. Ohio: Air Force Institute of Technology, January 1976.
2. FREQR - Frequency Response Program, Unpublished user's guide to computer program. Ohio: Air Force Institute of Technology, November 1975.
3. PARTL - Heaviside Partial Fraction Expansion and Time Response Program, Unpublished user's guide to computer program. Ohio: Air Force Institute of Technology, December 1974.
4. OPTCON - Solution of Linear Quadratic Optimal Control Problem, Unpublished user's guide to computer program. Ohio: Air Force Institute of Technology, January 1976.
5. MIMIC Version 3.4, Unpublished user's guide to computer program. Ohio: Air Force Institute of Technology, August 1974.
6. Porter, B. FORTRAC: A Software Package for the Design of Multivariable Digital Control Systems, Unpublished report. Salford, England: University of Salford, April 1977.
7. Porter, B. and Bradshaw, A. Unpublished set of 10 papers on discrete-time control. Salford, England: University of Salford.
8. Aplevich, J.D. "Direct Computation of Canonical Forms for Linear Systems by Elementry Matrix Operations." IEEE Transactions on Automatic Control, AC-19: pp 124-126 (April 1974)
9. Nikolai, Paul J. Solution of the Eigenproblem on the CDC-6600/Cyber-74 Processors. AFFDL-TM-76-131-FBR. Ohio: Air Force Flight Dynamics Lab, January 1977.
10. Control Data Corporation, FORTRAN Extended Version 4 Reference Manual. California: Control Data Corporation, 1975.
11. Control Data Corporation, SCOPE 3.4 Reference Manual. California: Control Data Corporation, 1975.
12. Honeywell, Inc. Military Transport (C-141) Fly-by-Wire Program, Vol I. AFFDL-TR-74-52. Minnesota: Honeywell, Inc., April 1974.
13. EE 6.41 Automatic Control Lab, Laboratory Experiment #3, Unpublished laboratory problem and data definition. Ohio: Air Force Institute of Technology, Sprint Qtr 1977.
14. Malcolm, Tobie. New Short Period Handling Qualities Criterion for Fighter Aircraft (C* Control Criterion). Washington: The Boeing Company, September 1965.

Appendix A

Listing of Specialized INTERAC Software

The following pages are a listion of the FORTRAN Extended source code for the specialized INTERAC software.

```

PROGRAM INTERAC(INPUT=1008/80,OUTPUT=10008,TAPE5=INPUT,
1 INTERAC
2 TAPE6=OUTPUT,TAPE8=1008,TAPE7=1008,ANSWER=1008,
3 INTERAC
4 TAPE1=1008,TAPE2=1009,TAPE3=1008,DATA=5009,
5 INTERAC
6 TAPE39=ANSWER,TAPE55=DATA,TAPE65=5008)
7 INTERAC
8 COMMON/LISTING/NL,LIST(75,7)
9 COMMON/LISTOUT/N,LISTO(75)
10 INTERAC
11 WHEN CHANGING DIMENSIONS OF THE ARRAY LIST THE ROW DIMENSION OF
12 LISTO AND THE DIMENSION STATEMENTS FOR LIST IN SUBROUTINES
13 FOPTRAC,COMND, OPEN, AND STRING MUST ALSO BE CHANGED ACCORDINGLY.
14 INTERAC
15 COMMON/RVAR/NTOT,TOT(3500)
16 COMMON/IVAR/NITOT,ITOT(25)
17 COMMON/CONTROL/DSP,BATCH
18 COMMON/TITLE/ITITLE(5)
19 COMMON/FORMS/NOREP,NDFW,NONSD,NOREP,NDFW,NONSD
20 DATA (LIST( 1,I),I=1,7),LISTO( 1)/10HENTER E,3HCMD,
21 DATA (LIST( 2,I),I=1,7),LISTO( 2)/10HSTOP 1, 0, 0, 0, 0, 1/
22 DATA (LIST( 3,I),I=1,7),LISTO( 3)/10HAMATRIX G,3HREL, 2, 0, 0, 0, 0,-1,1/
23 DATA (LIST( 4,I),I=1,7),LISTO( 4)/10HAMATRIX G,3HREL, 1, 1, 1,20,20,4/
24 DATA (LIST( 5,I),I=1,7),LISTO( 5)/10HEND 401, 1, 1,20,10,4/
25 DATA (LIST( 6,I),I=1,7),LISTO( 6)/10HPRINT 3, 0, 0, 0,-1,1/
26 DATA (LIST( 7,I),I=1,7),LISTO( 7)/10HDFORMAT G,3HCMD, 4, 0, 0, 0, 3,1/
27 DATA (LIST( 8,I),I=1,7),LISTO( 8)/10HCHANGE 5, 0, 0, 0,-1,1/
28 DATA (LIST( 9,I),I=1,7),LISTO( 9)/10HDISPLAY F,3HCMD, 6, 0, 0, 3, 0,1/
29 DATA (LIST(10,I),I=1,7),LISTO(10)/10HOUTPUT 7, 0, 0, 0, 0,1/
30
31
32
33
34
35
36

```


1	DATA (LIST(28,I),I=1,7),LISTO(28)/10HOFORMAT 22, 0, 0, 0,-1,1/	INTERAC	72
1	DATA (LIST(29,I),I=1,7),LISTO(29)/10HSETOIMEN H,3HCMD,	INTERAC	73
1	DATA (LIST(30,I),I=1,7),LISTO(30)/10HRMATRIX G,3HREL,	INTERAC	74
1	DATA (LIST(31,I),I=1,7),LISTO(31)/10HCMATRIX G,3HREL,	INTERAC	75
1	DATA (LIST(32,I),I=1,7),LISTO(32)/10HFMATRIX G,3HREL,	INTERAC	76
1	DATA (LIST(33,I),I=1,7),LISTO(33)/10HGMATRIX G,3HREL,	INTERAC	77
1	DATA (LIST(34,I),I=1,7),LISTO(34)/10HDEMATRIX H,3HREL,	INTERAC	78
1	DATA (LIST(35,I),I=1,7),LISTO(35)/10HORMATRIX H,3HREL,	INTERAC	79
1	DATA (LIST(36,I),I=1,7),LISTO(36)/10HAFMATRIX H,3HREL,	INTERAC	80
1	DATA (LIST(37,I),I=1,7),LISTO(37)/10HAGMATRIX H,3HREL,	INTERAC	81
1	DATA (LIST(38,I),I=1,7),LISTO(38)/10HAEMATRIX H,3HREL,	INTERAC	82
1	DATA (LIST(39,I),I=1,7),LISTO(39)/10HARMATRIX H,3HREL,	INTERAC	83
1	DATA (LIST(40,I),I=1,7),LISTO(40)/10HACMATRIX H,3HREL,	INTERAC	84
1	DATA (LIST(41,I),I=1,7),LISTO(41)/10HCEIGEN F,3HREL,	INTERAC	85
1	DATA (LIST(42,I),I=1,7),LISTO(42)/10HCLEIGEN G,3HREL,	INTERAC	86
1	DATA (LIST(43,I),I=1,7),LISTO(43)/10HBFMATRIX H,3HREL,	INTERAC	87
1	DATA (LIST(44,I),I=1,7),LISTO(44)/10HBGMATRIX H,3HREL,	INTERAC	88
1	DATA (LIST(45,I),I=1,7),LISTO(45)/10HTINVERSE H,3HREL,	INTERAC	89
		INTERAC	90
		INTERAC	91
		INTERAC	92
		INTERAC	93
		INTERAC	94
		INTERAC	95
		INTERAC	96
		INTERAC	97
		INTERAC	98
		INTERAC	99
		INTERAC	100
		INTERAC	101
		INTERAC	102
		INTERAC	103
		INTERAC	104
		INTERAC	105
		INTERAC	106


```

SUPROUTINE COMND
COMMON/LISTING/NL,LIST(75,7)
COMMON/LISTOUT/NO,LISTO(1)
COMMON/RVAR/NTOT,TOT(1)
COMMON/IVAR/NITOT,ITOT(1)
COMMON/CONTROL/OSP,RATCH
COMMON/FORMS/NDREP,NDFW,NONSD,NOREP,NOFW,NONSD
LOGICAL RATCH
INTEGER DSP,CHD(12),CMST,CMED
CALL CONNEC(FLINFUT)
CALL CONNEC(SLTAPE8)
CALL DATE(TODAY)
CALL TIME(CLOCK)
PRINT(8,*)" IF YOU ARE AT A TERMINAL TYPE TTY >"
READ 5,JCN,KCN
5 FORMAT(R3,1X,R1)
IF(JCN.EQ.3RTTY) GO TO 10
DSP=6
RATCH=.TRUE.
GO TO 12
10 DSP=8
RATCH=.FALSE.
CALL DISCON(6LOUTPUT)
REWIND 6
12 PRINT(6,15) TODAY,CLOCK
15 FORMAT("1",10X,"START
IF(KCN.NE.1RS) GO TO 5000
20 PRINT(F,*)
PRINT(F,*)
PRINT(6,*)" COMMAND >>"
NUM=12
CALL RFADCON(CHD,NUM)
IF(CHD(1).EQ.0) GO TO 90
CMFN=0
25 CMST=CMED+2
DATE ",A10,5X,"TIME ",A10,////)

```

2 COMND
 3 COMND
 4 COMND
 5 COMND
 6 COMND
 7 COMND
 8 COMND
 9 COMND
 10 COMND
 11 COMND
 12 COMND
 13 COMND
 14 COMND
 15 COMND
 16 COMND
 17 COMND
 18 COMND
 19 COMND
 20 COMND
 21 COMND
 22 COMND
 23 COMND
 24 COMND
 25 COMND
 26 COMND
 27 COMND
 28 COMND
 29 COMND
 30 COMND
 31 COMND
 32 COMND
 33 COMND
 34 COMND
 35 COMND
 36 COMND


```

37 COMND
38 COMND
39 COMND
40 COMND
41 COMND
42 COMND
43 COMND
44 COMND
45 COMND
46 COMND
47 COMND
48 COMND
49 COMND
50 COMND
51 COMND
52 COMND
53 COMND
54 COMND
55 COMND
56 COMND
57 COMND
58 COMND
59 COMND
60 COMND
61 COMND
62 COMND
63 COMND
64 COMND
65 COMND
66 COMND
67 COMND
68 COMND
69 COMND
70 COMND
71 COMND

K=0
DO 30 I=CMST,NUM
IF(CMD(I).GE.0) GO TO 35
K=K+1
30 CONTINUE
35 CMST=CMST-1
IF(K.EQ.0) GO TO 50
IF(LIST(CMD(CHST),2).NE.3HCMD) GO TO 45
IF(LIST(CMD(CHST),7).LT.0 .OR. LIST(CMD(CHST),7).GT.K) GO TO 45
CALL STPING(LIST(CMD(CHST),3),CMD(CHST+1),K),RETURNS(20))
GO TO 40
40 CMEN=CMST+K
IF(CMED.EQ.NUM) GO TO 20
GO TO 25
45 PRINT(6,46)AND(LIST(CMD(CHST),1),77R),LIST(CMD(CHST),1),
1 AND(LIST(-CMD(CHST+K),1),77R),LIST(-CMD(CHST+K),1)
IF(RATCH) STOP
GO TO 40
46 FORMAT(/,5X,"COMMAND STRING",3X,A=",",.....",A=,3X,
1 "ILLEGAL AND IGNORED.",/)
50 JCN=CMD(CHST)
IF(LIST(JCN,2).EQ.3HOUT) GO TO 60
IF(LIST(JCN,2).EQ.3HREL .OR. LIST(JCN,2).EQ.3HINT) GO TO 75
IF(LIST(JCN,7).EQ.-3 .OR. LIST(JCN,7).GT.0) GO TO 60
GO TO ( 100, 200, 300, 400, 500, 600, 700, 800, 900,1000,
1 1100,1200,1300,1400,3000,1600,1700,1800,1900,2000,
2 2100,2200,2300,3000,3000,3000,3000,3000,3000,3000)
3 LIST(JCN,3)
60 PRINT(6,65)AND(LIST(JCN,1),77R),LIST(JCN,1)
GO TO 40
65 FORMAT(/10X,A=,3X,"ILLEGAL AND IGNORED",/)
75 CALL OFEN(JCN),RETURNS(20)
GO TO 40
90 IF(RATCH) STOP

```

72	COMND	GO TO 20	
73	COMND	C*****ENTER**	
74	COMND	100 PRINT (0,150)	
75	COMND	NC=1	
76	COMND	CALL READCOM(ICOM,NC)	
77	COMND	IF(ICOM.EQ.0) GO TO 90	
78	COMND	IF(LIST(ICOM,1).EQ.10)END	
79	COMND	IF(LIST(ICOM,2).EQ.3)CMD .OR. LIST(ICOM,2).EQ.3)OUT> GO TO 120	
80	COMND	CALL OPEN(ICOM), RETURNS(20)	
81	COMND	GO TO 100	
82	COMND	120 PRINT (0,05)AND(LIST(ICOM,1),778),LIST(ICOM,1)	
83	COMND	GO TO 100	
84	COMND	150 FORMAT(/10X,"ENTER VARIABLE NAME OR END >>")	
85	COMND	C*****STOP** **END**	
86	COMND	200 PRINT (0,250)	
87	COMND	250 FORMAT(/,5X,"INTERAC TERMINATED, ANSWER IS DISCONNECTED AND CONTAI	
88	COMND	1NS",/, " YOUR PRINTED DATA. FILE DATA HAS THE CURRENT CONTENTS",	
89	COMND	2/, " OF ALL VARIABLES; THEY CAN BE RELOADED AT SOME FUTURE TIME BY	
90	COMND	3",/, " THE COMMAND RESTART."	
91	COMND	300 CALL DATE(TODAY)	
92	COMND	CALL TIME(CLOCK)	
93	COMND	PRINT (55,340) TODAY,CLOCK	
94	COMND	PRINT (0,345) TODAY,CLOCK	
95	COMND	340 FOPMAT(2A10)	
96	COMND	345 FORMAT(5X,"DATA SAVED AT ",2A10)	
97	COMND	PRINT (55,*) (TOT(I),I=1,NTOT)	
98	COMND	PRINT (55,*) (ITOT(I),I=1,NITOT)	
99	COMND	PRINT (55,*) ((LIST(I,4),LIST(I,5)),I=1,NL)	
100	COMND	PRINT (55,*) ((LISTO(I),I=1,NO)	
101	COMND	CALL RETURN(5LTAPE7)	
102	COMND	CALL RETURN(5LTAPE8)	
103	COMND	CALL RETURN(5LTAPE6,5)	
104	COMND	PRINT (0,350) TODAY,CLOCK	
105	COMND	350 FOPMAT("2",10X,"STOP	
106	COMND	DATE ",A10,5X,"TIME ",A10)	
		CALL FCRMOUT	

620	PRINT(8,65)AND(LIST(ICOM,1),778),LIST(ICOM,1)	142	COMND
GO TO 600		143	COMND
C*****DISPLAY**		144	COMND
700	PRINT(8,*)" VARIABLES TO BE DISPLAYED AT TERMINAL"	145	COMND
DO 720 J=1,NL		146	COMND
IF(LISTO(J).NE.2 .AND. LISTO(J).NE.3) GO TO 720		147	COMND
PRINT(8,750)AND(LIST(J,1),778),LIST(J,1)		148	COMND
720	CONTINUE	149	COMND
GO TO 40		150	COMND
750	FORMAT(5X,A=)	151	COMND
C*****OUTPUT**		152	COMND
800	PRINT(8,*)" VARIABLES TO BE PRINTED TO ANSWER"	153	COMND
DO 820 J=1,NL		154	COMND
IF(LISTO(J).NE.3 .AND. LISTO(J).NE.4) GO TO 820		155	COMND
PRINT(8,850)AND(LIST(J,1),778),LIST(J,1)		156	COMND
820	CONTINUE	157	COMND
GO TO 40		158	COMND
850	FORMAT(5X,A=)	159	COMND
C*****DELETE**		160	COMND
900	PRINT(8,*)" ALL VARIABLES HAVE BEEN REMOVED FROM THE DISPLAY AND	161	COMND
1000	OUTPUT LISTS."	162	COMND
DO 920 J=1,NL		163	COMND
LISTO(J)=1		164	COMND
920	CONTINUE	165	COMND
GO TO 40		166	COMND
C*****OPTIONS**		167	COMND
1000	PRINT(8,*)" VALID COMMANDS"	168	COMND
DO 1020 J=1,NL		169	COMND
IF(LIST(J,2).NE.3HCMD) GO TO 1020		170	COMND
IF(LIST(J,7)) 1005,1010,1015		171	COMND
1005	PRINT(8,1050)AND(LIST(J,1),778),LIST(J,1)	172	COMND
GO TO 1020		173	COMND
1010	PRINT(8,1060)AND(LIST(J,1),778),LIST(J,1)	174	COMND
GO TO 1020		175	COMND
1015	PRINT(8,1070)AND(LIST(J,1),778),LIST(J,1)	176	COMND


```

1020 CONTINUE
      GO TO 40
1050 FORMAT(5X,A=)
1060 FORMAT(5X,A=,T18,"(,PARAMETER LIST)")
1070 FORMAT(5X,A=,T18,"(,PARAMETER(S)")
C*****VARIABLES**
1100 PRINT(8,*)"  VALID VARIABLES          MAX DIMENSION ALLOWED"
      NO 1120 J=1,NL
      TF(LIST(J,2).EQ.3HCMD) GO TO 1120
      ISIM=1H
      IF(LIST(J,2).EQ.3HOUT) ISIM=1H*
      PRINT(8,1150)ISIM,AND(LIST(J,1),77B),LIST(J,1),LIST(J,6),LIST(J,7)
1120 CONTINUE
      PRINT(8,*)
      PRINT(8,*)" * -- TEMPORARY VARIABLE PRINTED ONLY DURING PROGRAM EXECUTION."
      GO TO 40
1150 FORMAT(5X,A2,A=,T33,"(",I3,"",I3,"")
C*****SAVE**
1200 CALL DATE(TODAY)
      CALL TIME(CLOCK)
      PRINT(55,340)TODAY,CLOCK
      PRINT(8,345)TODAY,CLOCK
      PRINT(55,*) (TOT(I), I=1,NTOT)
      PRINT(55,*) (ITOT(I), I=1,NITOT)
      PRINT(55,*) ((LIST(I,4),LIST(I,5)), I=1,NL)
      PRINT(55,*) (LISTO(I), I=1,NO)
      GO TO 40
C*****DESTART**
1300 REWIND 65
      PRINT(55,*) (TOT(I), I=1,NTOT)
      PRINT(55,*) (ITOT(I), I=1,NITOT)
      PRINT(55,*) ((LIST(I,4),LIST(I,5)), I=1,NL)
      PRINT(55,*) (LISTO(I), I=1,NO)
      REWIND(55,340)TODAY,CLOCK

```

```

IF(EOF(6LTAPE55).NE.0.0) GO TO 1350
READ(55,*)(TOT(I),I=1,NTOT)
IF(EOF(6LTAPE55).NE.0.0) GO TO 1350
READ(55,*)(ITOT(I),I=1,NTOT)
IF(EOF(6LTAPE55).NE.0.0) GO TO 1350
READ(55,*)((LIST(I,4),LIST(I,5)),I=1,NL)
IF(EOF(6LTAPE55).NE.0.0) GO TO 1350
READ(55,*)(LISTO(I),I=1,NO)
IF(EOF(6LTAPE55).NE.0.0) GO TO 1350
PRINT(8,1340) TODAY, CLOCK
1340 FORMAT(/,5X,"DATA SAVED AT ",2A10," RELOADED.")
GO TO 40
1350 PRINT(8,*)" EOF ENCOUNTERED ON RESTART - REWIND IS PROBABLY NEEDED"
1SARY"
REWIND 65
READ(65,*)(TOT(I),I=1,NTOT)
READ(65,*)(ITOT(I),I=1,NTOT)
READ(65,*)((LIST(I,4),LIST(I,5)),I=1,NL)
READ(65,*)(LISTO(I),I=1,NO)
GO TO 40
C*****REWIND**
1400 REWIND 55
GO TO 40
C*****FORTRAC**
1500 CALL FORTRAC(1)
GO TO 40
C*****SAMPLE**
1700 CALL FORTRAC(2)
GO TO 40
C*****AUGMAT**
1800 CALL FORTRAC(3)
GO TO 40
C*****TRANSFCRM**
1900 CALL FORTRAC(4)
GO TO 40

```



```

PEAD(65,*) (LISTO(I), I=1, NO)
RETURN
135 PRINT(A, 136) AND (LIST(CMD(K), 1), 778), LIST(CMD(K), 1)
136 FORMAT(/, 5X, "COMMAND STRING RESTART, ..., ", A=, " ILLEGAL AND IGNORE
10")
RETURN
137 PRINT(A, 138) AND (LIST(CMD, 1), 778), LIST(CMD, 1)
138 FORMAT(/, 5X, "RESTART, ", A=, " ILLEGAL AND IGNORED")
RETURN
C*****RUN**
150 IF(K.GT.1) GO TO 155
IF(LIST(CMD(1), 6).EQ.0 .OR. LIST(CMD(1), 7).NE.-2) GO TO 157
CALL FORTRAC(LIST(CMD(1), 6)+10)
RETURN
155 PRINT(A, 156) AND (LIST(CMD(K), 1), 778), LIST(CMD(K), 1)
156 FORMAT(/, 5X, "COMMAND STRING RUN, ..., ", A=, " ILLEGAL AND IGNORED")
RETURN
157 PRINT(A, 158) AND (LIST(CMD(1), 1), 778), LIST(CMD(1), 1)
158 FORMAT(/, 5X, "RUN, ", A=, " ILLEGAL AND IGNORED")
RETURN
500 PRINT(A, *) " COMMAND STRING NOT AVAILABLE"
RETURN
END

```

STRING 107
 STRING 108
 STRING 109
 STRING 110
 STRING 111
 STRING 112
 STRING 113
 STRING 114
 STRING 115
 STRING 116
 STRING 117
 STRING 118
 STRING 119
 STRING 120
 STRING 121
 STRING 122
 STRING 123
 STRING 124
 STRING 125
 STRING 126
 STRING 127
 STRING 128
 STRING 129

[illegible]


```

C*****
200 IF(IN(K).EQ.1R+) GO TO 247
   IF(POINTER.GT.NN .AND. END) GO TO 249
   NE=0
   NUMST=K
   I=1
   IF(IN(K).EQ.1R.) I=2
210 K=K+1
   IF(I.EQ.3) I=4
215 GO=AND(SHIFT(A(IN(K)+2),-3*I),AM1)
   GO TO (210,220,230,240,250,260,270) GO
220 IF(IN(K-1).LE.36) GO TO 225
   IF(K-2.EQ.0) GO TO 500
   IF(IN(K-2).LE.36) GO TO 225
   GO TO 500
225 I=3
   K=K+1
   NE=K
   GO TO 215
230 I=2
   GO TO 210
240 IF(I.EQ.3) GO TO 500
   IF(I.EQ.4) GO TO 241
   IF(IN(K-1).LE.36) GO TO 245
   IF(K-2.EQ.0) GO TO 500
   IF(IN(K-2).LE.36) GO TO 245
   GO TO 500
241 L=0
   IF(IN(NE).EQ.1R+ .OR. IN(NE).EQ.1R-) L=1
   IF(K-NF-3-L)245,242,500
242 IF(IN(NE+L)-29)245,243,500
243 IF(IN(NE+L+1)-36)245,244,500
244 IF(IN(NE+L+2).GT.1R3) GO TO 500
245 REWIND 7
   L=K-1

```

```

READNUM 167
READNUM 168
READNUM 169
READNUM 170
READNUM 171
READNUM 172
READNUM 173
READNUM 174
READNUM 175
READNUM 176
READNUM 177
READNUM 178
READNUM 179
READNUM 180
READNUM 181
READNUM 182
READNUM 183
READNUM 184
READNUM 185
READNUM 186
READNUM 187
READNUM 188
READNUM 189
READNUM 190
READNUM 191
READNUM 192
READNUM 193
READNUM 194
READNUM 195
READNUM 196
READNUM 197
READNUM 198
READNUM 199
READNUM 200
READNUM 201

```



```

WRITE(7,99)(IN(J),J=NUMST,L)
REWIND 7
READ(7,*)MATRIX(IROW,ICOL)
247 POINTER=POINTER+1
IROW=(POINTER-1)/M+1
ICOL=POINTER-(IROW-1)*M
IF(POINTER.GE.NN+1) GO TO 249
248 KK=K
IF(IN(K).EQ.1R*) KK=K+1
GO TO 10
249 END=.TRUE.
II=8
GO TO 248
250 GO TO 520
260 GO TO 560
270 GO TO 300
C*****
57 C
C*****
300 PEW=1
IF(IN(K).EQ.1R*) GO TO 330
305 WORDST=K
I=6
L=-1
310 K=K+1
L=L+1
GO=AND(SHIFT(A(IN(K)+2),-3*I),AM1)
IF(GO.EQ.5) GO TO 330
IF(GO.EQ.2) GO TO 320
IF(GO.EQ.3) GO TO 520
IF(L.GT.9) GO TO 530
IF(GO.EQ.6) GO TO 560
GO TO 310
320 WORD=IN(WORDST)
IF(L.EN.6) GO TO (324,366) REM

```

```

READNUM 202
READNUM 203
READNUM 204
READNUM 205
READNUM 206
READNUM 207
READNUM 208
READNUM 209
READNUM 210
READNUM 211
READNUM 212
READNUM 213
READNUM 214
READNUM 215
READNUM 216
READNUM 217
READNUM 218
READNUM 219
READNUM 220
READNUM 221
READNUM 222
READNUM 223
READNUM 224
READNUM 225
READNUM 226
READNUM 227
READNUM 228
READNUM 229
READNUM 230
READNUM 231
READNUM 232
READNUM 233
READNUM 234
READNUM 235
READNUM 236

```

```

00 322 J=1,L
322 WORD=OR(SHIFT(WORD,6),IN(WORDST+J))
IF(REW.EQ.2) GO TO 366
324 DO 325 J=1,5
IF(AND(SHIFT(COMMAND(J),6*(L+1)),AM(L+1)).EQ.WORD) GO TO 327
325 CONTINUE
GO TO 330
327 IF(IN(K).EQ.1R/ .AND. J.NE.5) GO TO 540
GO TO (330,340,350,360,370) J
330 IF(REAL) RETURN ABORT
DO 335 J=1,N
DO 335 JJ=1,M
335 IF(ABS(MATRIX(J,JJ)).GT.1E14) MATRIX(J,JJ)=0.0
CALL SURINT(MATRIX,MATRIX,N,M,ND)
RETURN ABORT
340 DO 345 J=1,N
DO 345 JJ=1,M
345 MATRIX(J,JJ)=0.0
END=.FALSE.
IROW=ICOL=POINTER=1
PRINT(NSP,399)AND(NAME,AM(1)),NAME
KK=K
GO TO 10
350 END=.TRUE.
GO TO 600
360 K=K+2
NUM=0
IF(IN(K-1).EQ.28) NUM=1
IF(IN(K-1).EQ.29) NUM=2
IF(IN(K-1).EQ.30) NUM=3
IF(IN(K-1).EQ.32) NUM=5
IF(NUM.EQ.0) GO TO 540
IF(IN(K).NE.45 .AND. IN(K).NE.46) GO TO 540
IF(IN(K+1).EQ.1RR) GO TO 365
364 INUNIT=NUM

```

```

READNUM 237
READNUM 238
READNUM 239
READNUM 240
READNUM 241
READNUM 242
READNUM 243
READNUM 244
READNUM 245
READNUM 246
READNUM 247
READNUM 248
READNUM 249
READNUM 250
READNUM 251
READNUM 252
READNUM 253
READNUM 254
READNUM 255
READNUM 256
READNUM 257
READNUM 258
READNUM 259
READNUM 260
READNUM 261
READNUM 262
READNUM 263
READNUM 264
READNUM 265
READNUM 266
READNUM 267
READNUM 268
READNUM 269
READNUM 270
READNUM 271

```



```

365 GO TO 2
    REV=2
    K=K+1
    GO TO 305
366 IF(AND(SHIFT(COMMAND(6),6*(L+1)),AM(L+1)).NE.WORD) GO TO 530
    REWIND NUM
    GO TO 364
370 IF(IN(K).NE.1R/ .AND. IN(K+1).NE.1R/) GO TO 372
    IF(IN(K).NE.1R/) K=K+1
    LIST=.TRUE.
    GO TO 400
372 SLASH(1)=SLASH(2)=0
375 LIST=.FALSE.
    IF(SLASH(2).GT.M .OR. SLASH(1).GT.NN) GO TO 550
    IF(SLASH(2).NE.0 .AND. SLASH(1).GT.N) GO TO 550
    IF(M.EQ.1 .AND.N.EQ.1) GO TO 380
    IF(M.EQ.1) GO TO 381
    IF(N.EQ.1) GO TO 383
    IF(SLASH(1).EQ.0 .AND. SLASH(2).EQ.0) GO TO 388
    IF(SLASH(1).EQ.0) GO TO 386
    IF(SLASH(2).EQ.0) GO TO 387
    PRINT (NSP,390)AND(NAME,AM(1)),NAME,SLASH(1),SLASH(2),MATRIX(SLASH(
    ^1),SLASH(2))
377 KK=K+1
    END=.FALSE.
    GO TO 10
380 PRINT (NSP,398)AND(NAME,AM(1)),NAME,MATRIX(1,1)
    GO TO 377
381 IF(SLASH(1).EQ.0) GO TO 382
    PRINT (NSP,397)AND(NAME,AM(1)),NAME,SLASH(1),MATRIX(SLASH(1),1)
    GO TO 377
382 J=1
    PRINT (NSP,396)AND(NAME,AM(1)),NAME,J
    PRINT (NSP,395) (MATRIX(JJ,J),JJ=1,N)
    PRINT (NSP,*)

```

```

READNUM 272
READNUM 273
READNUM 274
READNUM 275
READNUM 276
READNUM 277
READNUM 278
READNUM 279
READNUM 280
READNUM 281
READNUM 282
READNUM 283
READNUM 284
READNUM 285
READNUM 286
READNUM 287
READNUM 288
READNUM 289
READNUM 290
READNUM 291
READNUM 292
READNUM 293
READNUM 294
READNUM 295
READNUM 295
READNUM 297
READNUM 298
READNUM 299
READNUM 300
READNUM 301
READNUM 302
READNUM 303
READNUM 304
READNUM 305
READNUM 306

```

307 READNUM
 308 READNUM
 309 READNUM
 310 READNUM
 311 READNUM
 312 READNUM
 313 READNUM
 314 READNUM
 315 READNUM
 316 READNUM
 317 READNUM
 318 READNUM
 319 READNUM
 320 READNUM
 321 READNUM
 322 READNUM
 323 READNUM
 324 READNUM
 325 READNUM
 326 READNUM
 327 READNUM
 328 READNUM
 329 READNUM
 330 READNUM
 331 READNUM
 332 READNUM
 333 READNUM
 334 READNUM
 335 READNUM
 336 READNUM
 337 READNUM
 338 READNUM
 339 READNUM
 340 READNUM
 341 READNUM

```

    GO TO 377
383 IF(SLASH(2).EQ.0 .AND. SLASH(1).GT.1) GO TO 384
    IF(SLASH(2).EQ.0) GO TO 385
    PRINT(DSP,397)AND(NAME,AM(1)),NAME,SLASH(2),MATRIX(1,SLASH(2))
    GO TO 377
384 PRINT(DSP,397)AND(NAME,AM(1)),NAME,SLASH(1),MATRIX(1,SLASH(1))
    GO TO 377
385 J=1
    PRINT(DSP,394)AND(NAME,AM(1)),NAME,J
    PRINT(DSP,393)(MATRIX(J,J),JJ=1,M)
    PRINT(DSP,*)
    GO TO 377
386 PRINT(DSP,396)AND(NAME,AM(1)),NAME,SLASH(2)
    PRINT(DSP,395)(MATRIX(J,SLASH(2)),J=1,N)
    PRINT(DSP,*)
    GO TO 377
387 PRINT(DSP,394)AND(NAME,AM(1)),NAME,SLASH(1)
    PRINT(DSP,393)(MATRIX(SLASH(1),J),J=1,M)
    PRINT(DSP,*)
    GO TO 377
388 PRINT(DSP,392)AND(NAME,AM(1)),NAME
    DO 389 J=1,N
389 PRINT(DSP,391)J,(MATRIX(J,J),JJ=1,M)
    PRINT(DSP,*)
    GO TO 377
390 FORMAT(5X,A=,"(",I3,"",I3,"") = ",G20.14,/)
391 FORMAT(/1X,I3,"",G15.8,3(2X,G15.8),/,4(2X,G15.8))
392 FORMAT(25X,A=)
393 FORMAT(2X,4(3X,G15.8))
394 FORMAT(5X,A=,"",ROW",I3,/)
395 FORMAT(5X,G15.8)
396 FORMAT(5X,A=,"",COLUMN",I3,/)
397 FORMAT(5X,A=,"(",I3,"") = ",G20.14,/)
398 FORMAT(5X,A=,"",G20.14,/)
399 FORMAT(/,30X,A=," SET TO ZERO",/)
  
```


READNUM 342
 READNUM 343
 READNUM 344
 READNUM 345
 READNUM 346
 READNUM 347
 READNUM 348
 READNUM 349
 READNUM 350
 READNUM 351
 READNUM 352
 READNUM 353
 READNUM 354
 READNUM 355
 READNUM 356
 READNUM 357
 READNUM 358
 READNUM 359
 READNUM 360
 READNUM 361
 READNUM 362
 READNUM 363
 READNUM 364
 READNUM 365
 READNUM 366
 READNUM 367
 READNUM 368
 READNUM 369
 READNUM 370
 READNUM 371
 READNUM 372
 READNUM 373
 READNUM 374
 READNUM 375
 READNUM 376

C*****
 C DECODES INDECIES WITHIN SLASHES
 C*****
 400 SLASH(1)=SLASH(2)=0
 I=5
 K=K+1
 GO=AND(SHIFT(A(IN(K)+2),-3*I),AM1)
 GO TO (410,420,430,440,520,560,300) GO
 410 PET=1
 GO TO 480
 411 SLASH(1)=NUM
 IF(IN(K).EQ.1R/) GO TO 440
 K=K+1
 IF(IN(K).EQ.1R/) GO TO 440
 IF(IN(K).EQ.45 .AND. IN(K-1).NE.46) GO TO 540
 IF(IN(K).EQ.45) GO TO 415
 IF(IN(K).EQ.1R\$) GO TO 300
 IF(AND(SHIFT(A(IN(K)+2),-3*I),AM1).NE.1) GO TO 540
 RET=2
 GO TO 480
 412 SLASH(2)=NUM
 K=K-1
 415 K=K+1
 IF(IN(K).EQ.1R/) GO TO 440
 IF(IN(K).EQ.1R\$) GO TO 300
 GO TO 540
 420 K=K+1
 IF(IN(K).EQ.1R/) GO TO 440
 IF(IN(K).NE.46) GO TO 435
 430 K=K+1
 435 IF(IN(K).EQ.1R\$) GO TO 300
 IF(IN(K).EQ.-1) GO TO 560
 IF(AND(SHIFT(A(IN(K)+2),-3*I),AM1).NE.1) GO TO 540
 RET=2
 GO TO 480


```

IF(IN(K).NE.45 .AND. IN(K).NE.46 .AND. IN(K).NE. 1R/) GO TO 520
L=K-1
REWIND 7
WRITE(7,99)(IN(J),J=NUMST,L)
REWIND 7
RFAD(7,*)TEMP
IF(TEMP.LT.0.0) GO TO 550
NUM=IFIX(TEMP)
GO TO (411,412) RET
497 FORMAT(/,30X,A=," IS A NONDIMENSIONED VARIABLE",/)
498 FORMAT(/,30X,"NEXT ENTRY GOES INTO",/,30X,A=,"(",I3,"")"/)
499 FORMAT(/,30X,"NEXT ENTRY GOES INTO",/,30X,A=,"(",I3,"",I3,"")"/)
C*****
C      ERROR PRINTING SECTION
C*****
500 LL=KK-1
L1=K-KK
PRINT(0SP,599)(IN(J),J=1,72)
PRINT(0SP,596)LL,L1
PRINT(0SP,590)
505 IF(LL.F0.0) GO TO 510
PRINT(0SP,599)(IN(J),J=1,LL),COMMA
GO TO 515
510 PRINT(0SP,*)
515 PRINT(0SP,598)KK
GO TO 575
520 LL=KK-1
L=K-1
PRINT(0SP,599)(IN(J),J=1,72)
PRINT(0SP,597)L
PRINT(0SP,591)
GO TO 505
530 LL=KK-1
L1=K-KK
PRINT(0SP,599)(IN(J),J=1,72)

```

```

READNUM 412
READNUM 413
READNUM 414
READNUM 415
READNUM 416
READNUM 417
READNUM 418
READNUM 419
READNUM 420
READNUM 421
READNUM 422
READNUM 423
READNUM 424
READNUM 425
READNUM 426
READNUM 427
READNUM 428
READNUM 429
READNUM 430
READNUM 431
READNUM 432
READNUM 433
READNUM 434
READNUM 435
READNUM 436
READNUM 437
READNUM 438
READNUM 439
READNUM 440
READNUM 441
READNUM 442
READNUM 443
READNUM 444
READNUM 445
READNUM 446

```

447 READNUM
 448 READNUM
 449 READNUM
 450 READNUM
 451 READNUM
 452 READNUM
 453 READNUM
 454 READNUM
 455 READNUM
 456 READNUM
 457 READNUM
 458 READNUM
 459 READNUM
 460 READNUM
 461 READNUM
 462 READNUM
 463 READNUM
 464 READNUM
 465 READNUM
 466 READNUM
 467 READNUM
 468 READNUM
 469 READNUM
 470 READNUM
 471 READNUM
 472 READNUM
 473 READNUM
 474 READNUM
 475 READNUM
 476 READNUM
 477 READNUM
 478 READNUM
 479 READNUM
 480 READNUM
 481 READNUM

```

    PRINT(OSP,596)LL,L1
    PRINT(OSP,592)
    GO TO 505
540 LL=KK-1
    L1=K-KK
    PRINT(OSP,599) (IN(J),J=1,72)
    PRINT(OSP,596)LL,L1
    PRINT(OSP,593)
    GO TO 505
550 LL=KK-1
    L1=K-KK
    PRINT(OSP,599) (IN(J),J=1,72)
    PRINT(OSP,596)LL,L1
    PRINT(OSP,594)
    GO TO 505
560 LL=KK-1
    L1=K-KK
    PRINT(OSP,599) (IN(J),J=1,72)
    PRINT(OSP,596)LL,L1
    PRINT(OSP,595)
    PRINT(OSP,599) (IN(J),J=1,LL),COMMA
    PRINT(OSP,*)
575 INUNIT=5
    IF(BATCH) RETURN ABORT
    LIST=.FALSE.
    II=1
    GO TO 2
590 FORMAT(/,30X,"ILL FORMATED NUMBER--PLEASE RETYPE FROM:",//)
591 FORMAT(/,30X,"ILLEGAL CHARACTER--PLEASE RETYPE FROM:",//)
592 FORMAT(/,30X,"COMMAND UNRECOGNIZED--PLEASE RETYPE FROM:",//)
593 FORMAT(/,30X,"FORMAT ERROR--PLEASE RETYPE FROM:",//)
594 FORMAT(/,3JX,"INDICE OUT OF RANGE--PLEASE RETYPE FROM:",//)
595 FORMAT(/,30X,"MORE THAN 72 CHARACTERS--RETYPE ALL AFTER:",//)
596 FORMAT(1X,=X,=("^^"))
597 FORMAT(1X,=X,="^^")
  
```



```

598 FORMAT(" ",X)
599 FORMAT(1X,72R1)
C*****
C      TERMINATION OF READNUM OPERATION
C*****
600 IF(END) GO TO 650
    GO TO 2
650 IF(REAL) RETURN
    DO 660 J=1,N
    DO 660 JJ=1,M
    IF(ABS(MATRIX(J,JJ)).GT.1E14) GO TO 670
660 CONTINUE
    CALL SUBINT(MATRIX,MATRIX,N,M,ND)
    RETURN
670 PRINT(NSP,699)AND(NAME,AM(1)),NAME,J,JJ,MATRIX(J,JJ)
    IRCW=J
    ICOL=JJ
    POINTEP=(IROW-1)*M+ICOL
    GO TO 575
699 FORMAT(30X,A=" IS INTEGER MODE:",/30X,"ELEMENT(",I3,"",I3,
    ^") = ",G16.10,/30X,"IS TOO LARGE--PLEASE REENTER",/)
C*****
C      GLITCH IN PROGRAM
C*****
700 PRINT(NSP,799)
    RETURN ABORT
799 FORMAT(72(" ",/3X,"PROGRAM IMPROPERLY LOADED--TERMINATE AND RELO
    ^AD",/72(" ",/
    END

```

```

READNUM 482
READNUM 483
READNUM 484
READNUM 485
READNUM 486
READNUM 487
READNUM 488
READNUM 489
READNUM 490
READNUM 491
READNUM 492
READNUM 493
READNUM 494
READNUM 495
READNUM 496
READNUM 497
READNUM 498
READNUM 499
READNUM 500
READNUM 501
READNUM 502
READNUM 503
READNUM 504
READNUM 505
READNUM 506
READNUM 507
READNUM 508
READNUM 509
READNUM 510

```

```

SUBROUTINE SUBINT(DAT,IOAT,N,H,NO)
DIMENSION DAT(ND,H),IOAT(ND,H)
DO 10 J=1,N
DO 10 JJ=1,H
IOAT(J,JJ)=IFIX(DAT(J,JJ))
10 CONTINUE
RETURN
ENTRY SUBREL
DO 20 J=1,N
DO 20 JJ=1,H
20 DAT(J,JJ)=FLOAT(IOAT(J,JJ))
RETURN
END

```

```

READNUM 511
READNUM 512
READNUM 513
READNUM 514
READNUM 515
READNUM 516
READNUM 517
READNUM 518
READNUM 519
READNUM 520
READNUM 521
READNUM 522
READNUM 523

```



```

SUBROUTINE READCOM(COM,NUM)
INTEGER COM(NUM),IN(80),GO,ANS,DSP,AM(10),DUP
COMMON/CONTROL/DSP,RATCH
COMMON/LISTING/NL,LIST(1)
LOGICAL COMMA,BATCH
DATA AM/777R,7777R,7777778,777777778,777777777778,
^777777777777R,777777777777778,77777777777777778,
^777777777777777777R,77777777777777777777778/
5 DO 16 I=1,NUM
10 COM(I)=0
J=0
COMMA=.FALSE.
20 READ 90,(IN(K),K=1,80)
KK=1
GO=0
DO 40 K=1,80
IF(IN(K).EQ.1R$) GO TO 86
IF(IN(K).EQ.1R,) GO=1
IF(IN(K).EQ.1R ) GO=2
IF(IN(K).EQ.1R/) GO=3
IF(GO.EQ.0) GO TO 40
LL=K-KK
IF(LL.GT.9) GO TO 50
IF(LL.FQ.0.AND.GO.FQ.3) COMMA=.FALSE.
IF(LL.EQ.0) GO TO 35
IF(GO.NE.2) GO TO 25
IF(IN(K+1).NE.1R ) GO TO 36
25 J=J+1
IF(J.GT.NUM) GO TO 55
COM(J)=0
DO 30 L=1,LL
30 COM(J)=OR(SHIFT(COM(J),6),IN(KK+L-1))
IF(COM(J).NE.3R**) GO TO 31
J=J-1
GO TO 20

```

```

READCOM 2
READCOM 67
READCOM 68
READCOM 69
READCOM 70
READCOM 71
READCOM 72
READCOM 73
READCOM 74
READCOM 75
READCOM 76
READCOM 77
READCOM 78
READCOM 79
READCOM 80
READCOM 81
READCOM 82
READCOM 83
READCOM 84
READCOM 85
READCOM 86
READCOM 87
READCOM 88
READCOM 89
READCOM 90
READCOM 91
READCOM 92
READCOM 93
READCOM 94
READCOM 95
READCOM 96
READCOM 97
READCOM 98
READCOM 99
READCOM 100

```

```

31 CONTINUE
  DUP=0
  DO 32 I=1,ML
    IF(COM(J).NE.AND(SHIFT(LIST(I),LL*6),AH(LL))) GO TO 32
    DUP=DUP+1
    IF(DUP.GT.1) GO TO 60
    ICOM=I
32 CONTINUE
    IF(DUP.EQ.0) GO TO 65
    COM(J)=ICOM
    IF(COM=I) COM(J)=~ICOM
    COM=I
    IF(GO.FQ.1) COM=I.TRUE.
35 KK=K+1
36 GO=0
40 CONTINUE
    IF(KK.NE.81) GO TO 70
    NUM=J
    RETURN
3*****ERROR MORE THAN 9 CHARACTERS IN COMMAND NAME**
50 K=KK+9
  PRINT(DSP,98)(IN(L),L=KK,K)
  GO TO 75
C*****ERROR TOO MANY COMMAND NAMES ENTERED**
55 K=K-1
56 PRINT(DSP,97)NUM,LL,(IN(L),L=KK,K)
  PRINT(DSP,96)
  IF(RATCH) GO TO 88
  READ(5,99)ANS
  IF(ANS.EQ.1R) RETURN
  IF(ANS.EQ.1R$) GO TO 88
  IF(ANS.NE.1RN) GO TO 56
  PRINT(DSP,95)
  GO TO 5
C*****ERROR NON UNIQUE ABBREVIATION USED**

```

```

READCOM 101
READCOM 102
READCOM 103
READCOM 104
READCOM 105
READCOM 106
READCOM 107
READCOM 108
READCOM 109
READCOM 110
READCOM 111
READCOM 112
READCOM 113
READCOM 114
READCOM 115
READCOM 116
READCOM 117
READCOM 118
READCOM 119
READCOM 120
READCOM 121
READCOM 122
READCOM 123
READCOM 124
READCOM 125
READCOM 126
READCOM 127
READCOM 128
READCOM 129
READCOM 130
READCOM 131
READCOM 132
READCOM 133
READCOM 134
READCOM 135

```



```

60 J=J-1
K=K-1
PRINT (DSP,94)LL, (IN(L),L=KK,K),AND(LIST(ICOM),AM(1)),LIST(ICOM),
1 AND(LIST(I),AM(1)),LIST(I)
GO TO 75
C*****ERROR COMMAND NAME UNRECOGNIZED**
65 J=J-1
K=K-1
PRINT (DSP,93)LL, (IN(L),L=KK,K)
GO TO 75
C*****ERROR INCOMPLETE COMMAND DUE INPUT CHARACTER COUNT LIMITATION**
70 PRINT (DSP,92)
75 IF (J.EQ.0) GO TO 80
ICOM=IARS(COM(J))
PRINT (DSP,91)AND(LIST(ICOM),AM(1)),LIST(ICOM)
GO TO 85
80 PRINT (DSP,95)
85 IF (BATCH) GO TO 88
GO TO 20
C*****ABORT DUE TO $ OR ERROR IN BATCH MODE**
88 NUM=COM(1)=0
RETURN
91 FORMAT(/5X,"PLEASE RETYPE ALL AFTER - ",A=,/ )
92 FORMAT(/10X,"COMMAND TRUNCATED DUE TO CHARACTER COUNT.")
93 FORMAT(/1X,=R1," - IS AN INVALID COMMAND.")
94 FORMAT(/1X,=R1," IS A NON UNIQUE ABBREVIATION ("",A=,
1 " " OR ",A=,"???").")
95 FORMAT(/5X,"PLEASE REENTER COMMAND >>")
96 FORMAT(20X,"TYPE YES -- TO PROCESS CURRENT COMMANDS",/,
1 27X,"NO -- TO REENTER ALL COMMANDS >>")
97 FORMAT(/10X,"MORE THAN ",I2," COMMANDS ENTERED: ",=R1,
1 " AND ALL AFTER",/,10X,"WILL BE IGNORED.")
98 FORMAT(/1X,10R1,"... IS ILLFGL (MORE THAN 9 CHARACTERS).")
99 FORMAT(80R1)
END

```

```

READCOM 136
READCOM 137
READCOM 138
READCOM 139
READCOM 140
READCOM 141
READCOM 142
READCOM 143
READCOM 144
READCOM 145
READCOM 146
READCOM 147
READCOM 148
READCOM 149
READCOM 150
READCOM 151
READCOM 152
READCOM 153
READCOM 154
READCOM 155
READCOM 156
READCOM 157
READCOM 158
READCOM 159
READCOM 160
READCOM 161
READCOM 162
READCOM 163
READCOM 164
READCOM 165
READCOM 166
READCOM 167
READCOM 168
READCOM 169
READCOM 170

```



```

SURROUTINE LISTER(NAME,VAR,N,M,ND,FCR,DST)
COMMON/FORMS/NOREP,NOFW,NONSD,NOREP,NOFW,NONSD
COMMON/CONTROL/DSP
INTEGER DST,OSP
DIMENSION VAR(ND,M)
NUM=NOREP
IFW1=NOFW
NSD1=NONSD
IF(DST.NE.OSP) GO TO 10
NUM=NOREP
IFW1=NOFW
NSD1=NONSD
10 CONTINUE
IF(M.EQ.1.AND. N.EQ.1) GO TO 100
IF(M.EQ.1) GO TO 200
IF(N.EQ.1) GO TO 300
GO TO 400
100 PRINT(DST,110)AND(NAME,778),NAME,FOR,IFW1,NSD1,VAR(N,M)
GO TO 500
110 FORMAT(1P,/,10X,A," = ",V=.,/)
200 PRINT(DST,210)AND(NAME,778),NAME
PRINT(DST,211)((FOR,IFW1,NSD1,VAR(J,1)),J=1,N)
GO TO 500
210 FORMAT(1P,/,10X,A=.,/)
211 FORMAT(1P,6X,V=.=)
300 PRINT(DST,310)AND(NAME,778),NAME
MM=M
IF(M.GT.NUM) MM=NUM
PRINT(DST,311)MM,((FOR,IFW1,NSD1,VAR(1,J)),J=1,MM)
IF(M.GT.NUM) GO TO 406
GO TO 500
310 FORMAT(1P,/,10X,A=.,/)
311 FORMAT(1P,4X,=(1X,V=.=))
400 PRINT(DST,410)AND(NAME,778),NAME
MM=M

```

[illegible]

2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36


```

SUBROUTINE FORMOUT
COMMON/CONTROL/DSP,BATCH
LOGICAL BATCH,STP
INTEGER A(61,15)
REWIND 6
STP=.FALSE.
ICOUNT=0
K=1
READ(6,10)(A(K,I),I=1,15)
K=K+1
5 READ(6,10)(A(K,I),I=1,15)
10 FOPMAT(1R1,13A10,1R1)
IF(EOF(6).NE.0.0) GO TO 50
IF(A(K,1).EQ.1RN .OR. A(K,1).EQ.1R1) GO TO 20
IF(K.EQ.61) GO TO 20
K=K+1
GO TO 5
20 K=K-1
IF(ICOUNT+K.GT.59) A(1,1)=1R1
IF(A(1,1).EQ.1R1)ICOUNT=0
NO 30 J=1,K
30 PRINT(99,10)(A(I,J),J=1,15)
NO 40 J=1,15
40 A(1,I)=A(K+1,I)
ICOUNT=ICOUNT+K
K=2
IF(STP) GO TO 55
GO TO 5
50 STP=.TRUE.
GO TO 20
55 CALL CONNOC(6LOUTPUT)
IF(.NOT.BATCH) STOP
REWIND 99
REWIND 6
100 READ(99,10)(A(1,I),I=1,15)

```


PRINT (A(1,1), I=1, 15)
IF (END) GOTO 100
GOTO 100

FORMOUT
FORMOUT
FORMOUT
FORMOUT

37
38
39
40

2	3	4	5	6	7	8	9	10	11	12	13	14	15	15	17	18	19	20	21	22	23
RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN	RETURN

IDENT	RETURN
ENTRY	RETURN
EXT	SYS=
BSSZ	1
SB1	1
MX4	42
SA2	X1
SA3	2
ZR	X3, RETURN
MX5	X3+X4
BX5	X2-X5
ZR	X5, GOTIT
SA3	A3+R1
EO	LOOP2
BX3	-X4+X3
SA3	X3+2
SA4	UNLOAD
BX6	X3+X4
SA6	A3
EO	RETURN
VFN	28/0, 2/3, 30/0
END	

RETURN

LOOP2

GOTIT

UNLOAD


```

SUBROUTINE ADDMAT(A, NAD, N, M, 9, NBD, C, NCD)
DIMENSION A(NAD, M), B(NBD, M), C(NCD, M)
LOGICAL SUB
SUR=.FALSE.
5 DO 10 J=1, M
  DO 10 I=1, N
    A(I, J)=B(I, J)+C(I, J)
  IF(SUB) A(I, J)=B(I, J)-C(I, J)
10 CONTINUE
RETURN
ENTRY SURMAT
SUR=.TRUE.
GO TO 5
END

```

```

MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX

```

```

2
3
4
5
6
7
8
9
10
11
12
13
14
15

```



```

SUBROUTINE INVERT(A,NAD,B,NBD,N,IFAIL)
DIMENSION A(NAD,N),B(NBD,N)
COMMON/CONTROL/DSP
LOGICAL IFAIL
INTEGER DSP
CALL IDENT(A,NAD,N)
35 K=0
TEMP=0.0
DO 40 I=J,N
IF(TEMP.GT.ABS(B(I,J))) GO TO 40
TEMP=ABS(B(I,J))
K=I
40 CONTINUE
IF(TEMP.EQ.0.0) GO TO 90
IF(K.EQ.J) GO TO 50
DO 45 J=1,N
TEMP=R(K,I)
B(K,I)=R(J,I)
B(J,I)=TEMP
TEMP=A(K,I)
A(K,I)=A(J,I)
A(J,I)=TEMP
45 A(J,I)=TEMP
50 IF(B(J,J).EQ.1.0) GO TO 60
TEMP=R(J,J)
DO 55 I=1,N
R(J,I)=R(J,I)/TEMP
55 A(J,I)=A(J,I)/TEMP
60 DO 70 I=1,N
IF(I.EQ.J) GO TO 70
IF(B(I,J).EQ.0.0) GO TO 70
TEMP=R(I,J)
DO 65 Y=1,N
B(I,K)=B(I,K)-TEMP*B(J,K)
65 A(I,K)=A(I,K)-TEMP*A(J,K)
70 CONTINUE

```

```

J=J+1
IF(J.LE.N) GO TO 35
IFAIL=.FALSE.
RETURN
90 IFAIL=.TRUE.
PRINT(=SP,*)" SUBROUTINE INVERT -- MATRIX IS SINGULAR"
RETURN
END

```

```

MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
MATRIX
106
107
108
109
110
111
112
113

```


Appendix B

INTERAC User's Guide

INTERAC is an interactive software package for synthesizing a discrete state-variable feedback gain matrix to control a multi-input, multi-output continuous plant described by Eq (1).

$$\begin{aligned}\dot{\underline{x}}(t) &= \underline{A} \underline{x}(t) + \underline{B} \underline{u}(t) + \underline{R} \underline{d}(t) \\ \underline{y}(t) &= \underline{C} \underline{x}(t)\end{aligned}\tag{1}$$

where $\underline{A}$ is an N by N state matrix,
 $\underline{B}$ is an N by M control matrix,
 $\underline{R}$ is an N by NDD disturbance matrix, and
 $\underline{C}$ is an (NCC or NDD) by N output matrix.

and $\underline{y}(t)$ defines the NDD outputs that are to reject disturbances or the NCC outputs that are to track input commands.

Three types of problems can be examined: (1) the regulator problem, (2) the disturbance rejector problem, and (3) the tracking problem. The computer input of the problem type and the plant description matrices can be accomplished directly by the user or in response to requests from the computer. 23 alphanumeric options are used in response to the prompt "COMMAND >>" to control data input, data output, and program sequence control.

Commands

To execute the program for the first time, type FORTRAC. All further input data will be requested by the computer. To terminate the program type STOP in response to the prompt "COMMAND >>".

For the do-it-yourself user the following commands are available for complete input, output, and program sequence control.

(1) ENTER [, parameters] This command allows the user to specify the dimensions of a matrix and then to enter the values for the matrix elements. The parameters are variable names and their use is optional. If the variable names are not included they will be requested by the computer. The legal variable names are described under options 16 through 21.

(2) STOP This command terminates the program and prints a termination message informing the user of the contents of the files ANSWER and DATA that are left by the program.

(3) END This command has two functions. First in response to the prompt "COMMAND >>", the program is terminated, but the termination message is not printed. END is also used to terminate sub-options such as when the command ENTER is given without a parameter list.

(4) PRINT [, parameters] This command causes the current values of the variables identified in the parameter list to be printed at the terminal and to be written to the file ANSWER. The parameters are optional, and if not used the variable names will be requested by the computer.

(5) DFORMAT This command allows the user to specify the number of significant digits to be used for numbers printed at the terminal.

(6) CHANGE [, parameters] This command allows the user to change the values stored in a variable. The dimensions of matrices cannot be changed with this command. The parameters are variable names and their use is optional. If the parameters are not included the

names will be requested by the computer.

(7) DISPLAY [, parameters] This is a dual function command. With the parameters (variable names), the variables to be printed at the terminal are specified. Without the parameters, the variables already designated to be printed at the terminal are identified and the names are printed. The values of the variables on the display list are printed at the terminal whenever the variable is first encountered or when its value is changed in the control design algorithms.

(8) OUTPUT [, parameters] This command is used the same as DISPLAY, except OUTPUT refers to the variables to be written to the file ANSWER.

(9) DELETE [, parameters] This command deletes the named variables (parameters) from both the display and output lists. If no parameters are given, then all variables are removed from both lists.

(10) OPTIONS This command causes a list of the valid options to be printed at the terminal.

(11) VARIABLES This command causes a list of the legal variable names to be printed at the terminal.

(12) SAVE This command causes the current contents of all variables, all matrix dimensions, and the display/output list to be saved on the file DATA. The date and time of the save is printed, and they can be used to identify the data set when it is reloaded with the command RESTART.

(13) RESTART [, REWIND] This command causes the values saved on the file DATA to be reloaded into memory. The optional parameter REWIND is used to rewind the file before reading from it.

The file DATA is a sequential file, and thus RESTART may have to be used several times to get the desired data set if SAVE was used more than once. The date and time that the data set was saved is printed to identify the set that is reloaded.

(14) REWIND This command can be used by itself to rewind the file DATA or as a parameter for RESTART as shown above.

(15) RUN ,parameter This command causes the control design algorithms to be executed starting with the subroutine named in the parameter and terminating after calculating the closed-loop matrix. The valid parameters are FORTRAC, SAMPLE, AUGMAT, TRANSFORM, CONLAW, and CLOSELOOP. All necessary data is assumed to have been entered and all matrices are assumed properly dimensioned.

(16) FORTRAC This command by itself causes all necessary data to be requested and then all control design algorithms to be executed. When this command is used as a parameter for RUN, all control design algorithms are executed with the data currently in memory.

(17) SAMPLE This command causes the sampled-data system description to be computed from the continuous system description. The variables must be entered prior to issuing this command: AMATRIX - the state matrix, BMATRIX - the control matrix, DIMENSION - the array containing the plant dimensions (N, M, NCC, NDD), RMATRIX - the disturbance matrix only if NDD is non-zero, and TSAMPLE - the desired sampling time interval. Values for the following variables are calculated: CEIGEN - the continuous eigenvalues; MODAL - the modal matrix; INVMODAL - the inverse modal matrix; and the discrete system matrices FMATRIX - the state matrix, GMATRIX - the control

matrix, DRMATRIX - the disturbance matrix if NDD is non-zero, and DEIGN - the z-plane eigenvalues.

(18) AUGMAT This command causes the sampled-data system description to be augmented with discrete integrators for disturbance rejection or tracking problems. These variables must be available in memory prior to issuing this command: FMATRIX, GMATRIX, DRMATRIX - if NDD is non-zero, CMATRIX - the output matrix if NDD or NCC is non-zero, DIMENSION, and INTEGRATE - an array containing the number of integrators desired for each NDD or NCC output. One integrator is used to track or reject a step input, two integrators for a ramp, etc. Values for the following variables are calculated: AFMATRIX - the augmented state matrix, AGMATRIX - the augmented control matrix, AEMATRIX - the augmented input command matrix, ARMATRIX - the augmented disturbance matrix, and ACMATRIX - the augmented output matrix.

(19) TRANSFORM This command causes the AF and AG matrices from the augmented system description to be transformed into the Brunovsky controllable canonical form. AFMATRIX, AGMATRIX, and DIMENSION must be available in memory prior to issuing this command. Values for the following variables are calculated: UREFORM - the upper row echelon form of the matrix [AG AF], APLFORM - the Aplevich form of the matrix [AG AF], BRNFORM - the Brunovsky form of the matrix [AG AF], BFMATRIX and BGMATRIX - the Brunovsky controllable canonical form of AFMATRIX and AGMATRIX respectively, TINVERSE - the inverse transformation matrix, and CINDICES - the controllability indices.

(20) CONLAW This command causes the state-variable feedback

gain matrix K to be calculated for the desired closed-loop eigenvalues. BFMATRIX, BGMATRIX, TINVERSE, DIMENSION, and DESEIGEN - the desired eigenvalues, must be available in memory prior to issuing this command. Values for the following variables are calculated: COFMATRIX - the coefficient matrix which will be the zero matrix unless non-zero desired eigenvalues are entered, and KMATRIX - the feedback gain matrix.

(21) CLOSELOOP This command is used to calculate the closed-loop state matrix and the closed-loop eigenvalues. The variables required in memory prior to issuing this command are: AFMATRIX, AGMATRIX, DIMENSION, and KMATRIX. CLMATRIX - the closed-loop matrix and CLEIGEN - the closed-loop eigenvalues are calculated.

(22) OFORMAT This command allows the user to specify the number of significant digits to be used for numbers written to the file ANSWER.

(23) SETDIMEN This command properly dimensions all matrices after the plant dimensions, in the array DIMENSION, have been entered.

(24) "variable name" When the variable name is typed for a command, the user is allowed to set the dimensions of the variable and to enter or change the values of the elements of the variable.

(25) DIMENSION This command opens the array DIMENSION and allows the user to enter N - the number of states, M - the number of controls, and NCC - the number of commands or NDD - the number of disturbances. The elements of DIMENSION are individually addressable by the variable names STATES, CONTROLS, COMMANDS, and DISTURBS.

The above commands are used in response to the prompt "COMMAND >>". If the command has parameters, the parameters must be separated by

AD-A053 446

AIR FORCE INST OF TECH WRIGHT-PATTERSON AFB OHIO SCH--ETC F/6 12/1
INTERAC - AN INTERACTIVE SOFTWARE PACKAGE FOR DIRECT DIGITAL CO--ETC(U)
DEC 77 J A COLGATE

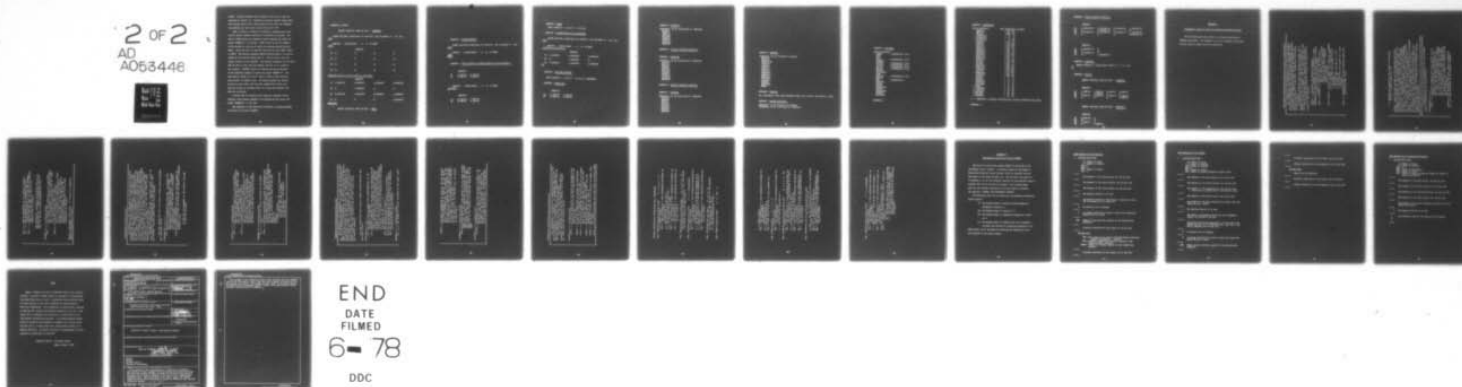
UNCLASSIFIED

AFIT/66C/EE/77D-1

NL

2 OF 2

AD
A053446



END

DATE

FILMED

6-78

DDC



commas. Multiple commands may be entered on one line if they are separated by slashes (/). Execution of entered commands begins after the carriage return, but to delay execution and enter more commands or parameters, the last entry on the line must be "***".

When a variable is opened for entering or changing data, there are six special commands available to facilitate the process. The first is ABORT which will terminate current commands and return the prompt "COMMAND>>" to the user. ZERO is used to set all elements of the variable to zero and is useful for entering sparse matrices. READ n allows the user to input the data from the file TAPE1, TAPE2, or TAPE3. The optional parameter REWIND following READ n, is used to rewind the file before reading from it. LIST is used to list the current contents of the variable. The optional parameter "/n,m/" after LIST is used to print only one element, one row, or one column of the variable. CONTINUE closes the variable and starts execution of any remaining commands or issues the prompt "COMMAND>>". The last special command is "/n,m/", which is used to reset the data entry pointer to element (n,m). All matrix elements are entered by rows in free format, and this last command can be used to go back and correct an erroneous entry or to skip over elements that need not be entered.

A dollar sign (\$) entered at any time will terminate current activity, cause pending commands to be ignored, and will return the prompt "COMMAND>>" to the user.

The remainder of this guide will be devoted to showing examples of the use of the above commands.

COMMAND >>ENTER

ENTER VARIABLE NAME OR END >>AMATRIX

ENTER DESIRED DIMENSIONS OF AMATRIX, MAX ALLOWED IS (20, 20).

4,4

AMATRIX DIMENSIONED (4, 4) IS OPEN

LIST

AMATRIX

11	0.	0.	0.	0.
21	0.	0.	0.	0.
31	0.	0.	0.	0.
41	0.	0.	0.	0.

1,2,3,4,/2,2/,1,/3,1/,4,3,2,1,/4,4/,5,L

AMATRIX

11	1.0000000	2.0000000	3.0000000	4.0000000
21	0.	1.0000000	0.	0.
31	4.0000000	3.0000000	2.0000000	1.0000000
41	0.	0.	0.	5.0000000

CONTINUE

ENTER VARIABLE NAME OR END >>END

COMMAND >>ENTER,AMATRIX

ENTER DESIRED DIMENSIONS OF AMATRIX, MAX ALLOWED IS (20,

2,2

AMATRIX DIMENSIONED (2, 2) IS OPEN

1,2

2,3

COMMAND >>PRINT,AMATRIX/CHANGE,AMATRIX/PRINT,AMATRIX

AMATRIX

11	1.00E+00	2.00E+00
21	2.00E+00	3.00E+00

AMATRIX DIMENSIONED (2, 2) IS OPEN

6,7,8,9

AMATRIX

11	6.00E+00	7.00E+00
21	8.00E+00	9.00E+00

COMMAND >>SAVE

DATA SAVED AT 12/13/77 17.15.37.

COMMAND >>ENTER,AMATRIX/RUN,FORTRAC

ENTER DESIRED DIMENSIONS OF AMATRIX, MAX ALLOWED IS (20, 20).

3,3

AMATRIX DIMENSIONED (3, 3) IS OPEN

1,2,3,0,1,2,3,2,2,L

AMATRIX

11	1.0000000	2.0000000	3.0000000
21	0.	1.0000000	0.
31	4.0000000	2.0000000	2.0000000

S

COMMAND >>RESTART,REWIND

DATA SAVED AT 12/13/77 17.15.37. RELOADED.

COMMAND >>PRINT,AM

AMATRIX

11	6.00E+00	7.00E+00
21	8.00E+00	9.00E+00

COMMAND >>DISPLAY

VARIABLES TO BE DISPLAYED AT TERMINAL

DEIGEN
FMATRIX
GMATRIX
CLEIGEN
KMATRIX
TSAMPLE
CINDICES
BRNFORM

COMMAND >>DISPLAY,AMATRIX,BMATRIX

COMMAND >>DISPLAY

VARIABLES TO BE DISPLAYED AT TERMINAL

AMATRIX
BMATRIX
DEIGEN
FMATRIX
GMATRIX
CLEIGEN
KMATRIX
TSAMPLE
CINDICES
BRNFORM

COMMAND >>DELETE,TSAMPLE,CINDICES

COMMAND >>DISPLAY

VARIABLES TO BE DISPLAYED AT TERMINAL

AMATRIX
BMATRIX
DEIGEN
FMATRIX
GMATRIX
CLEIGEN
KMATRIX
BRNFORM

COMMAND >>OUTPUT

VARIABLES TO BE PRINTED TO ANSWER

AMATRIX
BMATRIX
DIMENSION
RMATRIX
CMATRIX
FMATRIX
GMATRIX
AFMATRIX
AGMATRIX
AEMATRIX
ARMATRIX
ACMATRIX
CLEIGEN
KMATRIX
DESEIGEN
MODAL

COMMAND >>DELETE

ALL VARIABLES HAVE BEEN REMOVED FROM THE DISPLAY AND OUTPUT LISTS.

COMMAND >>OUTPUT/DISPLAY

VARIABLES TO BE PRINTED TO ANSWER
VARIABLES TO BE DISPLAYED AT TERMINAL

COMMAND >>OPTIONS

VALID COMMANDS

ENTER	[,PARAMETER LIST]
STOP	
END	
PRINT	[,PARAMETER LIST]
DFORMAT	
CHANGE	[,PARAMETER LIST]
DISPLAY	[,PARAMETER LIST]
OUTPUT	[,PARAMETER LIST]
DELETE	[,PARAMETER LIST]
OPTIONS	
VARIABLES	
SAVE	
RESTART	[,PARAMETER LIST]
REWIND	
RUN	,PARAMETER(S)
FORTRAC	
SAMPLE	
AUGMAT	
TRANSFORM	
CONLAW	
CLOSELOOP	
DFORMAT	
SETDIMEN	

COMMAND >>

COMMAND >> VARIABLES

VALID VARIABLES	MAX DIMENSION ALLOWED
AMATRIX	(20, 20)
BMATRIX	(20, 10)
DEIGEN	(20, 2)
DIMENSION	(1, 1)
◆ CFMATRIX	(0, 0)
INTEGRATE	(1, 10)
RMATRIX	(20, 5)
CMATRIX	(10, 20)
FMATRIX	(20, 20)
GMATRIX	(20, 10)
DEMATRIX	(20, 5)
DRMATRIX	(20, 5)
AFMATRIX	(20, 20)
AGMATRIX	(20, 10)
AEMATRIX	(20, 5)
ARMATRIX	(20, 5)
ACMATRIX	(10, 20)
CEIGEN	(20, 2)
CLEIGEN	(20, 2)
BFMATRIX	(20, 20)
BGMATRIX	(20, 10)
TINVERSE	(20, 20)
KMATRIX	(10, 20)
DESEIGEN	(20, 2)
CLMATRIX	(20, 20)
TSAMPLE	(1, 1)
STATES	(1, 1)
CONTROLS	(1, 1)
COMMANDS	(1, 1)
DISTURBS	(1, 1)
◆ MODAL	(0, 0)
◆ INVMODAL	(0, 0)
◆ CINDICES	(0, 0)
◆ UREFORM	(0, 0)
◆ APLFORM	(0, 0)
◆ BRNFORM	(0, 0)

◆ -- TEMPORARY VARIABLE PRINTED ONLY DURING PROGRAM EXECUTION.

COMMAND >>

COMMAND >>PRINT,AMATRIX,BMATRIX

AMATRIX

1]	1.00000E+00	2.00000E+00	3.00000E+00	4.00000E+00
2]	0.	1.00000E+00	0.	0.
3]	4.00000E+00	3.00000E+00	2.00000E+00	1.00000E+00
4]	0.	0.	0.	5.00000E+00

BMATRIX

1]	3.00000E+00	0.
2]	0.	0.
3]	1.00000E+00	0.
4]	0.	1.00000E+00

COMMAND >>DFORMAT

ENTER NUMBER OF SIGNIFICANT DIGITS 0 < N < 15.

3

COMMAND >>PRINT

ENTER VARIABLE NAME OR END >>AMATRIX

AMATRIX

1]	1.00E+00	2.00E+00	3.00E+00	4.00E+00
2]	0.	1.00E+00	0.	0.
3]	4.00E+00	3.00E+00	2.00E+00	1.00E+00
4]	0.	0.	0.	5.00E+00

ENTER VARIABLE NAME OR END >>BMATRIX

BMATRIX

1]	3.00E+00	0.
2]	0.	0.
3]	1.00E+00	0.
4]	0.	1.00E+00

Appendix C

Programmer's Guide for Using the Specialized INTERAC Software

The following pages give syntax for calling the specialized INTERAC subroutines. All elements in the call statement are defined and any required common blocks are identified.


```

C*****SUBROUTINE READNUM(NAME,MATRIX,N,M,N0,REAL), RETURNS(ABORT)*****
C
C      THE PURPOSE OF THIS SUBROUTINE IS TO ALLOW (NON,SINGLE,
C      OR DOUBLE DIMENSIONED AND REAL OR INTEGER) VARIABLES TO BE
C      FILLED WITH DATA, EXAMINED, AND/OR SELECTIVE ELEMENTS CHANGED.
C      ALL ENTRIES ARE CHECKED FOR VALID FORMAT AND REENTRY IS
C      REQUESTED IF VALIDITY IS NOT MET.  NUMBERS MAY BE ENTERED
C      IN FREE FORMAT.
C
C      DIMENSIONED VARIABLE DATA ARE ENTERED BY ROWS, IN 72
C      COLUMN CARD IMAGES ON AS MANY LINES AS ARE REQUIRED OR
C      DESIRED TO FILL ALL ELEMENTS.  THE TELL WILL RING EACH TIME
C      THE ROUTINE IS WAITING FOR A 72 COLUMN CARD IMAGE INPUT.
C      THE FOLLOWING ALPHANUMERIC COMMANDS ARE AVAILABLE TO
C      ASSIST IN THE PROCESS.
C
C      ABORT.....TERMINATE THE ROUTINE ABNORMALLY
C      ZERO.....SET ALL ELEMENTS OF THE VARIABLE TO ZERO
C      CONTINUE...ALL DATA DESIRED HAS BEEN ENTERED, EXECUTE
C               A NORMAL RETURN FROM THE ROUTINE
C      READ N (REWIND).....READ THE DATA FOR INPUT FROM
C                          TAPE(N), WHERE N = 1, 2, 3, OR 5.
C                          REWIND IS AN OPTIONAL PARAMETER
C                          CAUSING THE TAPE TO BE REWOUND
C                          PRIOR TO READING
C      LIST (/N,M/)... 'LIST' BY ITSELF WILL PRINT ALL ELEMENTS
C                      OF THE VARIABLE.  N & M CAN BE USED
C                      1) TO PRINT ONLY ONE ELEMENT (N & M
C                          SPECIFIED),
C                      2) TO PRINT ONLY ONE ROW (M = 0 OR
C                          MISSING), OR
C                      3) TO PRINT ONLY ONE COLUMN (N = 0 OR
C                          MISSING)
C      /N,M/.....SPECIFIES THAT THE NEXT NUMERIC ENTRY WILL
C                      GO INTO ELEMENT (N,M)

```

100

M.....COLUMN DIMENSION OF THE VARIABLE DESIRED.
 ND.....ROW DIMENSION OF THE VARIABLE FROM THE
 DIMENSION STATEMENT IN THE CALLING PROGRAM.
 REAL.....LOGICAL CONSTANT EQUAL TO TRUE IF THE VARIABLE
 MODE IS REAL AND EQUAL TO FALSE IF THE
 VARIABLE MODE IS INTEGER.

NON DIMENSIONED VARIABLES ARE ENTERED WITH N, M,
 AND ND EQUAL TO 1. SINGLE DIMENSIONED VARIABLES
 HAVE THE NORMALLY UNUSED DIMENSION SET TO 1.

COMMON-----LATELED CONTROL
 DSP.....THE INTEGER TAPE NUMBER OR A LEFT JUSTIFIED,
 ZERO FILLED HOLLERITH CONSTANT INDICATING WHICH
 FILE THE SUBROUTINE IS TO WRITE OUT ON. THIS
 FILE SHOULD BE A CONNECTED FILE.

BATCH.....A LOGICAL CONSTANT WITH A VALUE OF
 TRUE IF THIS ROUTINE IS BEING USED IN A BATCH
 JOB AND A VALUE OF FALSE IF USED FOR AN
 INTERACTIVE JOB. (IF ANY FORMAT ERROR IS
 DETECTED WHEN RUNNING A BATCH JOB THE
 DIAGNOSTIC IS PRINTED AND THEN AN ABORT
 TERMINATION IS EXECUTED)

TAPES IS THE NORMAL INPUT FILE. IT SHOULD BE CONNECTED
 AND/OR EQUIVALENCED TO INPUT.

```

COMMON-----Labeled control
DSP.....THE INTEGER TAPE NUMBER OR A LEFT JUSTIFIED,
ZERO FILLED HOLLERITH CONSTANT INDICATING WHICH
FILE THE SURROUTINE IS TO WRITE OUT ON. THIS
FILE SHOULD BE A CONNECTED FILE.

BATCH.....A LOGICAL CONSTANT WITH A VALUE OF
TRUE IF THIS ROUTINE IS BEING USED IN A BATCH
JOB AND A VALUE OF FALSE IF USED FOR AN
INTERACTIVE JOB. (IF ANY VALIDITY ERROR IS
DETECTED WHEN RUNNING A BATCH JOB THE
DIAGNOSTIC IS PRINTED AND THEN COM(1) AND
NUM ARE SET TO ZERO AND A RETURN IS EXECUTED.

COMMON-----Labeled listing
NL.....THE NUMBER OF ACTIVE ELEMENTS IN THE ARRAY
LIST.....AN ARRAY DIMENSIONED AT LEAST NL CONTAINING THE
COMMAND NAMES TO BE CHECKED. EACH LIST(K) IS A
HOLLERITH CONSTANT WITH THE 'NAME' OF THE COMMAND IN
THE FIRST 9 CHARACTERS, LEFT JUSTIFIED, AND THE
NUMBER OF LETTERS IN THE 'NAME' IN THE LAST
CHARACTER.
EXAMPLE....
A COMMAND 10HTIME 0
THE DISPLAY CODE FOR D IS 04, WHICH WHEN
THE REST OF THE WORD IS MASKED
IS THE INTEGER 4 - THE NUMBER OF
LETTERS IN TIME.
*****

```

```

*****
SUBROUTINE PRINTR(NAM,VAR,N,M,ND)
*****
C THE PURPOSE OF THIS SUBROUTINE IS TO PRINT THE VALUES OF A NON,
C ONE, OR TWO DIMENSIONAL VARIABLE. THE VARIABLE MAY BE PRINTED TO
C TAPE 6 OF TO A FILE DESIGNATED BY THE USER OR TO BOTH OR TO NEITHER.
C THE DECISION ON WHERE TO PRINT IS BASED ON A CODE PASSED IN THE
C COMMON BLOCK LISTOUT FOR EACH VARIABLE. ONCE THE DECISION IS MADE
C AS TO WHERE TO PRINT, THE SUBROUTINE LISTER IS CALLED TO DO THE
C ACTUAL PRINTING. THIS SUBROUTINE IS CALLED WITH THE ENTRY
C POINT PRINTP FOR REAL VARIABLES AND THE ENTRY POINT PRINTI
C FOR INTEGER VARIABLES
*****

```

THE CALL PARAMETERS AND LABELED COMMON BLOCKS REQUIRED ARE
EXPLAINED BELOW.

```

PARAMETERS-----
NA...THE INDEX FOR THE ARRAY LISTO CORRESPONDING TO THE
      VARIABLE TO BE PRINTED.
NAM...MOLLERITH CONSTANT WITH THE NAME OF THE VARIABLE IN THE
      FIRST 9 CHARACTERS, LEFT JUSTIFIED, AND THE NUMBER OF
      LETTERS IN THE NAME IN THE LAST CHARACTER; THIS LAST
      CHARACTER WILL BE AN INTEGER VALUE FOR THE NUMBER OF
      LETTERS IN THE NAME WHEN THE PRECEDING NINE CHARACTERS
      ARE MASKED. IF NO NAME IS TO BE PRINTED WITH THE VARIABLE
      THIS SHOULD BE 10H A.
VAR...THE VARIABLE TO BE PRINTED
N....THE ROW DIMENSION OF THE VARIABLE TO BE PRINTED
M....THE COLUMN DIMENSION OF THE VARIABLE TO BE PRINTED
ND...THE ROW DIMENSION OF THE VARIABLE FROM THE DIMENSION
      STATEMENT IN THE CALLING PROGRAM.

```

```

COMMON-----LABELED LISTOUT
NL.....THE NUMBER OF ACTIVE ELEMENTS IN THE ARRAY LISTO

```



```

LISTO.....AN ARRAY DIMENSIONED AT LEAST NL WITH AN ELEMENT
FOR EACH VARIABLE TO BE PRINTED. IF THE VALUE
IS A 1 THE VARIABLE WILL NOT BE PRINTED. IF THE
VALUE IS A 2 THE VARIABLE WILL BE PRINTED TO THE
FILE SPECIFIED BY DSP IN LABELED COMMON -CONTROL-.
IF THE VALUE IS A 3 THE VARIABLE WILL BE PRINTED
TO BOTH TAPE 6 AND THE FILE DSP.
IF THE VALUE IS A 4 THE VARIABLE WILL BE PRINTED
TO TAPE 6 ONLY.

COMMON-----LABELED CONTROL
DSP..AN INTEGER SPECIFYING A TAPE NUMBER FOR THE USERS DESIRED
PRINT FILE OR A LEFT JUSTIFIED, ZERO FILLED NAME OF THE
USERS FILE. (USUALLY TAPE 5 IS DISCONNECTED FOR HARD
COPY STORAGE OF THE PRINTED VALUES AND THE FILE DSP IS
CONNECTED TO THE TERMINAL FOR IMMEDIATE DISPLAY TO THE
USER)

COMMON-----LABELED TITLE
ITITLE...DIMENSIONED 5, THIS PROVIDES SPACE TO PRINT A 50
CHARACTER TITLE ABOVE THE VARIABLE PRINTOUT. IF
THE ARRAY TITLE IS NON BLANK THE NAME OF THE
VARIABLE WILL NOT BE PRINTED. THE ARRAY IS SET
TO ALL BLANKS AFTER EACH CALL TO THIS SUBROUTINE.
*****

```

```

C ***** SUBROUTINE LISTER(NAME,VAR,N,M,ND,FOR,DST) *****
C
C THIS SUBROUTINE IS CALLED BY THE SUBROUTINE PRINTR TO TO THE
C ACTUAL PRINTING OF THE VARIABLES. THE PARAMETERS USED IN THE
C CALL ARE GENERATED BY PRINTR AND WILL NOT BE EXPLAINED HERE, AS
C THE TWO SUBROUTINES ARE MEANT TO BE USED TOGETHER.
C THIS SUBROUTINE USES ONE OTHER LABELED COMMON BLOCK TO
C DETERMINE THE DESIRED FORMAT FOR THE VARIABLE TO BE PRINTED
C IN. THESE TERMS ARE EXPLAINED BELOW.
C
C *****
C
C COMMON-----LABELED FORMS
C
C NDRFP.....AN INTEGER INDICATING THE NUMBER OF COLUMNS TO BE
C PRINTED ON ONE LINE ON THE FILE DESIGNATED BY THE
C USER WITH DSP.
C
C NONSD.....AN INTEGER INDICATING THE NUMBER OF SIGNIFICANT
C DIGITS TO BE PRINTED TO THE FILE DSP.
C
C NOFW.....AN INTEGER INDICATING THE FIELD WIDTH OF THE FORMAT
C USED FOR PRINTING THE VARIABLE TO THE FILE DSP.
C THIS NUMBER MUST BE 7 MORE THAN NONSD.
C
C NORFP.....AN INTEGER SIMILAR TO NDRFP, EXCEPT THIS ONE
C CORRESPONDS TO THE FILE TAPE 6.
C
C NONSD.....AN INTEGER SERVING THE SAME FUNCTION AS NONSD, ONLY
C FOR TAPE 6
C
C NOFW.....AN INTEGER SPECIFYING THE FIELD WIDTH FOR TAPE 6 PRINT
C *****

```


MATRIX MANIPULATION SUBROUTINE CALL PARAMETERS

```
CALL ADDMAT(A,NAD,N,M,B,NBD,C,NCD)  ** A = B + C **
A -- OUTPUT MATRIX
NAD -- ROW DIMENSION OF A FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
N -- ROW DIMENSION OF A,B, AND C
M -- COLUMNS DIMENSION OF A,B, AND C
B & C -- MATRICES TO BE ADDED
NBD -- ROW DIMENSION OF B FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
NCD -- ROW DIMENSION OF C FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
```

```
CALL SUBMAT(A,NAD,N,M,NB,NBD,C,NCD)  ** A = B - C **
SAME PARAMETERS AS FOR ADDMAT EXCEPT THAT NOW MATRIX
C IS BEING SUBTRACTED FROM MATRIX B
```

```
CALL TRANPOS(A,NAD,N,M,B,NBD)  ** A = B (TRANPOSE)
A -- OUTPUT MATRIX
NAD -- ROW DIMENSION OF A FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
N -- ROW DIMENSION OF A AND COLUMN DIMENSION OF B
M -- COLUMN DIMENSION OF A AND ROW DIMENSION OF B
B -- INPUT MATRIX
NBD -- ROW DIMENSION OF B FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
```

```
CALL COPYAB(A,NAD,N,M,B,NBD)  ** A = B **
A, NAD, B, AND NBD ARE THE SAME AS FOR TRANPOS
N -- ROW DIMENSION OF A AND B
M -- COLUMN DIMENSION OF A AND B
```

```
CALL PRESET(A,NAD,N,M,PPE)  ** A = PRE **
```

```

A -- OUTPUT MATRIX
NAD -- ROW DIMENSION OF A FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
N -- ROW DIMENSION OF A
M -- COLUMN DIMENSION OF A
PRE -- REAL VALUE TO BE SET IN EACH ELEMENT OF A

```

```

CALL IDENT(A,NAD,N) ** A = I **
A -- OUTPUT MATRIX
NAD -- ROW DIMENSION OF A FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
N -- ROW AND COLUMN DIMENSION OF A

```

```

CALL MAT2MPY(A,NAD,N,M,B,NBD,N1,C,NCD) $$ A = 9 * C $$
A -- OUTPUT MATRIX
NAD -- ROW DIMENSION OF A FROM DIMENSION STATEMENT IN
      CALLING PROGRAM; N -- ROW DIMENSION OF A AND B
M -- COLUMN DIMENSION OF A AND C
R & C -- MATRICES TO BE MULTIPLIED
NRD -- ROW DIMENSION OF B FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
N1 -- COLUMN DIMENSION OF B AND ROW DIMENSION OF C
NCD -- ROW DIMENSION OF C FROM DIMENSION STATEMENT IN
      CALLING PROGRAM

```

```

CALL MAT3MPY(A,NAD,N,M,B,NBD,N1,C,NCD,N2,D,NDD)
$$ A = R * C * D $
A, NAD, N, NBD, N1, AND NCD ARE THE SAME AS FOR MAT2MPY
M -- COLUMN DIMENSION OF A AND D
R,C, & D -- MATRICES TO BE MULTIPLIED
N2 -- COLUMN DIMENSION OF C AND ROW DIMENSION OF D
NDD -- ROW DIMENSION OF D FROM DIMENSION STATEMENT IN
      CALLING PROGRAM

```



```

CALL INVERT(A,NAD,B,NBD,N,IFAIL)      ** A = B(INVERSE)
A -- OUTPUT MATRIX
NAD -- ROW DIMENSION OF A FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
B -- INPUT MATRIX (THIS MATRIX DESTROYED DURING
      EXECUTION)
NBD -- ROW DIMENSION OF B FROM DIMENSION STATEMENT IN
      CALLING PROGRAM
N -- ROW AND COLUMN DIMENSIONS OF A AND B
IFAIL -- LOGICAL VARIABLE RETURNED TRUE IF MATRIX IS
        SINGULAR
(1) SUBROUTINE INENT IS CALLED BY THIS ROUTINE
(2) ONE ELEMENT OF LABELED COMMON -CONTROL- IS
      USED TO INDICATE WHICH FILE THE ERROR
      MESSAGE SHOULD BE WRITTEN TO IF THE MATRIX
      IS SINGULAR

```

Appendix D

Card Sequence Required To Utilize FORTRAC

The use of the batch mode program FORTRAC is described in the preliminary report "FORTRAC: A Software Package for the Design of Multivariable Digital Control Systems" which is available from the FGL branch of the Flight Dynamics Lab. The one point that needs to be expanded on is the card sequence required for the different types of problems that can be run with the program. The following pages give the card sequence required for the three types of problems; the regulator, tracker, and disturbance rejector.

The following limits are in effect with the presently dimensioned master program.

- (1) The maximum number of states including augmenting integrators allowed is 7.
- (2) The maximum number of controls is 4.
- (3) The maximum number of augmenting integrators allowed is 3.
- (4) The maximum number of samples that can be displayed in either the discrete or continuous simulation is 33.

These limits can be increased by increasing the dimensions on the 133 variables in the master program.

Card Sequence for the Regulator

1 N,M,NCC,NDD,NYY,NPP

 N = number of states
 M = number of controls
NCC = 0
NDD = 0
NYY = number of outputs
NPP = 0

 The elements of the state matrix, one row per card

 The elements of the control matrix, one row per card

 The elements of the output matrix, one row per card

T The sampling interval to be used

 The desired closed-loop eigenvalues, N cards with a real
 and an imaginary part on each card

0 No observer will be designed

K An integer indicating the power to which the closed-loop
 matrix should be raised

NDS Number of time intervals required in the discrete-time
 simulation

 N initial conditions for the states, all on one card

KPS,NDS,NSIDS

 KPS = the accuracy to be used in the Kutta-Merson subroutine
 (suggest approximately .0000001)

 NDS = number of sample-time intervals required in the
 continuous-time simulation

 NSIDS = number of divisions required in each sample-time
 interval

 N initial conditions for the states, all on one card

Card Sequence for the Tracker

1 N,M,NCC,NDD,NYY,NPP

N = number of states
M = number of controls
NCC = number of commands
NDD = 0
NYY = number of outputs
NPP = number of outputs integrated (equal to NCC)

The elements of the state matrix, one row per card

The elements of the control matrix, one row per card

The elements of the command matrix, one row per card
(this must be entered even if the matrix is null)

The elements of the output matrix, one row per card

The elements of the matrix defining the outputs that will
track the input command

T The sampling interval to be used

The number of integrators desired for each integrated
output, all entered on one card

The desired closed-loop eigenvalues, (N + the sum of the
numbers entered on the preceding card) cards with a real
and an imaginary part on each card

0 No observer will be designed

K An integer indicating the power to which the closed-loop
matrix should be raised

NDS Number of time intervals required in the discrete-time
simulation

N initial conditions for the states, all on one card

Initial conditions for the integrators, all on one card

EPS,NDS,NSIDS

Same as for the regulator

N initial conditions for the states, all on one card

Initial conditions for the integrators, all on one card

Card Sequence for the Disturbance Rejector

1 N,M,NCC,NDD,NYY,NPP

N = number of states
M = number of controls
NCC = 0
NDD = number of disturbances
NYY = number of outputs
NPP = number of outputs integrated (number of outputs to
reject disturbances)

The elements of the state matrix, one row per card

The elements of the control matrix, one row per card

The elements of the disturbance matrix, one row per card

The elements of the output matrix, one row per card

The elements of the matrix defining the outputs that will
reject disturbances

T The sampling interval to be used

.. All remaining cards are the same as for the tracker

Vita

James A. Colgate was born on 13 February 1948 in Fort Collins, Colorado. He moved to Miami, Arizona at the age of 7 and graduated from Miami High School in 1966. He graduated from the United States Air Force Academy in 1970 with a Bachelor of Science Degree in Electrical Engineering. After graduation, he entered pilot training at Williams AFB, Arizona and completed training in July 1971. From August 1971 to September 1974 he served as a WC-135 pilot in the 55th Weather Reconnaissance Squadron. He attended Squadron Officer School in residence from September to December 1974 and then spent the next year at a remote radar site, Indian Mountain Alaska, as a Weapons Controller. He entered the School of Engineering, Air Force Institute of Technology, in June 1976.

Permanent address: 110 Miami Gardens

Miami, Arizona 85539

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER 14 AFIT/GGC/EE/77D-1	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) 6 INTERAC - AN INTERACTIVE SOFTWARE PACKAGE FOR DIRECT DIGITAL CONTROL DESIGN		5. TYPE OF REPORT & PERIOD COVERED 9 Master's thesis
7. AUTHOR(s) 10 James A. Colgate		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN) Wright-Patterson AFB, Ohio 45433		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE 11 Dec 1977
		13. SECURITY CLASS. (of this report) Unclassified
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES Approved for public release; AFR 190-17 JEROME F. GUESS, Captain, USAF Director of Information		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Control Software Discrete Control Interactive Programming		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The purpose of this investigation is to develop an interactive user oriented software package for direct digital control system design. The batch mode program, FORTAC, developed by Professor Brian Porter of the University of Salford, England, is the source of the computational algorithms used. This report details the input, output, and program sequence control software developed to produce an efficient, user oriented interactive package. <i>over</i>		

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

012225

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

✓ The package is very forgiving of user errors and gives the user complete control over what data to input, what data will be output, and which parts of the program are executed. The package is quite useful in the design process for discrete-time time-optimal control systems, where many possible control parameter variations must be examined. ↑

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)