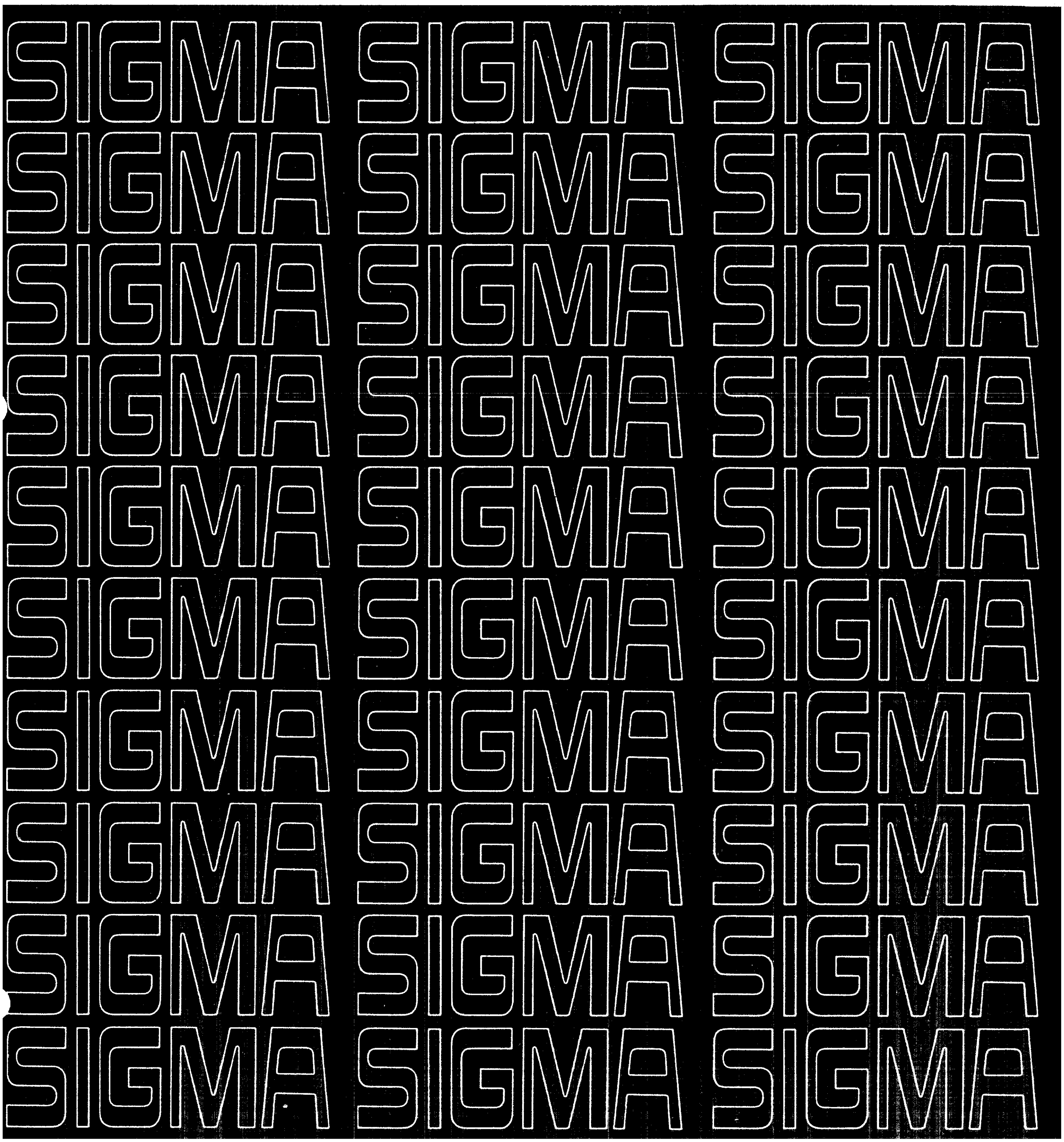
SDS SIGMA 5/7 FUNCTIONAL MATHEMATICAL PROGRAMMING SYSTEM

Reference Manual

SCIENTIFIC DATA SYSTEMS



Price: $3.50

FUNCTIONAL MATHEMATICAL PROGRAMMING SYSTEM

REFERENCE MANUAL

for

SDS SIGMA 5/7 COMPUTERS

PRELIMINARY EDITION

90 16 09A

April 1969

SDS

SCIENTIFIC DATA SYSTEMS/701 South Aviation Boulevard/El Segundo, California 90245

Printed in U.S.A.

RELATED PUBLICATIONS

Title	Publication No.
SDS Sigma 5 Computer Reference Manual	90 09 59
SDS Sigma 7 Computer Reference Manual	90 09 50
SDS Sigma 5/7 Batch Processing Monitor (BPM) Reference Manual	90 09 54
SDS Sigma 5/7 Batch Processing Monitor (BPM) Operations Manual	90 11 98

NOTICE

The specifications of the software system described in this publication are subject to change without notice. The availability or performance of some features may depend on a specific configuration of equipment such as additional tape units or larger memory. Customers should consult their SDS sales representative for details.

CONTENTS

APPENDIXES

TABLES

ILLUSTRATIONS

1. INTRODUCTION

This manual describes the Functional Mathematical Programming System (FMPS) for SDS Sigma 5/7 computers. FMPS is a mathematical programming system composed of functions for solving linear programming (LP) problems. The manual is designed for the user who is familiar with mathematical programming theory and application. Chapter 1 provides general information about FMPS features. These features include:

- Subroutines, called "procedures", for solving linear programming problems.
- A user-oriented control language for sequencing operations, controlling exception conditions, and adjusting tolerances.
- The flexible design of communication files and format options, and the ability, at the level of each major function, to direct the output stream to magnetic tape (in addition to the printer), permitting FMPS to be used as a free-standing package, or as part of a user-designed optimization package.

Chapters 2 and 4 discuss basic concepts and basic procedures, respectively, of FMPS that are applicable to all operating modes. FMPS control language statements are described in detail in Chapter 3. Chapter 5 presents data formats and data deck organization. Chapter 6 outlines procedures used in the linear programming operating mode, and Chapter 7 describes procedures used in the separable programming operating mode. When these procedures are identical in both modes, they are repeated in Chapter 7 for user convenience. Appendix A describes parametric programming and ranging procedures (an optional extension to the basic system); Appendix B is a list of error messages; and Appendix C presents an FMPS LP mode sample run.

PROCEDURES

FMPS procedures and their functions are given in Table 1 below. (Basic FMPS operating procedures are given in Chapter 4.)

Table 1. FMPS Procedures

Procedure	Purpose
INPUT	Reads matrix data from cards or tape in standard FMPS format or in various SHARE formats such as LP 90/94, UNIVAC 1108 LP, or CDC CDM4.
OUTPUT	Displays the input or current matrix in various formats.
REVISE	Reads correction data for modifying the matrix.
CRASH	Creates an initial basis structure for the current matrix and performs preliminary validity checks on the matrix.
OPTIMIZE, INVERT	Performs the actual linear programming solution.
SOLUTION	Displays the solution values in various formats.
ERRORS	Displays the computation errors incurred during the solution process for the primal and dual problems.
CONDITION	Prints out the communications region contents.
GET	Retrieves information about a row or column and alters the strategy in the control language.
BASISOUT	Punches or files (FILE parameter) the current basis structure and bounds status.
SAVE	Saves the contents of the communications region, the various internal work areas, and all internal files (MATRIX, INVERSE, etc.) on the tape file RESTART.
BASISIN	Inputs a new basis or modifies the existing basis.
RESTORE	Restores (from file RESTART) the data areas and internal files saved by SAVE.
PARARHS, PARAOBJ	Performs post-optimal parametric analysis of the solution with respect to the right-hand-side and objective function. (Refer to Appendix A.)
RANGE	Performs post-optimal range analysis. (Refer to Appendix A.)
LOADLIST	Loads a list of row labels and/or column labels to be used as selection lists or masks during the OUTPUT, SOLUTION, and/or RANGE procedures when selective output is desired.

CONTROL LANGUAGE

The sequence of operations executed in an FMPS run is controlled through statements, written in a user-oriented control language, that

- Initialize and, if desired, modify tolerances during execution.
- Assign input/output devices at the FMPS level.
- Preprogram action to be taken in case of exception or error conditions.

In the following chapters of the manual, certain conventions have been adopted for defining FMPS commands. Capital letters indicate command words that are required in the literal form shown. Lower case letters are figurative representations of parameters. Command parameters enclosed by braces ({ }) indicate a required choice. Bracketed ([ ]) parameters are optional. The format of the FMPS control language closely resembles the FORTRAN language. A procedure is activated by using the CALL statement as shown below,

```
CALL procedure [(argument)]
```

where CALL is followed by the name of the procedure and, if required, a list of arguments enclosed by parentheses to be used by the procedure. For example, the statement

```
CALL OUTPUT (BYROWS)
```

causes the input matrix to be listed by rows.

Initialization and modification of tolerances are performed by means of assignment statements. Reserved names have been assigned to each tolerance available to the user. For example, the statement

```
FDJZT = 1.0D-6
```

assigns to the DJ zero tolerance the value 0.000001. Other examples of tolerances available to the user are FMPIVT (minimum pivot clearance during optimization) and ILINES (number of lines to be printed per page).

Provision is made for user working-storage variables. The language allows execution of simple arithmetic such as

```
IF (FWD3.LT.1.0D-8) GO TO 325
FWD3=FWD3/10.
```

where

FWD3 is a user working-storage variable.

325 is the label of a statement in the control program as in a FORTRAN program.

Reserved variable names have been assigned for the handling of exception interrupts. For example, the statement

```
ASSIGN 460 TO KUBS
```

can be used to cause statement 460 to be executed if unboundedness occurs during optimization or parametric procedures. Assignments are dynamic and can be modified under program control during the course of execution.

COMMUNICATION REGION

An area of computer memory called the communications region (CR) contains all variables with reserved names (such as FDJZT, ILINES, KUBS, etc.). FMPS initializes these variables to standard values; therefore, it is not necessary to initialize them in the control program if the standard values are appropriate.

FILES

Data is carried in disc or tape files. Their purpose is to hold FMPS data in a format allowing maximum processing speed. The standard FMPS files are MATRIX, INVERSE, UTIL1, and UTIL2. These files carry the matrix, its inverse, and various intermediate information (UTIL1 and UTIL2). In addition, the RESTART file may be used for intermediate dumping of the run status. The DEVICE and ATTACH procedures must be used to define Data Control Blocks (DCBs) through which files are to be used and to assign these files to these devices. (See Chapter 4 for a detailed description of these procedures.) The files are internal to FMPS and are not intended to be used as input or output files by user-designed programs.

INPUT DATA

Data can be input to FMPS from cards or tapes, in either card image format, or FORTRAN unformatted WRITE format. (FORTRAN unformatted WRITE format provides for better data packing when using user-written matrix generators.) Input data for FMPS is accepted by the following procedures: INPUT, REVISE, LOADLIST, and BASISIN.

OUTPUT

Most FMPS procedures create printer output. The OUTPUT, SOLUTION, and BASISOUT procedures write output on magnetic tape in addition to the printer if the user so chooses. The magnetic tape output for OUTPUT and

SOLUTION is in FORTRAN unformatted WRITE format, which provides a compact data format for interface with user-designed report writers. The BASISOUT procedure produces either punched cards or card images on magnetic tape. Both are suitable for subsequent reloading by the BASISIN procedure. As with input files, a symbolic unit for each output file must be declared by means of the ATTACH procedure.

Users need not be concerned with the format of the FMPS internal files since INPUT and OUTPUT transfer data to or retrieve data from them in a user-oriented format. However, note that the user must assign DCBs for the internal files at the beginning of the run.

To provide a convenient method for abstracting the output results (whether they are written on tape or printed), the OUTPUT, SOLUTION, and RANGE procedures include many optional parameters. For example, OUTPUT provides for listing the matrix by rows, by columns, in matrix tableau format, or in coded format. (In coded format, coefficients are symbolized by letters showing the sign and magnitude of the coefficients.) Similarly, the RANGE procedure can be made selective with respect to the type of variable printed, that is, printing only the basic, only the nonbasic, or both. Furthermore, RANGE can select individual items of information for printing.

All three procedures can be made selective with respect to the individual rows and/or columns to be printed, that is,

1. Print only specified rows.
2. Print all rows except specified rows.
3. Print all rows which match specified masks.
4. Print all rows except those which match specified masks.

Similar options are available independently for columns.

SELECTION LISTS

Selection lists consist of names (rows and/or columns) and/or masks (rows and/or column names with an asterisk matching any character in the row or column name in the corresponding position in FMPS internal files). Since the same selection list usually applies over an entire run, a single procedure, LOADLIST, is used to load the rows-and-columns selection lists.

Items selected are controlled by optional arguments. For example

```
CALL SOLUTION (ROWS,LISTR,COLS,
EXCEPT,LISTC,
1     RCHAPTER,2,5,CCHAPTER,2,4,8,FILE,
      'SOLFILE')
```

causes the solution to be written on the user file SOLFILE as well as on the printer, outputting only the rows included in row selection list LISTR, the columns not included in column selection list LISTC, the row name and its slack activity for rows, and the column name and its activity for columns. One selection list may be used to control output items during several procedures such as OUTPUT, SOLUTION, and RANGE. Such procedures have an optional parameter indicating whether the information to be output is to be controlled by the selection list. The list need be loaded only once. In some procedures such as RANGE, reduction of output and calculations will result in sizable savings in execution time.

2. FMPS FUNDAMENTALS

This chapter describes in detail some basic elements of FMPS such as variables and constants available in the control language, internal files, selection lists, and the structure of communication files.

CONSTANTS

The FMPS control language uses three types of constants in Arithmetic statements and as parameters in procedural CALL statements. They are: integer, floating-point, and character.

INTEGER

A number written without a decimal point is called an integer constant. An integer constant is composed of one to seven decimal digits. It may be preceded by a plus sign, a minus sign, or a blank. If unsigned, it is assumed to be positive. It may not contain any embedded blanks. Sample valid and invalid integer constants are shown in the tables below.

VALID INTEGER CONSTANTS

0
100000
-54
+1

INVALID INTEGER CONSTANTS

-17 35	Contains an embedded blank
100, 000	Contains a comma

FLOATING-POINT

A number with a decimal point, optionally followed by a decimal exponent (written as the letter D followed by a signed or unsigned one- or two-digit integer constant) is called a floating-point constant. The magnitude of a real constant must be compatible with that allowed by FORTRAN for the machine being used. However, only eight significant digits are allowed. A floating-point constant may be preceded by a plus sign, a minus sign, or a blank. Embedded blanks are not allowed. The first table shown below gives correct floating-point constants and their real magnitudes. The second table shows invalid representations of floating-point constants.

VALID FLOATING-POINT CONSTANTS

-3.49	-3.49
1.47D3	1470.
-.23D-4	-.000023
0.0	zero
.2D+2	20.

INVALID FLOATING-POINT CONSTANTS

.123456789D1	Will be truncated to eight significant digits
1,217.2	Contains a comma
1.7D 2	Contains a blank between D and 2
1.3E4	E not valid - must use D

CHARACTER

A string of from one to eight characters, enclosed by single quotation marks, is called a character constant. (The single quotation mark is represented by a 5-8 punch on the card.) Character constants, sometimes called literals, may be composed of alphabetic, numeric, special, or blank characters. The quotation marks are not part of the character constant, but are used to delimit it. The quotation mark itself is the only special character not allowed within the body of the character constant. Correct character constants are shown directly below, incorrect examples in the second table.

VALID CHARACTER CONSTANTS

'ROWS'
'THE END'
'2+3'
'DOG/CAT'
'   '

INVALID CHARACTER CONSTANTS

'OPERATION'	Only eight characters are allowed
'ABD	Second quotation mark missing
'A'BC'	Embedded quotation mark not allowed

VARIABLES

Variables (storage references) are symbolic names of either locations within the control program (user working-storage variables), or locations in the FMPS communication region (CR variables).

All storage within FMPS is identified by type. The four types of variables, each identified by its leading character, are shown in Table 2 below.

Table 2. Types of Variables

Code	Type
I	Integer
F	Floating-Point
A	Alphanumeric
K	Interrupt

User-created variables are distinguished from CR variables by their second character, which must be a W. Also, user-created variable names may contain a maximum of four characters, while CR variable names may contain a maximum of eight characters. User-created variable names containing more than four characters will be truncated to four. The user may create a total of 50 integer and K-type variables and a total of 50 floating-point and alphanumeric variables. Each distinct type is discussed below.

INTEGER

Each integer (I-type) variable is a single precision word containing a single precision integer value. Integer variables may assume any of the values of an integer constant. An I-type variable may be used in an Arithmetic statement, an IF statement, a WRITE statement, or as a parameter in a procedure CALL statement. Table 3 contains a list of all CR integer variables and an explanation of each.

Some sample integer variables are shown in the following tables.

VALID INTEGER VARIABLE NAMES

IFREQI	CR variable for inversion iteration frequency
IWBG	User working-storage variable
IW3	User working-storage variable

INVALID INTEGER VARIABLE NAMES

IU5	Not a valid CR variable name nor a valid user working-storage name since second character is not W
KROW	Integer names must begin with I

FLOATING-POINT

Each floating-point (F-type) variable is a double precision word and contains a double precision floating-point value. A floating-point variable may assume any of the values of a floating-point constant. It may be used in an Arithmetic statement, an IF statement, a WRITE statement, or as a parameter in a procedure CALL statement. Table 4 contains a list of all floating-point CR variables and an explanation of each.

Table 3. Integer (I-type) CR Variables

CR Variables	Initialized Value	Explanation
IDNFSOL	0	Number of feasible solutions found for the integer problem.
IDULSTOP	0	Controls the brake on DUAL in MIP operating mode. If IDULSTOP is nonzero, DUAL will run to a feasible solution to the (possibly reduced) problem every IDULSTOP major iterations.
IESWT	0	The console jump switch to interrogate. IESWT must be 0 - 8. If zero, no switch is tested. If IESWT is 1 - 8, and the jump switch is on, KESWT interrupt will occur.
IFREQA	0	Iteration frequency interrupt for OPTIMIZE, PARAOBJ, and PARARHS. If IFREQA is 0, no interrupt will occur. Otherwise, the KFREQA interrupt will occur every IFREQA iterations.
IFREQI	0	Iteration frequency interrupt for inversion. In the iterating procedures OPTIMIZE, PARAOBJ, and PARARHS, the KINV interrupt will occur every IFREQI iterations (IFREQI > 0).
IIWGHT	0	Infeasibility weighting switch. When IIWGHT is 1, the reciprocal of the amount of infeasibility is used as a weighting factor. When IIWGHT is -1, the amount of each infeasibility is used as a weighting factor. When IIWGHT is 0, all infeasibilities are given equal weight.
ILOGC	0	Iteration logging frequency on console typewriter.
ILOGP	0	Iteration logging frequency on standard printing device.
ILOGSS	0	On/Off switch for printing column selection messages during pricing of matrix.
ILINES	50	Maximum number of lines to be printed on a page.
INCAND	0	Number of profitable candidates from which one is selected during pricing of the matrix. For example, if INCAND is 5, then from each group of 5 profitable columns, the most profitable is selected. If INCAND is 0, the system will attempt to choose the optimum set.
ININF	0	Current number of infeasible variables in the basis.

Table 3. Integer (I-type) CR Variables (cont.)

CR Variables	Initialized Value	Explanation
INVTIME	0	Switch controlling the KINV interrupt timing routine in the PRIMAL procedure. If INVTIME is 0, the timing routine is active and causes KINV interrupts at times such that the total optimization time tends to be minimum. If INVTIME is -1, the timing routine is not active.
IPARAM	0	Parametric programming mode indicator. If IPARAM is -1, PARAOBJ is in effect, if IPARAM is 1, PARARHS is in effect, and if IPARAM is 2, PARARIM is in effect.
IPASS	2000	Number of assignments allowed during solution of the integer subproblem in MIP mode before the KASS interrupt occurs.
IPFES	2000	Number of feasible solutions allowed to the integer subproblem in the MIP mode before the KPFES interrupt occurs.
IPSOLTN	0	After solution of an integer subproblem in MIP operating mode, IPSOLTN will be nonzero if there was a change in the integer solution and will be zero if the integer solution has remained the same.
ITCNT	0	Current iteration count.
ITIME	0	The length of time, in minutes, before the KTIME interrupt will occur. The KTIME interrupt does not occur if KTIME is set to zero. Whenever the KTIME interrupt occurs, KTIME is set to zero. Time for KTIME is measured from the time of the last initialization of ITIME.

Table 4. Floating-Point (F-type) CR Variables

CR Variables	Initialized Value	Explanation
FABSZT	1.0D-12	Absolute zero tolerance. Any computed number is replaced by zero if its absolute value is less than FABSZT.
FCMPDJ	0.5D0	Factor used in determining effective DJ when infeasible, that is, DJE=FCMPDJ*DJ+(1.0-FCMPDJ)*DJI where DJE is Effective DJ, DJ is True DJ of column, and DJI is DJ based on infeasibility removal qualities of column.
FDJZT	1.0D-07	DJ zero tolerance. If the absolute value of the reduced cost (DJ) is less than FDJZT, it is considered zero.
FEPSILON	0.0	The value used to replace zero right-hand-side elements of inequalities on degenerate problems. If the constraint is of the less-than type, a zero RHS element is replaced with FEPSILON. If the constraint is of the greater-than type, a zero RHS element is replaced with -FEPSILON.
FINFZT	1.0D-07	Infeasibility zero tolerance. If the absolute value of the amount of infeasibility is less FINFZT, the variable is considered feasible.
FMINVT	1.0D-09	Minimum inversion pivot tolerance. During INVERT, in the nontriangularized portion, an element is not considered as potentially pivotal unless its absolute value is greater than FMINVT.
FMPIVT	1.0D-08	Minimum pivot tolerance. During any optimization procedure (here, INVERT is not considered an optimization procedure), an element is not considered as potentially pivotal unless its absolute value is greater than FMPIVT.
FOBJVAL	0.0	Current objective function value.
FOBJWT	-1.0	Objective function weight: -1.0 for maximization, 1.0 for minimization.

Table 4. Floating-Point (F-type) CR Variables (cont.)

CR Variables	Initialized Value	Explanation
FRDIFT	4096.0	Relative difference tolerance. This tolerance represents a power of 2, that is, 2.0**12 is 4096. If the difference of two numbers is in the low-order twelve bits, the numbers are considered identical. Any user-specified value must be a power of 2, such as 8192.0 or 16384.0.
FRELZT	0.0	Relative zero tolerance. If the absolute value of the summation of a series of numbers divided by the absolute value of the largest sum or number is less than FRELZT, the summation is considered to be zero.
FSINF	0.0	Current sum of infeasibility. Each infeasibility is summed in absolute terms.
FTHETAC	0.0	Initial value of THETA for PARAOBJ.
FTHETACM	0.0	Maximum value of THETA for PARAOBJ.
FTHETACP	0.0	The incremental value for THETA during PARAOBJ for which the KSOLTN interrupt will occur.
FTHETAR	0.0	Initial value of THETA for PARARHS.
FTHETARM	0.0	Maximum value of THETA for PARARHS.
FTHETARP	0.0	The incremental value for THETA during PARARHS for which the KSOLTN interrupt will occur.

Correct and incorrect floating-point variable names are shown in the tables below.

VALID FLOATING-POINT VARIABLE NAMES

FMPIVT	CR variable for minimum pivot tolerance for optimization
FW01	User working-storage variable
FW5D	User working-storage variable

INVALID FLOATING-POINT VARIABLE NAMES

FDOG	Not a valid CR variable name nor a valid user working-storage name since second character is not W
AW07	Floating-point names must begin with F

ALPHANUMERIC

Each alphanumeric (A-type) variable is a double precision word and contains up to eight characters. An alphanumeric variable may assume any of the values of a character constant. It may be used in a simple Arithmetic statement, in an IF statement, in a WRITE statement, or as a parameter in a procedure CALL statement. Table 5 contains a list of all alphanumeric CR variables and an explanation of each, followed by tables showing valid and invalid alphanumeric variables.

VALID ALPHANUMERIC VARIABLE NAMES

ARHS	CR variable for name of current right-hand-side
AWLD	User working-storage variable
AW07	User working-storage variable

INVALID ALPHANUMERIC VARIABLE NAMES

AMESS	Neither a valid CR variable name nor a valid user working-storage name since second character is not a W
NAME	Alphanumeric names must begin with A

INTERRUPT

During the execution of a mathematical programming system, many conditions arise which require some form of corrective action. Although much thought is generally given to the corrective action to be taken, no particular action is suitable under all circumstances. The interrupt processing concept in FMPS has been developed to facilitate initiation of appropriate corrective action when it is required.

For each condition requiring corrective action or for any point where greater user flexibility is desired, a CR interrupt variable is reserved. The function of each variable is to serve as a pointer to a control language statement or group of statements that will perform the corrective active or procedural steps desired by the user and allow for the resumption or exiting of the procedure causing the interrupt.

FMPS will initialize all interrupt variables to perform standard recovery techniques. The user, through the use of the ASSIGN command, may reset any interrupt variable to perform his own sequence of commands.

An interrupt (K-type) variable may assume the value of any valid statement number. The user working-storage K-type variable may be used in a GO TO statement, an ASSIGN statement, or a WRITE statement. Conversely, a K-type CR variable may only be referenced in a WRITE statement or an ASSIGN statement. The K-type CR variable is a single precision word containing a pointer to a control language sequence of instructions to be executed if an interrupt in a procedure occurs. Table 6 contains a list of all interrupt variables and an explanation of each. Sample K-type variables are shown in the tables below.

VALID K-TYPE VARIABLE NAMES

KMAJER	CR major error interrupt variable used by many procedures
KWST	User working-storage variable

INVALID K-TYPE VARIABLE NAMES

KQUIT	Not a valid CR variable name nor a user valid working-storage name since second character is not a W
IWAL	K-type names must begin with K

FILES

FMPS includes two types of files:

•	INTERNAL FILES	For intermediate storage during FMPS procedures (magnetic tape or disc)
•	COMMUNICATION FILES	For communication between FMPS and user-designed programs (magnetic tape)

Table 7 lists required and optional files for operating in the linear programming (LP) or separable programming (SEP) operating mode. This table also indicates the input/output device type (sequential such as tape, or random-access such as disc) that is required, preferred, or optional.

Table 5. Alphanumeric (A-Type) CR Variables

CR Variables	Initialized Value	Explanation
ADATA	None	Contains the name of the data deck for data reading procedures such as INPUT, REVISE, etc. Also used by data-outputting procedures (such as BASISOUT) to name output data deck. It specifies the name that appears on the NAME card of image input. (Refer to Chapter 5 for general data formats).
AOBJ	None	Contains name of objective function row.
APBNAME	None	Contains name of problem.
APOBJ	None	Contains name of PARAOBJ change row.
APRHS	None	Contains name of PARARHS change column.
ARHS	None	Contains name of right-hand-side.

Table 6. Interrupt (K-Type) CR Variables

CR Variables	Initialized Value	Explanation
KFREQA	None	Iteration frequency A interrupt. This interrupt will occur when IFREQA iterations occur.
KINV	None	Inversion interrupt. This interrupt will occur when IFREQI iterations occur or an inversion is required.
KIOER	Terminate Run	Input/output device error interrupt.
KMAJER	Terminate Run	Major error interrupt.
KMINER	None	Minor error interrupt.
KNFS	None	No feasible solution interrupt.
KSOLTN	None	SOLUTION print interrupt.
KTIME	None	Elapsed time interrupt. This interrupt will occur when ITIME minutes have elapsed.
KUBS	None	Unbounded solution interrupt.

Table 7. Internal and Communication Files

Required Internal Files		
File Name	Device Type	Description of File
MATRIX	Sequential or Random-Access	Contains the internal representation of the matrix processed by INPUT.
INVERSE	Preferably Random-Access	Contains the internal representation of the product form of the inverse.
UTIL1	Sequential or Random-Access	A utility file used by many procedures for scratch storage.
UTIL2	Sequential or Random-Access	A utility file used by many procedures for scratch use.
Optional Internal Files		
RESTART	Sequential	Used by the SAVE procedure for storing all files for later resumption of run. Used by the RESTORE procedure for restoring the machine to the state at the time the SAVE procedure prepared the file.
Optional Communication Files		
'filename'	Sequential	Any user-defined file used for internal communication between FMPS and user's programs. Several such files can be used. The quote marks are part of the name of the file.

INTERNAL FILES

Within each operating mode of FMPS, a minimum number of internal files is required. Each internal file has been assigned a unique preempted name, and these names will be referred to throughout this manual. The user is required to attach the required files to appropriate DCBs (see Chapter 4).

STORAGE REQUIREMENTS FOR INTERNAL FILES

The number of words of disc storage required by the MATRIX file is specified by the following equation.

$$2.25\ (5M + NSP + 4N + NNZ + 4NRHS + NNZRHS)$$

where

M is the number of rows in the matrix.

NSP is the number of slack prices.

N is the number of columns in the matrix.

NNZ is the number of nonzero elements in columns.

NRHS is the number of right-hand-sides.

NNZRHS is the number of nonzero elements in right-hand-side(s).

The number of words of disc storage required by the INVERSE file is specified by the following equation

$$4.5\ (M * 1\ .25\ ANNZ)$$

where

M is the number of rows in the matrix.

ANNZ is the average number of nonzero elements in a matrix column.

The number of words of disc storage required by files UTIL1 and UTIL2 is the same as for the MATRIX file.

These estimates for disc storage may vary during certain procedures. For example, during REVISE, the storage requirement for the INVERSE file is generally twice that of the MATRIX file.

For large problems, it may not be possible to assign all files to disc storage during preliminary phases such as INPUT and REVISE. Since it is desirable to have the files on disc during the iterating procedures (OPTIMIZE, INVERT, etc.), it is suggested that the user assign all files to magnetic tape during the INPUT/REVISE phase. Following this, he may call the CONDITION and SAVE procedures.

The CONDITION output will list the current storage requirements (in words) for each file and the maximum storage required to date. The current size of the MATRIX file can be used for its disc storage requirements as well as for UTIL1 and UTIL2. The current storage requirements stated for the INVERSE file cannot be used for disc estimating since the iterating procedures have not yet been used.

For maximum efficiency, the following priority should be given in assigning files to disc for the iterating procedures.

Priority	Procedure
1	INVERSE
2	MATRIX
3	UTIL1
4	UTIL2

COMMUNICATION FILES

Communication files are the means of communication between FMPS and user-written programs. FMPS input procedures accept data from a standard card reading device or, optionally, from communication files. FMPS output procedures retrieve data from internal files and prepare printed reports. Optionally, the data may be written on a communication file.

To provide a mutually-convenient form of communication, such files are structured to be read or written with FORTRAN READ or WRITE statements. By using FORTRAN input/output as the basic means of communication, the user can write his own specific matrix generators and report writers in FORTRAN.

The following table identifies the FMPS procedures that include the option of accepting input from communication files or of writing output on communication files.

Table 8. Procedures Using Communication Files

Procedure (in LP mode)	Card Format	FORTRAN Format
LOADLIST	Yes	No
INPUT	Yes	Yes
OUTPUT	No	Yes
REVISE	Yes	No
SOLUTION	No	Yes
BASISOUT	Yes	No
BASISIN	Yes	No

The following paragraphs describe basic communication file structure and the means by which FORTRAN READ and WRITE statements may be used to access the data.

CARD FORMAT FILES

All data decks that may be read or written on a CARD file are organized as described in Chapter 5. Each data deck is preceded by a NAME card which identifies the data, and each data deck is terminated with an ENDATA card.

Whenever a procedure requires input data, the input device, whether card reader or CARD file, is searched for a NAME card with an identification field (columns 15 to 22) that matches the current contents of communication region variable ADATA.

Whenever a procedure produces a data deck (that is, BASISOUT), NAME and ENDATA cards are also produced. If the output device is other than the card punch, that is, a CARD file, the card file is positioned to the logical end-of-file and the new data deck is written. The logical end-of-file is assumed to be a NAME card with zzzzzzzz in the identification field.

Procedures such as INPUT, REVISE, and LOADLIST require data input. Whether the input is from cards, card images on magnetic tape, or in FORTRAN unformatted WRITE format, the following conventions apply:

1. The data must be preceded by a name record identifying the data record, and the data must be followed by an ENDATA record.

2. In the control program, the CR variable ADATA must be initialized with the name of the data set to be loaded before the procedure requiring input is called. For example,the following sequence,

   ```
   ADATA = 'MATRIX1'
   CALL INPUT
   ```

   causes the card data set with the name MATRIX1 to be loaded by the INPUT procedure.

3. The card data sets must be placed after the END statement of the control program. The card data sets must follow each other in the sequence of input.

4. Input records on magnetic tape can occur in any sequence. FMPS will rewind the input tape, if necessary, to locate the desired set of data if the tape was positioned beyond the record to be loaded.

5. For proper operation, it is necessary that all input files include as the last record a NAME record with the name zzzzzzzz and an ENDATA record. This constitutes the logical end-of-file for FMPS.

6. When writing output on magnetic tape, FMPS automatically supplies the NAME and ENDATA records. The name is copied from the current CR variable ADATA which must be initialized to the desired name by the user before executing the output. If the tape includes data prior to the output operation, the new output data is appended to the current data and a logical end-of-file (NAME zzzzzzzz and ENDATA) is added. Decks punched by FMPS also include the NAME and ENDATA records.

7. The INPUT procedure includes the option of reading card decks or magnetic tape reels prepared for other linear programming packages such as LP 90/94, 1108 LP, and CDM4. When reading such data from cards, NAME and ENDATA must precede and follow the input data. When reading from magnetic tape, the NAME and ENDATA records must not be present on the tape.

FORTRAN FORMAT FILES

A FORTRAN format file consists of a series of unformatted, FORTRAN-written records on tape. Each record contains 60 double precision (DP) words. The structure of each record is shown in Figure 1. The first three DP words are used to identify the record.

The first DP word contains the name of the procedure generating the record or the name of the procedure for which this record is input. N14, the left half of the first DP word, contains the first four characters of the name, and N58, the right half of the first DP word, contains the last four characters of the name.

The second DP word contains the subname of the record. SN14, the left half of the second DP word, contains the first four characters of the subname. SN58, the right half of the second DP word, contains the last four characters of the subname.

The third DP word contains the record number and index of the word last used in the record. RN, the left half of the third DP word, contains the record number. RN is used to signal the end of a series of records.

As an example, if three 60-word records were required to contain the information, RN in the first record would be -1, in the second -2, and in the third 3. Therefore, if RN is negative, it indicates that there is more of the same kind of information in the next record. When RN is positive, it indicates that this is the end of records containing the stated information. ILAST, the right half of the third DP word, contains the index of the last item in the record. ILAST is always less than or equal to 60. The fourth through the sixtieth DP words contain the information in groups of three DP words.

DATA STORAGE ON RECORDS

Conventions for storage of data on records are outlined below.

1. All names (character strings of eight characters or less) are stored with the first four characters of the name in the left half of a DP word and the last four characters in the right half of the DP word.
2. Floating-point values are stored as double precision floating-point.
3. Integers are stored in the left half (most significant) of a DP word.

As with CARD communication files, all input data must be preceded by a NAME record. In addition, output will be preceded by a NAME record that contains the contents of CR cell ADATA. The format of a NAME record is shown in Figure 2.

The last record on a communication file will be a NAME record whose name is zzzzzzzz (supplied by the user).

Each time information is written on a FORTRAN communication file, the tape is positioned to the zzzzzzzz name record and the zzzzzzzz record is overwritten with a new NAME record containing the contents of CR cell ADATA. The information is then written followed by a new NAME zzzzzzzz record.

Record formats produced by SOLUTION are shown in Figure 3. Record formats for the INPUT procedure are shown in Figure 4.

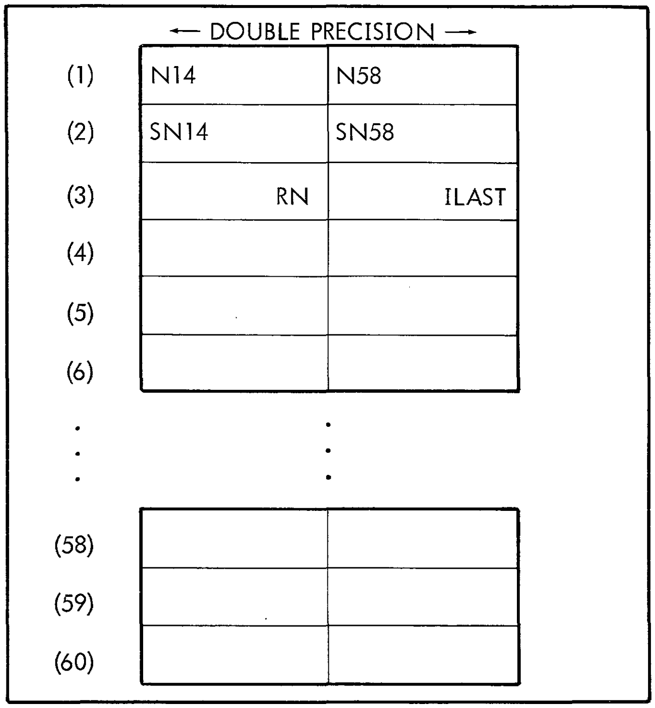


Figure 1. FORTRAN Communication File Record Structure

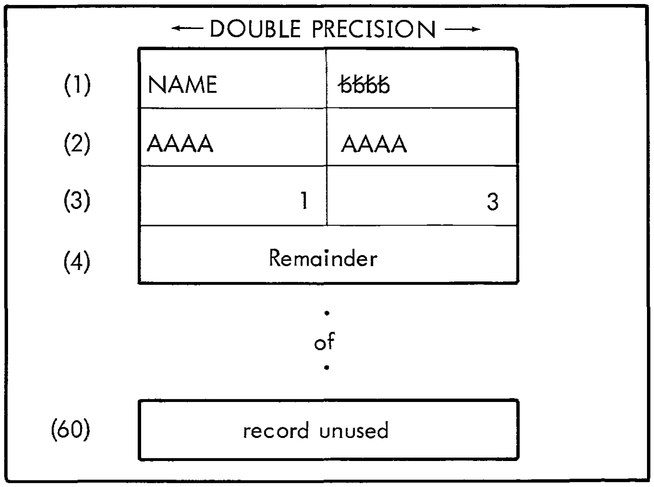


Figure 2. Format of a NAME Record

Identify

1	SOLU	TION	
2	IDEN	TIFY	
3	1	24	
4	IDEN	Tƀƀƀ	
5	APBN	AME	
6	AAAA	AAAA	Contents of CR Cell APBNAME
7	IDEN	Tƀƀƀ	
8	STAT	USƀƀ	
9	AAAA	AAAA	OPTIMALƀ, UNBOUNDƀ INFEASƀƀ, UNBOUNDƀ
10	IDENT	Tƀƀƀ	
11	FOBJ	WTƀƀ	
12	+1.0		
13	IDEN	Tƀƀƀ	
14	FUNC	TION	
15	C_o		Objective Function Value
16	IDEN	Tƀƀƀ	
17	ARMS	ƀƀƀƀ	
18	AAAA	AAAA	Name of Right-Hand Side
19	IDEN	Tƀƀƀ	
20	AOBJ	ƀƀƀƀ	
21	AAAA	AAAA	Name of Objective Row
22	IDEN	Tƀƀƀ	
23	ITER	ƀƀƀƀ	
24	I		Iteration Count
25			Remainder of record not used
⋮	⋮		
60			

Rows

1	SOLU	TION	
2	ROWS	ƀƀƀƀ	
3	RN	ILAST	
4	ROWN	AME1	
5	NUMB	ERƀƀ	
6	I		Row Number
7	ROWN	AME1	
8	ATƀƀ	ƀƀƀƀ	
9	AAƀƀ	ƀƀƀƀ	FR, EQ, LL, UL,
10	ROWN	AME1	
11	ACTI	VITY	
12			Row Activity Value
13	ROWN	AME1	
14	SLAC	Kƀƀƀ	
15			Slack Activity Time
16	ROWN	AME1	
17	LLIM	ITƀƀ	
18			None or Lower Limit Value
19	ROWN	AME1	
20	ULIM	ITƀƀ	
21			None or Upper Limit Value
22	ROWN	AME1	
23	DUAL	ACTƀ	
24			Dual Activity Value
25	ROWN	AME1	
26	COST	ƀƀƀƀ	
27			Slack Price Value
28	ROWN	AME1	
29	DJƀƀ	ƀƀƀƀ	
30			Reduced Cost Value of Slack
31	ROWN	AME2	
32	NUMB	ERƀƀ	
33	I		Row Number
⋮	⋮		
60			

Columns

1	SOLU	TION	
2	COLU	MNSƀ	
3	RN	ILAST	
4	COLN	AME1	
5	NUMB	ERƀƀ	
6	J		Column Number
7	COLN	AME1	
8	ATƀƀ	ƀƀƀƀ	
9	AAƀƀ	ƀƀƀƀ	BS, LL, UL, FR, FX,
10	COLN	AME1	
11	ACTI	VITY	
12			Column Activity Value
13	COLN	AME1	
14	COST	ƀƀƀƀ	
15			Column Cost Value
16	COLN	AME1	
17	LLIM	ITƀƀ	
18			None or Lower Limit Value
19	COLN	AME1	
20	ULIM	ITƀƀ	
21			None or Upper Limit Value
22	COLN	AME1	
23	DJƀƀ	ƀƀƀƀ	
24			Reduced Cost Value of Column
25	COLN	AME2	
26	NUMB	ERƀƀ	
27	J		Column Number
⋮	⋮		
60			

Figure 3. Record Formats Produced by SOLUTION

Rows

1	INPU	Tƀƀƀ	
2	ROWS	ƀƀƀƀ	
3	RN	ILAST	
4	Aƀƀƀ	ƀƀƀƀ	N, L, G, E, Row Type
5	ROWN	AME1	Name of Row
6	Ignored		
7	Aƀƀƀ	ƀƀƀƀ	
8	ROWN	AME2	
9	Ignored		
⋮	⋮		
58	Aƀƀƀ	ƀƀƀƀ	
59	ROWN	AMEM	
60	Ignored		

RHS

1	INPU	Tƀƀƀ	
2	RHSƀ	ƀƀƀƀ	
3	RN	ILAST	
4	RHSN	AME1	RHS Name
5	ROWN	AME1	Row Name
6	$\not{b}_i$		RHS Value
7	RHSN	AME1	
8	ROWN	AME1	
9	$\not{b}_i$		
⋮	⋮		
58	RMSN	AME5	
59	ROWN	AMEM	
60	$\not{b}_i$		

Sprices

1	INPU	Tƀƀƀ	
2	SPRI	CESƀ	
3	RN	ILAST	
4	SLKN	AME1	Name of Slack
5	COST	ROW	Name of Cost Row
6	C_i		Slack Price
7	ALKN	AME2	
8	LOST	ROWƀ	
9			
⋮	⋮		
58	SLKN	AMEM	
59	COST	ROWƀ	
60	C_i		

Ranges

1	INPU	Tƀƀƀ	
2	RANG	ESƀƀ	
3	RN	ILAST	
4	RNGN	AME1	Range Column Name
5	ROWN	AME1	Row Name
6	R_i		Range Value
7	RNGN	AME1	
8	ROWN	AME2	
9	R_i		
⋮	⋮		
58	RNGN	AME1	
59	ROWN	AMEM	
60	R_i		

Columns

1	INPU	Tƀƀƀ	
2	COLU	MNSƀ	
3	RN	ILAST	
4	COLN	AME1	Column Name
5	ROWN	AME1	Row Name
6	A_{ij}		Element Value
7	COLN	AME1	
8	ROWN	AME2	
9	A_{ij}		
⋮	⋮		
58	COLN	AMEM	
59	ROWN	AMEM	
60	A_{ij}		

Bounds

1	INPU	Tƀƀƀ	
2	BOUN	DSƀƀ	
3	RN	ILAST	
4	AAƀƀ	ƀƀƀƀ	} LO, UP, FX, FR, PL Type of Bound
5	COLN	AME1	Column Name
6	B		Bound Value
7	AAƀƀ	ƀƀƀƀ	
8	COLN	AME2	
9	B		
⋮	⋮		
58	AAƀƀ	ƀƀƀƀ	
59	COLN	AMEN	
60	B		

Endata

1	INPU	Tƀƀƀ
2	ENDA	TAƀƀ
3	I	

Figure 4. Record Formats for INPUT

3. FMPS CONTROL LANGUAGE STATEMENTS

INTRODUCTION

An FMPS run always includes a set of cards that specify the operations to be executed. These cards are grouped together in a control program. Rather than using fixed-format control cards, FMPS uses control statements that are compiled by FMPS at the beginning of the run.

STATEMENT TYPES

The control language for FMPS was designed to be a subset of the FORTRAN language. There are five basic types of statements:

1. The procedural CALL statement, which loads and transfers control to one of the FMPS procedures. This type of statement is analogous to a FORTRAN subroutine call.

2. Arithmetic statements, which evaluate simple arithmetic expressions.

3. Program flow control statements, such as ASSIGN, GO TO, EXIT, RETURN, and IF, which transfer control to a statement other than the next one in sequence.

4. The WRITE statement, which displays any user or common-storage variable on the standard output device. The TITLE statement provides a heading for each page of output.

5. Delimiting statements, which indicate the end of the control program. The END statement is a message to the compiler that there are no more statements to be processed. It is not executable. The STOP statement is executable and indicates that execution of the control program is to terminate.

CARD FORMAT

The card format for the FMPS control language is identical to that of FORTRAN.

Column 1 is used to indicate a comment card. A C punched in column 1 indicates that the rest of the card is a comment, and is not processed. The comment card will appear on the listing produced by the compiler. Comment cards may be used freely to give information or improve readability.

Any statement, other than an END statement, may be given a statement (step) number. A step number is any unsigned integer between 1 and 9999. It may be placed anywhere in columns 2-5 of the card.

Column 6 is reserved to indicate a continuation card. As many continuation cards as are needed may be used, but they can only be used to continue the parameter list of a procedure CALL statement. They may not be used with any other kind of statement. Any nonblank character punched in column 6 will indicate that the card is a continuation of the parameter list from the previous card. A statement may begin in column 7 or anywhere thereafter.

Columns 73-80 are ignored, and may be used for sequence numbers if the user wishes. A summary of card format is shown below.

```
[C] [step] {nonblank} statement    [sequence numbers]
           {blank   }
```

CONTROL LANGUAGE STATEMENTS

CALL The procedure CALL statement causes the specified procedure to be loaded into memory, control to be transferred to the procedure, and the set of parameters specified in the argument list to be communicated to it. The procedure CALL statement has the form

```
      CALL pname [(parameter 1,  parameter 2, . . .)]
```

where

pname is the name of the FMPS procedure to be executed.

parameter 1, . . . represents the values to be transmitted to the procedure. Parameters may be constants, variables (either CR variables or user working-storage variables), or keywords. Some procedures have no parameters associated with them. The parameters are always enclosed by parentheses and separated by commas.

Correct and incorrect procedure CALL statements are shown below.

VALID PROCEDURE CALL STATEMENTS

```
CALL OPTIMIZE
CALL ENTER (LP)
CALL ATTACH ('FILE1', 'F:F1')
```

Note that the CALL ATTACH procedure above could be written as

```
AWD4 = 'FILE1'
AW01 = 'F:F1'
CALL ATTACH (AWD4, AW01)
```

INVALID PROCEDURE CALL STATEMENTS

```
ENTER(LP)        CALL must be specified
CALL ENTER 'LP'      Missing parentheses
```

CALL ATTACH ('PROBFILE' 'FILETAPE')	Parameters not separated by commas

The parameter list of a procedural CALL may make use of a continuation card as in

```
  CALL ATTACH ('PROBFILE',
X'FILETAPE', CARD, NEW)
```

Note that a field must not be broken in the middle, and that the preceding card must end with a comma.

The examples shown below illustrate improper continuation cards for procedure CALL statements.

INVALID CONTINUATION CARDS FOR PROCEDURE CALL STATEMENT

CALL ATTACH X('PROBFILE', 'FILETAPE')	At least one parameter must be on first card
CALL ATTACH ('PROBFILE' X,'FILETAPE', CARD NEW)	Preceding card must end with a comma

ARITHMETIC

The Arithmetic statement is used to initialize or set all storage-reference variables (CR or user working-storage) except interrupt (K-type) variables. The Arithmetic statement has the form

```
srsym = arithex
```

where

srsym is either a CR or user working-storage variable.

arithex is an arithmetic expression of the form

variable

constant

variable $\left\{\begin{matrix}+\\-\\*\\/\end{matrix}\right\}$ constant

variable $\left\{\begin{matrix}+\\-\\*\\/\end{matrix}\right\}$ variable

and in which variable refers to either a CR or a user working-storage variable.

Mixed mode is allowed between integer and floating-point computations, but all alpha computations must not mix modes. An arithmetic expression that contains a floating-point number will be done in double precision floating-point arithmetic.

Compare the following tables of valid and invalid Arithmetic statements.

VALID ARITHMETIC STATEMENTS

```
ARHS = 'ALOY1'
FW01 = FW01 + 1
IWNM = 79.0
FW01 = FW01 * IWNM
ILOGP = IWNM/79
```

INVALID ARITHMETIC STATEMENTS

KW01	=	100	K-type cells cannot be defined with an Arithmetic statement
ARHS	=	FW01	Mixed mode not allowed with alpha type
IWNM	=	FW01 * IW01 + 4	Invalid form of arithmetic expression

ASSIGN The ASSIGN statement is used to initalize or set an interrupt (K-type) variable. It has the form

```
ASSIGN stmtno TO kxxx
```

where

stmtno is any valid statement number (1-9999) appearing in the control language program.

kxxx is a K-type CR or user working-storage variable.

The following two statements are correct uses of ASSIGN.

VALID ASSIGN STATEMENTS

```
ASSIGN 100 TO KMAJER
ASSIGN 20  TO KW01
```

This list shows incorrect uses of the ASSIGN statement.

INVALID ASSIGN STATEMENTS

ASSIGN SEVEN TO KWD1	Statement number must be an integer constant
ASSIGN 100 TO IW01	Assignment must be made to a K-type variable only

GO TO The GO TO statement causes the unconditional transfer of control to the statement specified by the statement number after GO TO. The GO TO statement has the form

GO TO $\begin{Bmatrix}\text{stmtno}\\ \text{kxxx}\end{Bmatrix}$

where

stmtno is any valid statement number (1-9999) appearing in the control language program.

kxxx is a K-type user working-storage variable that has been defined by an ASSIGN statement.

The two lists below present correct and incorrect uses of GO TO.

VALID GO TO STATEMENTS

```
GO TO 100
GO TO KW01
```

INVALID GO TO STATEMENTS

GO TO A	A is not a K-type user working-storage variable
GO TO KMAJER	KMAJER is a K-type CR variable, not a user working-storage variable

IF The IF statement makes a conditional transfer of control to the statement specified by a statement number. It may be used in the construction of loops. IF has the form

IF (srsym .op. $\begin{Bmatrix}\text{srsym}\\ \text{constant}\end{Bmatrix}$) GO TO stmtno

where

srsym is either a CR or user working-storage variable.

constant is a valid constant.

op enclosed by periods, is a two-letter code that represents one of the following conditions.

Code	Condition
GT	Greater than
GE	Greater than or equal
LT	Less than
LE	Less than or equal
EQ	Equal
NE	Not equal

stmtno is any valid statement number (1-9999) appearing in the control language program.

When IF is executed, the expression within the parentheses is evaluated first. If it is true, control is transferred to the specified statement number. If it is not true, control is passed to the next statement in the program sequence. Mixed mode is allowed if integer and floating-point quantities are involved. Mixed mode is not allowed if an alpha quantity is used.

The sample IF statements below are correct.

VALID IF STATEMENTS

```
IF (FOBJWT .GT. IW41)GO TO 30
IF (ARHS .EQ. 'ROWS') GO TO 150
```

These IF statements are incorrect.

INVALID IF STATEMENTS

IF (ARHS .EQ. FW01) GO TO 20 Mixed mode is not allowed if alpha quantity involved

IF (IWO1 LT 7) GO TO 10 LT must be enclosed in periods

IF (FW75) 10, 20, 30 This form of IF statement is not allowed in this control language

RETURN The RETURN statement is used to return control to a procedure that has created an interrupt. When an interrupt occurs, control will be given to the statement whose number has been assigned to the corresponding CR interrupt (K-type) variable for that particular condition. After the number, it may be desired to return to the procedure that caused the interrupt. The RETURN statement has the form

RETURN

An example of interrupt processing using a RETURN statement is shown below.

```
      ASSIGN 150 TO KINV
      IFREQI = 50
      CALL OPTIMIZE
      =
 150  CALL INVERT
      RETURN
```

Note that OPTIMIZE will interrupt for an INVERT every 50 iterations. Control will be transferred to statement 150 which is a CALL for INVERT, and following the INVERT, control will be transferred to OPTIMIZE via RETURN.

EXIT The EXIT statement is a special type of statement used in the FMPS control language. Like the RETURN statement, the EXIT statement is concerned with interrupt processing. After receiving an interrupt, it may not be desirable to return to the procedure causing the interrupt. The EXIT statement may be used to exit the procedure and to continue processing with the statement following the procedure CALL statement that triggered the interrupt. EXIT has the form

```
EXIT
```

An example of interrupt processing using an EXIT statement is given below.

```
    ASSIGN 200 TO KNFS
    CALL OPTIMIZE

200 CALL OUTPUT (BYROWS, ROWS, LISTI)
    EXIT
```

Note that if no feasible solution condition is encountered by OPTIMIZE, control is transferred to statement 200 to output the infeasible rows, and the following EXIT statement will cause control to be transferred to the statement after CALL OPTIMIZE.

WRITE The WRITE statement (not to be confused with the standard FORTRAN WRITE statement) may be used to display the current value of any CR or user working-storage variable on the system output device. The variable name and its value are printed. The WRITE statement has the form

```
WRITE srsym
```

where

srsym is either a CR or user working-storage reference symbol.

Notice that only one symbol may be referenced on a WRITE statement.

Some uses of WRITE are shown below.

```
AW01 = 'EXAMPLE'  Printout will contain
WRITE AW01        AW01 = EXAMPLE

FW07 = .2365D3    Printout will contain
WRITE FW07        FW07 = 236.5
```

TITLE This statement, which is a special FMPS control language statement, provides a page heading on each page of the output produced by execution of the control program. The TITLE statement has the form

```
TITLE heading
```

where

heading is a string of literal alphanumeric characters that terminate by column 72.

The title is printed out as shown below

```
TITLE THIS IS THE TITLE.
```

STOP The STOP statement terminates execution of the control program. The STOP statement has the form

```
STOP
```

END The END statement is a nonexecutable statement that defines the end of a source program for the compiler and must be the last statement of every program. Since the END statement is not executable, it should have a statement number. END has the form

```
END
```

SAMPLE FMPS PROGRAM

Figure 5 shows an example of a typical FMPS control language program.

```
C     DEFINE PAGE TITLE
      TITLE FMPS CONTROL LANGUAGE EXAMPLE
C     ENTER LINEAR PROGRAMMING OPERATING MODE
      CALL ENTER(LP)
C     INITIALIZE MAJOR AND MINOR ERROR INTERRUPTS
      ASSIGN 1000 TO KMAJER
      ASSIGN 1010 TO KMINER

      CALL DEVICE('DISC1',DISC,'B')
      CALL DEVICE('DISC2',DISC,'C')
      CALL DEVICE('DISC3',DISC,'D')
      CALL DEVICE('DISC4',DISC,'E')
C
C     ATTACH INTERNAL FILES MATRIX, INVERSE,UTIL1, UTIL2 TO THE SYMBOLIC
C     DISC UNITS DISC1, DISC2, DISC3, DISC4
      CALL ATTACH(MATRIX, 'DISC1')
      CALL ATTACH(INVERSE, 'DISC2')
      CALL ATTACH(UTIL1, 'DISC3')
      CALL ATTACH(UTIL2, 'DISC4)
C
C     DEFINE NAME OF INPUT DATA DECK
      ADATA = 'PLANT1'
C     INPUT THE LP MATRIX
      CALL INPUT
C     DEFINE NAME OF RHS AND OBJECTIVE FUNCTION ROW
      ARHS = 'RHS1'
      AOBJ = 'COSTROW'
C     OUTPUT BYROWS, THE NON-ZERO ELEMENTS OF INPUT MATRIX
      CALL OUTPUT(BYROWS)
C
C     INITIALIZE OPTIMIZE INTERRUPTS KINV, KNFS, KUBS
      ASSIGN 2000 TO KINV
      ASSIGN 2100 TO KNFS
      ASSIGN 2200 TO KUBS
C     SET INVERSION FREQUENCY TO 100
      IFREQI = 100
C     OPTIMIZE INPUT MATRIX
      CALL OPTIMIZE
C     OUTPUT THE OPTIMAL SOLUTION
      CALL SOLUTION
C     TERMINATE RUN
      STOP
C
C     PROCESS MAJOR ERROR INTERRUPT BY TERMINATING RUN
 1000 STOP
C
C     PROCESS MINOR ERROR INTERRUPT BY EXITING PROCEDURE CAUSING IT
 1010 EXIT
C
C     PROCESS INVERT INTERRUPT BY CALLING INVERT AND RETURNING TO
C     PROCEDURE REQUESTING IT.
 2000 CALL INVERT
      RETURN
C
C     PROCESS NO FEASIBLE SOLUTION INTERRUPT BY OUTPUTING THE INFEASIBLE
C     ROWS, PUNCHING THE CURRENT BASIS STRUCTURE, AND TERMINATING RUN
 2100 CALL OUTPUT(BYROWS, ROWS, LISTI)
 2110 CALL BASISOUT
      STOP
C
C     PROCESS UNBOUNDED SOLUTION INTERRUPT BY OUTPUTING THE UNBOUNDED
C     COLUMN, PUNCHING THE CURRENT BASIS, AND TERMINATING RUN.
 2200 CALL OUTPUT(BYCOLS, COLS, LISTU)
      GO TO 2110
C     END OF CONTROL PROGRAM
      END
```

Figure 5. Sample FMPS Control Language Program

4. BASIC FMPS PROCEDURES

This chapter describes those FMPS procedures that are available under all FMPS operating modes. These operating procedures perform the following functions.

- Establish the operating mode.
- Define input/output devices.
- Assign files to input/output devices.
- Define selection lists.

FMPS operating procedures and their functions are given in Table 9 below.

Table 9. FMPS Operating Procedures

Procedure	Purpose
ENTER	Establish the operating mode.
DEVICE	Defines storage media for run.
ATTACH	Attaches symbolic files to DCBs.
LOADLIST	Inputs names and/or masks to be used as a selection list.

OPERATING PROCEDURES REPERTOIRE

Each of the procedures outlined in Table 9 above will be explained in detail in the following paragraphs.

ENTER The ENTER procedure establishes the operating mode for FMPS. Therefore, it must be the first procedure used. The mode may not be changed during a run. The following list contains codes for parameters currently available for ENTER. One of the following parameters must be specified.

Parameters	Explanations
LP	FMPS establishes the linear programming operating mode.
SEP	FMPS establishes the separable programming operating mode.

The following interrupt may occur through misuse of ENTER.

Interrupt	Causes
KMAJER	1. Unrecognizable parameter. 2. Operating mode already established.

DEVICE The DEVICE procedure defines magnetic tapes and RAD files to be used as storage media during the FMPS run. This procedure contains parameters informing FMPS of the data control block (DCB) to be used with the file or tape and the organization of the file (consecutive-sequential or keyed direct-access). This data is given to BPM via the !ASSIGN control command; the DEVICE procedure passes it to FMPS.

Symbolic units must be defined by a call for DEVICE before FMPS files can be attached to them. A symbolic unit may be defined only once during a run.

DATA CONTROL BLOCKS

The data control blocks for use with FMPS are included in the system at installation. Nominally the system is built to the maximum of 10 DCBs whose names are F:1,F:2, . . .,F:10. Thus, !ASSIGN cards for a run are restricted to these DCBs. In addition, the F:1 DCB is preempted by FMPS in the storage of the control language programs. However, any of the remaining DCBs may be assigned to either tape or RAD. RAD DCBs may be organized sequentially or as direct-access. The internal FMPS file INVERSE should always be a RAD file and as such must be a keyed direct-access file. Note that the !ASSIGN control command designates the physical location (RAD or tape) of the data transmitted via a DCB.

DEVICE ARGUMENT

The DEVICE procedure requires three arguments, as in

```
CALL DEVICE ('symbolic unit' {TAPE   } 'DCB key')
                             {'DISC',}
```

where

'symbolic unit' specifies the symbolic unit defined by DEVICE to which internal and communication files may be attached.

TAPE indicates that the file or tape was specified as consecutive-sequential on the !ASSIGN card.

DISC Indicates that the file was specified as keyed direct-access on the !ASSIGN card.

'DCB key' is one of the following codes that specify the DCB name to be used.

Code	DCB Name
'B'	DCB F:2
'C'	DCB F:3
⋮	⋮
'J'	DCB F:10

For example, the procedural call CALL DEVICE ('INVS', 'C') would define symbolic unit 'INVS' to be a RAD file with keyed direct-access organization, to be driven via the F:3 DCB.

ATTACH The ATTACH procedure attaches symbolic files to DCBs. There are two classes of files that must be attached. The first class consists of files reserved for internal use by FMPS. All internal files have preempted names recognizable as keywords such as MATRIX, INVERSE, etc. (refer to Table 7). The second class of files consists of files used for communications between the user and FMPS. The user assigns symbolic names (eight or less characters enclosed by quotation marks) to communication files.

When attaching FMPS internal files to DCBs, ATTACH requires the use of two parameters. For example,

CALL ATTACH (INVERSE,'SYMB1')

assigns internal file INVERSE to the symbolic unit 'SYMB1'.

When attaching communication files to symbolic units ATTACH requires the use of four parameters. The third parameter (which is not required for internal FMPS files) describes the mode of the file. The mode may be specified as CARD, implying 80-column card image format, or FORTRAN, implying standard communication format. The fourth parameter, OLD or NEW, specifies whether the tape has previously been prepared by a program (or FMPS) and contains information to be preserved (OLD), or whether the tape is a new tape without information to be saved on it (NEW). If the NEW parameter is specified, FMPS writes a pseudo end-of-file record at the beginning of the tape (NAME zzzzzzzz, ENDATA). If it is an output file, it is defined as NEW. It is imperative that, if a communication file (whether CARD or FORTRAN) is defined, NEW or OLD follow the file definition.

Symbolic files may be reattached to different DCBs during a run. If the INVERSE file is reattached, an INVERT call must be made following the latest ATTACH. A common use of the reattach facility is in connection with the RESTART file. For example

```
CALL ATTACH (RESTART, = 'TAPE1')
CALL RESTORE
CALL ATTACH (RESTART, 'TAPE2')
  .
  .
  .
CALL SAVE
```

Also, the statement

CALL ATTACH('OUTFILE', 'COMMTAPE', FORTRAN, NEW)

assigns communication file 'OUTFILE' to DCB 'COMMTAPE' in standard communication format.

The following interrupt may occur within ATTACH.

Interrupt	Causes
KMAJER	1. Symbolic unit not defined.
	2. Internal FMPS file assigned as communication file.
	3. Unrecognizable parameter.
	4. Internal random-access file assigned to sequential-access device.
	5. Communication file not specified as OLD or NEW.

LOADLIST The LOADLIST procedure is responsible for the input of a list of names and/or masks from cards or communication files to be used as a selection list during output of procedures such as SOLUTION, OUTPUT, etc.

The first parameter of the procedure defines which of two lists, LISTR or LISTC, is to be loaded. LISTR is the list used to contain the names and/or masks for row selection or exception. LISTC is the list used to contain the names and/or masks for column selection or exception.

The names in a list correspond to the name of a row or column in the matrix. Masks are used to represent classes of rows or columns that have unique character configurations in their names. A mask is composed of eight characters. The characters in the mask are matched, position by position, with a row or column name. If all positions match, then that row or column name is considered part of the selection list. If one or more characters within the mask are an asterisk(*), that position(s) will match with the corresponding position(s) of any row or column name. For example,

CRUDE***

is a mask that considers any row or column name having CRUDE as its first five characters as part of the selection list.

Input to LOADLIST is from card images on the standard card reading device unless the FILE parameter is specified, in which case the third parameter must be the name of the file on which the data resides. The data format for the LOADLIST procedure is described in Chapter 5.

The communication region variable ADATA must be initialized before the call for LOADLIST. It contains the name of the data deck for data reading procedures such as INPUT, REVISE, etc. ADATA is also used by data outputting procedures, such as BASISOUT, to name output data deck. It specifies the name that appears on the NAME card of image input. (Refer to Chapter 5 for general data formats.)

The parameters available to LOADLIST are:

Parameter	Explanation
LISTR	Specifies that row selection list is to be loaded. If LISTR is not specified, LISTC must be.
LISTC	Specifies that column selection list is to be loaded. If LISTC is not specified, LISTR must be.
FILE	Specifies that data is on file 'filename' (card format only).

Parameter	Explanation
'filename'	Symbolic name of file, including quotation marks, on which data resides.

The FILE and 'filename' parameters are optional.

The following interrupts may occur within LOADLIST.

Interrupt	Causes
KMAJER	1. Unrecognizable parameter. 2. Undefined 'filename'. 3. NAMES or MASKS data not grouped together. 4. Unrecognizable data indicator.
KIOER	Irrecoverable input/output error on file.
KMINER	Core memory area exceeded by list. Remainder of data cards ignored.

5. DATA CARD FORMATS AND DECK ORGANIZATION

This chapter describes data card formats and data deck organization applicable for the various procedures (INPUT, REVISE, BASISIN/BASISOUT, and LOADLIST) in all FMPS operating modes. It also describes acceptable nonstandard data formats.

STANDARD CARD AND DECK FORMATS FOR INPUT

The data file for the INPUT procedure contains four types of cards in all cases.

1. NAME card
2. Indicator cards
3. Data cards
4. ENDATA card

Comment cards, identified by an asterisk (*) in column 1, may be inserted anywhere in a data deck.

NAME CARD

The first card of a data deck is always a NAME card. The NAME card gives a user-specified name to the data decks so that the data may be uniquely identified from the control program. NAME has the following format.

Columns	Description
1-4	NAME: card identification.
5-14	Blank
15-22	User-assigned name: from one to eight characters in length.
23-80	Blank

INDICATOR CARDS

The INPUT data deck consists of data cards grouped according to the type of data they contain. A group of cards containing the same type of data is called a chapter. The first card of a chapter is always an indicator card, which identifies the type of data in that chapter. The optional and required types of data appearing in a data deck for the INPUT procedure are:

Data Type	Status
ROWS	Required
SPRICES	Optional
COLUMNS	Required
RHS	Required
RANGES	Optional
BOUNDS	Optional

The format of indicator cards is given below.

Columns	Description
1-7	Data type: one of the six types shown above.
8-80	Blank

DATA CARDS

Data cards are divided into six fields. The type of data card determines the content of each field, but all data cards follow the same general format. The six fields of a data card are outlined below.

Columns	Description
1	Blank or *. If asterisk is present, it indicates that this is a comment card, which may be inserted anywhere in the data deck.
2-3	Field 1: code for type of row constraint or type of bound (see ROWS and BOUNDS cards).
5-12	Field 2: name of from one to eight alphanumeric and special characters.
15-22	Field 3: same as field 2 above.
25-36	Field 4: value of up to twelve characters, including decimal point. Sign specification is optional; if unspecified, it is assumed positive.
40-47	Field 5: same as field 2 above.
50-61	Field 6: same as field 4 above.

ENDATA CARD

The ENDATA card, which simply indicates that the end of the data deck has been reached, has the following format:

Columns	Description
1-6	ENDATA
7-80	Blank

DATA DECK ORGANIZATION

Figure 6 shows the organization of a complete INPUT data deck. Note that the dashed lines indicate optional cards and decks.

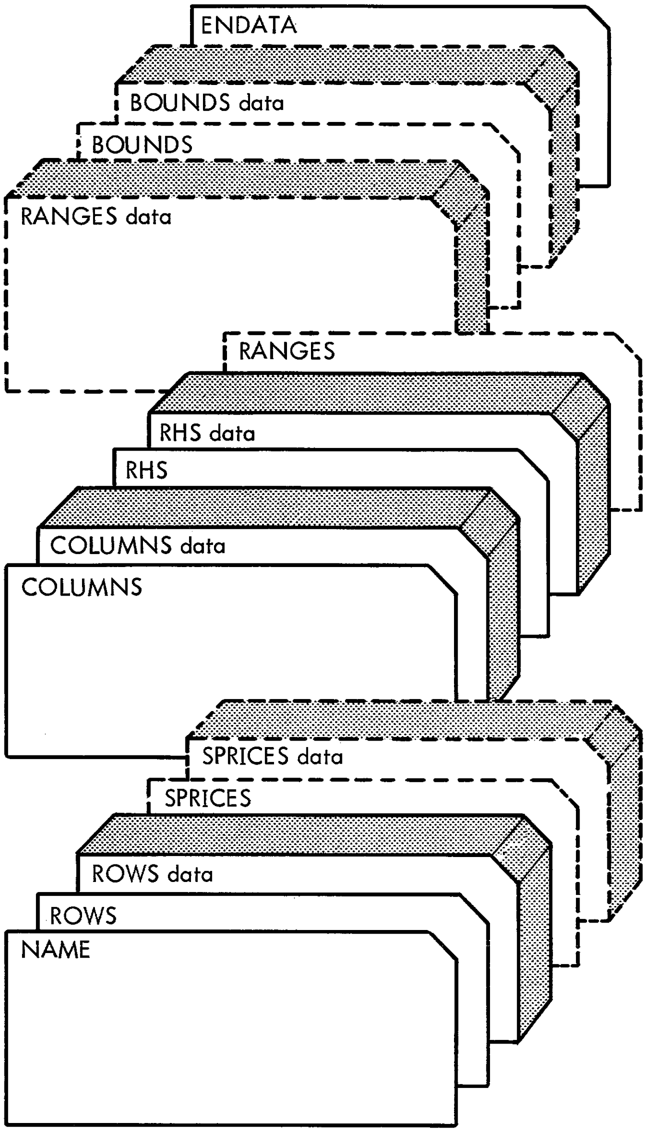


Figure 6. Data Deck Organization for INPUT

ROWS DATA CARDS

ROWS cards specify the name to be assigned to the rows of the matrix, as well as the type of constraint (equality or inequality) represented by the row. The ROWS data card format is shown below.

Columns	Description
2-3	Field 1: type of constraint as specified by the following codes:

Code	Meaning
b̸N or Nb̸	No constraint (change or objective row)
b̸G or Gb̸	Greater than or equal to
b̸L or Lb̸	Less than or equal to
b̸E or Eb̸	Equality

Columns	Description
5-12	Field 2: name of the row, where blanks are considered part of the name.
15-22	Field 3: blank
25-36	Field 4: blank
40-47	Field 5: blank
50-61	Field 6: blank

SPRICES DATA CARDS

SPRICES (slack prices) cards specify the price or prices to be associated with the slack vector of a row. The slack prices must be specified by slack: that is, when one price is given for a slack, any other prices for the same slack must be entered before the next slack is referenced. The slack prices must be entered in the same order as the slack name appears in the rows section. The SPRICES data card format is shown below.

Columns	Description
2-3	Field 1: blank
5-12	Field 2: name of the slack vector, which is identical to the name of the row with which it is associated.
15-22	Field 3: name of the cost row to which the price is associated.
25-36	Field 4: value of the slack price.
40-47	Field 5: optional and used like field 3.
50-61	Field 6: optional and used like field 4.

COLUMNS DATA CARDS

COLUMNS cards specify the names to be assigned to the columns (structural variables) in the LP matrix and define the actual values of the matrix elements in terms of column vectors. The matrix elements must be specified by column; that is, when one element is given, all other nonzero elements in that column must also be entered before another column is mentioned. Zero entries should not be specified, since they will be filled in automatically by the system. The COLUMNS data card format is shown below.

Columns	Description
2-3	Field 1: blank
5-12	Field 2: name of the column that is to contain the elements specified in the field that follow.

Columns	Description
15-22	Field 3: name of a row in which an element is to be entered.
25-36	Field 4: value of the element to be entered in the row and in the column of field 2.
40-47	Field 5: optional and used like field 3.
50-61	Field 6: optional and used like field 4.

RHS CARDS

RHS cards specify the names of the right-hand-side constraint vectors or change vectors (used in parametric programming). They define, in terms of column vectors, the values of these elements. The right-hand-side elements must be specified by RHS; that is, when one element is given, all other nonzero elements in that RHS must also be entered before another RHS is mentioned. The RHS data card format is shown below.

Columns	Description
2-3	Field 1: blank
5-12	Field 2: name of the right-hand-side (RHS) vectors or change vectors.
15-22	Field 3: name of the row in which an element is to be entered.
25-36	Field 4: value of the element to be entered in the row and in the RHS of field 2.
40-47	Field 5: optional and used like field 3.
50-61	Field 6: optional and used like field 4.

RANGES DATA CARDS

Range constraints are used when a row is to represent both a greater-than inequality and a less-than-or-equal-to inequality. When none of the rows have such double limits, range constraints are not used.

One of these limits is given in the normal manner when both upper and lower limits are desired. The type of row constraint is specified in the ROW data, and one limit (upper or lower) is specified in the RHS data. The other limit specified in this section of the data is the allowable magnitude by which the right-hand-side may vary from the value previously specified.

If b_i is the value given in the RHS section, the range r_i is specified as follows:

Type of Row	Resultant Upper Limit on Right-Hand-Side	Resultant Lower Limit on Right-Hand-Side
G	$b_i + r_i$	b_i
L	b_i	$b_i - r_i$

The set of ranges is defined as a column vector with a name specified by the user. Only one vector of ranges will be loaded by the INPUT procedure. If more than one is present, the additional vectors will be punched in REVISE format. ·

The RANGES data card format is shown below.

Columns	Description
2-3	Field 1: blank
5-12	Field 2: name of the column of ranges.
15-22	Field 3: name of a G or L row to which this range is to be applied.
25-36	Field 4: value of the range (r_i).
40-47	Field 5: optional and used like field 3.
50-61	Field 6: optional and used like field 4.

BOUNDS DATA CARDS

BOUNDS data cards impose limits on the values which the activities, or "structural variables", may assume. If none of the variables are bounded, this section of input is not needed.

When bounds are desired, they are entered as a row vector with a name specified by the user. Bounds are automatically set at 0 and $+\infty$ for all columns not specified in a BOUNDS card. Only one vector of bounds will be loaded by the INPUT procedure. However, if more than one is present, the additional vectors will be punched in REVISE format.

Within a given bounds row vector, the column (structural variable) names must appear in matrix order (that is, the same order in which column names appear in the COLUMNS section).

The user may specify both an upper and a lower bound, a lower bound only, or an upper bound only. When a single bound is specified, the other bound will remain as $+\infty$ or 0. When both upper and lower bounds on a single variable are desired, they must be entered on separate cards. Possible combinations are:

LO - UP

LO - PL

Since an upper bound of $+\infty$ is automatically generated, PL cards are ignored by INPUT.

To fix a variable at zero, the code FX with a value of zero must be used.

Lower bound values may be positive or negative; upper bound values must be positive.

The BOUNDS data card format is shown below.

Columns	Description
2-3	Field 1: type of bound as specified by the following codes:

Code	Meaning
LO	Lower bound
UP	Upper bound
FX	Fixed value
FR	Free variable (- ∞ to + ∞)
PL	Upper bound is + ∞

Columns	Description
5-12	Field 2: name of the row of bounds.
15-22	Field 3: name of the column with which the variable to be bounded is associated.
25-36	Field 4: value of the bound for an LO, UP, or FX card; otherwise blank.
40-47	Field 5: blank.
50-61	Field 6: blank.

NONSTANDARD CARD FORMATS FOR INPUT

Three nonstandard input formats are acceptable to the INPUT procedure when the parameter SHARE is used. They are:

1. LP/90/94 LP
2. UNIVAC 1108 LP
3. CDM4 LP

LP/90/94 SHARE FORMAT

The INPUT format when using LP/90/94 LP is

```
        CALL INPUT (SHARE, 'LP90')
```

where the LP90 parameter must be enclosed by single quotation marks.

LP/90/94 CHAPTERS

The following chapters of input information will be processed when using LP/90/94.

ROW ID	FIRST B
BASIS	NEXT B,kkkk
MATRIX	EOF

RHS NAMES

FMPS assigns the RHS name from the contents of columns 7 to 12 of the data cards for the FIRST B or NEXT B chapter. If these columns are blank for the FIRST B chapter data cards, the name *B1ƀƀƀ (where ƀ represents a blank) will be assigned to this RHS. If columns 7 to 12 are blank for the NEXT B chapter data cards, the RHS vectors will be named *Bkkkk, where kkkk are characters copied from the NEXT B,kkkk header card.

BASIS DATA CHAPTER

When the BASIS chapter header is encountered by the INPUT procedure, its data is punched on cards in a format acceptable to the BASISIN routine. No further processing of BASIS data occurs, but the punched cards can be loaded as a part of the FMPS input to a subsequent run. The BASIS data chapter can appear in any order relative to the other chapter headings in the input stream.

ORDER OF INPUT

The following data chapters are directly processed upon input and must appear in the order listed.

Data Type	Status
1. ROW ID	Required
2. MATRIX	Required
3. FIRST B	Required
4. NEXT B,kkkk	Optional

CARD FORMAT

ROW ID. The first card of the ROW ID chapter is a ROW ID indicator card. The card format is shown below.

Columns	Description
1-6	ROWƀID: where the characters ƀ represent a blank. This parameter is present on the first ROW ID card only; columns 1 to 6 are blank on all other ROW ID cards.
12	Row type: where the type is specified by one of the following codes.

Code	Row Type
+	Less than or equal to
-	Greater than or equal to
0	Equal to
ƀ	Indicates a Free Row (for example, Cost Row)

Columns	Description
13-18	Row name.
24	Row type.
25-30	Row name.
36	Row type.
37-42	Row name.
48	Row type.
49-54	Row name.
60	Row type.
61-66	Row name.

A pair of fields is ignored if both the row type and the row name are blank.

MATRIX. The first card of the MATRIX chapter is a MATRIX indicator card. The MATRIX data is entered column by column (all coefficients pertinent to one column must be grouped together) as shown in the format outline below. Note that only one coefficient can be defined per data card.

Columns	Description
1-6	MATRIX. This parameter is present on the first MATRIX card only; columns 1 to 6 are blank on all other MATRIX cards.
7-12	Column name.
13-18	Row name.
19-30	Coefficient value; assumed format is F12.6.

FIRST B. The first card of the FIRST B chapter is a FIRST B indicator card. This card has FIRSTƀB punched in columns 1 to 7. The data format is identical to that for MATRIX. If columns 7 to 12 are blank on the data cards, the column (right-hand-side) will automatically be named *B1ƀƀƀ.

NEXT B,kkkk. The first card of the NEXT B,kkkk chapter is a NEXT B,kkkk indicator card. This card has NEXT B,kkkk punched in columns 1 to 11. The data format is identical to that for MATRIX; if columns 7 to 12 are blank on the data cards, the column (right-hand-side) is automatically named *Bkkki, where the characters kkkk are copied from the indicator card.

BASIS. The first card of the BASIS chapter is a BASIS indicator card. BASIS data cards contain up to five pairs of names, as shown below.

Columns	Description
1-5	BASIS. This parameter is present on the first BASIS card only; columns 1 to 5 are blank on all other BASIS cards.
7-12	Variable to enter the basis.
13-18	Variable to be excluded from the basis.
19-24	Variable to enter the basis.
25-30	Variable to be excluded from the basis.
31-36	Variable to enter the basis.
37-42	Variable to be excluded from the basis.
43-48	Variable to enter the basis.
49-54	Variable to be excluded from the basis.
55-60	Variable to enter the basis.
61-66	Variable to be excluded from the basis.

EOF. The EOF card has EOF punched in columns 1 to 3.

UNIVAC 1108 SHARE FORMAT

The INPUT format when using UNIVAC 1108 LP is

```
        CALL INPUT (SHARE, '1108')
```

where the 1108 parameter must be enclosed by single quotation marks.

UNIVAC 1108 CHAPTERS

The following chapters of input information will be processed when using UNIVAC 1108.

DELETE
ROW ID
BASIS
MATRIX
FIRST B
NEXT B,kkkk
SPRICES
EOF
ENDATA

A maximum of 100 column or row names may be input as part of the DELETE data. A minor error interrupt will occur if this number is exceeded, and only the first 100 names will be used.

RHS NAMES

RHS names are formed in the same manner as described for LP/90/94 data above.

ORDER OF INPUT

The following data chapters are directly processed upon input and must appear in the order listed.

Data Type	Status
1. DELETE	Optional
2. ROW ID	Required
3. MATRIX	Required
4. FIRST B	Required
5. NEXT B,kkkk	Optional
6. SPRICES	Optional

The BASIS chapter is optional and may appear anywhere in the input deck. It is processed in the same manner described for LP/90/94. If the SPRICES chapter is present in the input data and is to be used, the argument 'SPRICES' must be present in the CALL INPUT argument list, as in

```
CALL INPUT (SHARE, '1108', 'SPRICES')
```

when the input source is the card reader, the SPRICES chapter must be placed directly after the ROW ID chapter in the data deck. When the input source is tape, the SPRICES chapter may appear at the end.

If SPRICES is used, AOBJ must be set (through the control language) to the name of the cost row for which the slack prices apply. This must be done before the call to INPUT.

CARD FORMAT

DELETE. The first card of the DELETE chapter is a DELETE indicator card. This card has DELETE punched in columns 1 to 6, and contains up to eleven name fields in columns 7-12, 13-18,...,67-72. All blank fields are ignored.

ROW ID, MATRIX, FIRST B, NEXT B,kkkk, and BASIS. These data formats are identical to the corresponding data formats for LP/90/94 SHARE.

SPRICES. The first card of the SPRICES chapter is a SPRICES indicator card. This card has the format shown below.

Columns	Description
1-7	SPRICES. This parameter is present on the first SPRICES card only; columns 1 to 5 are blank on all other SPRICES cards.
7-12	Row (slack) name.
19-30	Slack price: assumed format is F12.6.

Pairs for which both fields are blank are ignored. Inclusion of variable names which do not correspond to any variable in the matrix will cause an error comment during subsequent processing of the punched BASIS cards, but will not cause this run to be discontinued.

EOF. The EOF card has EOF punched in columns 1 to 3.

ENDATA. The ENDATA card has ENDATA punched in columns 1 to 6.

CDM4 SHARE FORMAT

The INPUT format when using CDM4 LP is

```
CALL INPUT (SHARE, 'CDM4')
```

where the CDM4 parameter must be enclosed by single quotation marks.

CDM4 CHAPTERS

The following chapters of input information will be processed when using CDM4.

ROW ID
MATRIX
FIRST B
RHS
BASIS
NEWRHS
SECOND
ENDRHS
EOR
EOF

RHS NAMES

FMPS will introduce a new RHS vector in the input matrix for every redefinition of the RHS vector in the input data. Upon input, the original RHS vector is automatically named *B0001; the first revised RHS vector, *B0002; the second revised vector, *B0003, etc. Any of the vectors can be specified for solution by assigning its name to the ARHS communication cell, for example, ARHS = '*B0002'.

ORDER OF INPUT

The following data chapters are directly processed upon input and must appear in the order listed.

Data Type	Status
1. ROW ID	Required
2. EOR	Optional
3. MATRIX	Required
4. EOR	Optional
5. FIRST B OR RHS	Required

Data Type	Status
6. EOR OR ENDRHS	Optional
7. NEWRHS OR SECOND	Optional
8. EOR OR ENDRHS	Optional
9. EOI = ENDATA	Required

The BASIS chapter is optional and is treated in the same manner as it is in LP/90/94 format.

CARD FORMAT

All data formats for CDM4 SHARE are identical to those specified for LP/90/94 except ROW ID.

The first card of the ROW ID chapter is the ROW ID indicator card. This card has the format shown below.

Columns	Description
1-6	ROWb̸ID: This parameter is present on the first ROW ID cardonly; columns 1 to 6 are blank on all other ROW ID cards.
12	Row type: where the type is specified by one of the following codes. Code — Row Type + — Less than or equal to - — Greater than or equal to 0 — Equal to b̸ — Indicates a Free Row (for example, Cost Row)
13-18	Row name.
24	Row type.
25-30	Row name.
36	Row type.
37-42	Row name.
48	Row type.
49-54	Row name.
60	Row type.
61-66	Row name.

Row types and names on ROW ID data cards are interpreted as outlined below.

1. If columns 19 to 24 or columns 12 to 18, or both, of the data card are blank, the card is ignored.

2. If columns 19 to 24 and columns 12 to 18 of the data card are nonblank, the data is read as follows:

Column 12	Row type.
Columns 13-18	Row name.

NAME AND ENDATA CARDS

Data may be read from cards or tape. When read from cards, the data must be preceded by a standard NAME card and must end with an ENDATA card. When read from tape, no NAME or ENDATA card is required.

OUTPUT

The input data may include NAME cards other than the ones mentioned above. FMPS will ignore the NAME card and its associated data. However, a listing of this ignored data is produced on the output medium. It is listed shifted to the right beginning in print position 30.

The chapter headings, but not the associated data, which are processed by FMPS are listed on the output medium left-justified as they are read from the input stream.

SLACK INDICATORS ON ROWS CARDS

The row type is coded as shown for the ROW ID indicator card above. If cost rows are not specified with a blank slack indicator, the REVISE procedure must be called following the INPUT procedure to define the cost rows as nonrestraining.

REVISE DATA CARDS

In the control language program, a procedure REVISE modifies data previously processed by INPUT.

Essentially, the REVISE data deck is identical to the INPUT data deck. It is composed of the same six chapters of data: ROWS, SPRICES, COLUMNS, RHS, RANGES, and BOUNDS. However, only those chapters to be actually changed are included. Within each chapter, four types of revisions are possible:

MODIFY

DELETE

BEFORE

AFTER

These revisions are stated on data cards similar to those used for INPUT. First, the chapter to be revised is identified by a chapter indicator card. Kinds of changes to be made are then specified by REVISE control cards (MODIFY, DELETE, BEFORE, and AFTER) and by actual data cards composing the changes. This sequence is repeated for each section to be revised. The use of REVISE is subject to the following conditions.

1. Modifications may be made in any order subject to the rule forbidding splitting of modifications in a given vector.

2. If an existing nonzero element is to be changed to zero, it must be defined with the value of zero in the REVISE data deck.

3. Any new vector to be added must be given a name that is different from the name given to any old vector, even if that vector is to be deleted.

4. If an E-, L-, or G-type row is modified intoan N-type row, range elements in the row are automatically removed.

5. A modified row or bound element must be entirely redefined, that is, a row must have its type of constraint specified. A bound element must have both its lower and upper limits specified even if only one is modified.

6. To keep each individual modification in core, the REVISE deck should not include more than 100 data cards for any individual revision type (MODIFY, DELETE, etc.) within a chapter. If the deck is too large, the KMAJER interrupt is taken. If revisions are extensive enough to require more than 100 data cards for any individual revision type within a chapter, the revision data should be separated into individual decks of proper size, and one call for REVISE should be made for each deck. NAME and ENDATA cards must be inserted before and after each deck.

7. If a row is added by using BEFORE or AFTER in the ROWS section, values are entered in this row for existing columns by using MODIFY.

ROWS CARDS FOR REVISE

MODIFY The MODIFY chapter indicator card signifies that the row definition cards that follow redefine the existing type of row. The command word MODIFY is punched in columns 2 to 7, as in

```
 MODIFY
```

DELETE The DELETE chapter indicator card signifies that the data cards that follow contain the names of existing row (punched in columns 5 to 12) are to be deleted. DELETE is punched in columns 2 to 7, as in

```
 DELETE
```

BEFORE The BEFORE chapter indicator card signifies thatrowdefinition cards that followare tobe inserted before the row named in the indicator card (specified in columns 15 to 22). If no row is specified, the rows will be inserted before the first row. BEFORE is punched in columns 2 to 7. Hence, the card takes the form

```
 BEFORE    name
```

AFTER The AFTER chapter indicator card signifies that row definition cards that follow are to be inserted after the row named in the indicator card (specified in columns 15 to 22). If no row is specified, the rows will be inserted after the last row. AFTER is punched in columns 2 to 7. Hence, the card takes the form

```
 AFTER     name
```

SPRICES CARDS FOR REVISE

Slack prices for any new rows must be defined immediately following the SPRICES chapter indicator. The format of the data cards is the same as required by INPUT. Do not use BEFORE or AFTER indicators.

MODIFY The MODIFY indicator card signifies that the following data cards define new slack prices for existing slacks. All prices for an existing slack must be redefined, even if only one price is modified. MODIFY is punched in columns 2 to 7, as in

```
 MODIFY
```

COLUMNS CARDS FOR REVISE

MODIFY The MODIFY indicator card signifies that the following data cards redefine coefficients in existing columns and/or places coefficients in new rows of existing columns. All modified coefficents for the same column must be grouped together. The command word MODIFY is punched in columns 2 to 7, as in

```
 MODIFY
```

DELETE The DELETE indicator card signifies that the following data cards contain the names (in columns 5 to 12) of existing columns to be deleted from the matrix. DELETE is punched in columns 2 to 7, as in

```
 DELETE
```

BEFORE The BEFORE indicator card signifies that the following data cards define new matrix columns that are to be inserted in the matrix before the existing column named in the indicator card (specified in columns 15 to 22). If no column is specified, the newcolumnswill be inserted before the first existing column. BEFORE takes the form,

```
 BEFORE    name
```

AFTER The AFTER indicator card signifies that the following data cards define new matrix columns that are to be inserted in the matrix after the existing column named in the indicator card (specified in columns 15 to 22). If colums 15 to 22 are blank, the new columns will be inserted after the last existing column. AFTER is punched in columns 2 to 6. The form of the AFTER command is

```
    AFTER    name
```

RHS CARDS FOR REVISE

Revisions to the RHS chapter are the same as for the COLUMNS chapter with the exception that the name field (columns 15 to 22) of the BEFORE and AFTER indicator card refers to names of the RHS vectors.

RANGES CARDS FOR REVISE

Range values for new rows must be first. They may be introduced by BEFORE or AFTER, but neither is necessary.

MODIFY The MODIFY indicator card signifies that the following data cards redefine a range value on an existing row. MODIFY is punched in columns 2 to 7, as in

```
    MODIFY
```

DELETE The DELETE indicator card signifies that the following cards contain (in columns 5 to 12) the name of the row that is to have its range value removed. DELETE is punched in columns 2 to 7, as in

```
    DELETE
```

BOUNDS CARDS FOR REVISE

MODIFY The MODIFY indicator card signifies that the data cards that follow redefine the bounds on existing columns. Note that the bounds on any column must be restated completely. For example, if only the lower bound was being changed, any upper bound on that column must be restated. MODIFY is punched in columns 2 to 7, as in

```
    MODIFY
```

DELETE The DELETE indicator card signifies that the following data cards contain (in columns 5 to 12) the name of the existing column for which all bounds will be removed. DELETE is punched in columns 2 to 7, as in

```
    DELETE
```

BEFORE The BEFORE indicator card signifies that the data cards that follow define the bounds for new columns. The BEFORE card should be identical to the BEFORE card that defined the new columns in the COLUMNS chapter. BEFORE has the form

```
    BEFORE    name
```

AFTER The AFTER indicator card signifies that the data cards that follow define the bounds for new columns. The AFTER card should be identical to the AFTER card that defined the new columns in the COLUMNS chapter.

BASISIN/BASISOUT DATA CARDS

Data for the BASISIN procedure is the same as the output from the BASISOUT procedure. As with all data decks, the data is preceded by a NAME card and terminated by an ENDATA card. The general form of the data card is shown below.

<table>
<tr><th>Columns</th><th>Description</th></tr>
<tr><td>2-3</td><td>Field 1: two-letter indicator that specifies one of the following actions.
<table>
<tr><th>Code</th><th>Action</th></tr>
<tr><td>XU</td><td>Remove the variable named in Field 3 from the basis and set it at upper bound. Put the variable named in Field 2 in the basis.</td></tr>
<tr><td>XL</td><td>Remove the variable named in Field 3 from the basis and set it at lower bound. Put the variable named in Field 2 in the basis.</td></tr>
<tr><td>UL</td><td>Set the variable named in Field 2 at upper bound. Field 3 is ignored.</td></tr>
<tr><td>LL</td><td>Set the variable named in Field 2 at lower bound. Field 3 is ignored.</td></tr>
</table></td></tr>
<tr><td>5-12</td><td>Field 2: name 1.</td></tr>
<tr><td>15-22</td><td>Field 3: name 2.</td></tr>
<tr><td>25-36</td><td>Field 4: not used.</td></tr>
<tr><td>40-47</td><td>Field 5: not used.</td></tr>
<tr><td>50-61</td><td>Field 6: not used.</td></tr>
</table>

LL indicators are not necessary if the MODIFY parameter is not used on BASISIN since all variables will be automatically initialized to lower bound. BASISOUT will not output any LL indicators.

LOADLIST DATA CARDS

As with all data decks, LOADLIST data is preceded by a NAME card and terminated by an ENDATA card.

INDICATOR CARDS

The LOADLIST data deck consists of data cards grouped according to the type of data (names or masks) they contain. A group of cards containing the same type of data is called a chapter. The first card of a chapter is always an indicator card which identifies the type of data in that chapter. Indicator cards contain only one word (NAMES or MASKS, beginning in column 1) which specifies the type of data cards that follow.

DATA CARDS

Data cards are divided into ten 8-column fields. Field 1 is always blank. The ten fields of a data card are outlined below.

Columns	Description
1-8	Field 1: blank
9-16	Field 2: name or mask.
17-24	Field 3: name or mask.
25-32	Field 4: name or mask.
33-40	Field 5: name or mask.
41-48	Field 6: name or mask.
49-56	Field 7: name or mask.
57-64	Field 8: name or mask.
65-72	Field 9: name or mask.
73-80	Field 10: name or mask.

NAMES DATA CARDS

NAMES cards specify the names of rows or columns in the selection list. Each data card contains up to nine names in Fields 2 to 10. Field 1 is always blank. If a field other than 1 contains all blanks, it is ignored.

MASKS DATA CARDS

MASKS cards specify the masks for selecting rows or columns. Each data card contains up to nine masks in Fields 2 to 10. Field 1 is always blank. If a field other than 1 contains all blanks, it is ignored.

6. LINEAR PROGRAMMING OPERATING MODE

Use and operation of procedures in the linear programming mode will be described in this chapter. The procedures are presented in four logical phases.

1. Input
2. Optimization
3. Output
4. Preservation and Restoration

(Parametric programming, an optional procedure available for use in the linear programming operating mode, is described in Appendix A.)

INPUT PHASE

The input phase consists of two procedures, INPUT and REVISE. An outline of each is given in Table 10 below.

Table 10. Input Procedures

Procedure	Purpose
INPUT	Initially states the LP matrix.
REVISE	Makes revisions to the LP matrix.

INPUT The INPUT procedure specifies a linear programming matrix to FMPS. This procedure reads the input data and converts it into a compact internal representation on file MATRIX. The following internal files (see Table 7) must be defined before the call for INPUT.

1. MATRIX
2. INVERSE
3. UTIL1
4. UTIL2

Also, if INPUT's data are on file, the user's communication file must be defined too.

The input file may consist of more than one reel of tape. The primary input unit must be defined through the DEVICE and ATTACH procedures. The second unit will be the next reel specified in the BPM assign control command. The occurrence of a tape end-of-file on the input tape causes switching to the alternate input tape.

For example, consider the case where input consists of three reels of tape, numbered 104, 59, and 73. The user provides ASSIGN statements to mount tapes 104, 59, and 73 on the primary input unit in that order. He also provides a DEVICE and ATTACH statement to define the primary input unit, as in

```
!ASSIGN F:6, (DEVICE,MT),(INSN,104,59,73)...
:
CALL DEVICE ('TAPE6',TAPE,'F')
:
CALL ATTACH ('MYFILE','TAPE6',FORTRAN,OLD)
:
CALL INPUT (FILE,'MYFILE')
```

The data deck setup for the INPUT procedure is shown in Chapter 5.

The INPUT procedure will also accept input in the SHARE formats of other LP systems. These include 1108 LP data, LP/90/94 data, and CDM4 LP data. Chapter 5 contains detailed information about SHARE input formats.

The following CR variables must be initialized before the call for INPUT.

CR Variable	Explanation
ADATA	Contains the name of the data deck for data reading procedures such as INPUT and REVISE. Also used by data outputting procedures such as BASISOUT to name output data deck.
APBNAME	The name to be assigned to the LP problem.

Optional parameters for INPUT are given below.

Parameter	Explanation
SHARE	Indicates that the input is in SHARE format and not in standard FMPS format. If this parameter is not present, standard FMPS format is assumed.
'1108'	Input is in UNIVAC 1108 LP SHARE format. The quotation marks are required.
'LP90'	Input is in LP/90/94 SHARE format. The quotation marks are required.
'CDM4'	Input is in CDM4 SHARE format. The quotation marks are required.
'SPRICES'	Indicates that the slack prices chapter is present in the input data and is to be used. Used only with SHARE.
FILE	Indicates that the input data are to be found on file 'filename'. If the parameter is not used, INPUT data are assumed to be on the standard card input device.

Parameter	Explanation
'filename'	The symbolic name of the communication file on which the input data reside. The quotation marks are required.

The following interrupts may occur within INPUT.

Interrupt	Causes
KMAJER	1. Invalid parameter. 2. Input data not found. 3. Minimum required input not found (ROWS, COLUMNS, and RHS). 4. Undefined files. 5. Rows chapter exceeds available memory. 6. FILE 'filename' undefined.
KMINER	1. Duplicate columns. The duplicate column is ignored. 2. Duplicate element. The duplicate element is ignored. 3. Invalid indicator in ROWS or BOUNDS chapter. 4. Invalid combination of indicators in BOUNDS chapter. 5. Columns out of sort in BOUNDS chapter.
KIOER	1. An irrecoverable input/output error has occurred. 2. Insufficient storage allocated for internal files.

REVISE The REVISE procedure modifies a matrix according to the input data from the standard card input device or from an internal communication file. Any element of the matrix can be modified, deleted, or inserted.

REVISE requires that the matrix to be revised be currently loaded in the MATRIX file, and that all of the standard FMPS internal files be defined. Initial loading of the matrix may be performed by INPUT or RESTORE. Matrix information is not destroyed or modified during execution of any other procedure except for CRASH (see "Optimization Phase" later in this chapter), which may alter the bound status of certain variables and set certain equations nonrestraining if the MODIFY parameter is used. CR variable ADATA contains the name of the REVISE data deck or identification record name if the data is on file.

Calling the REVISE procedure causes the problem to be initialized to a slack basis. If REVISE is called at a stage of the problem where the basis is not a slack basis, it may be desirable to preserve the current basis (BASISOUT) prior to the call for REVISE, and to reinstate the current basis following the call for REVISE (BASISIN and INVERT).

The data card format is the same as for INPUT. Refer to Chapter 5 for information about data deck setup.

Optional parameters for REVISE are given below.

Parameter	Explanation
FILE	Indicates that the input data for REVISE are on the file 'filename'.
'filename'	The symbolic name of the communication file on which the input data resides.

The following interrupts may occur within REVISE.

Interrupt	Causes
KMAJER	1. Invalid parameter. 2. Input data not found. 3. Undefined files. 4. ROWS chapter exceeds available memory. 5. No matrix exists to REVISE.
KMINER	1. Duplicate columns. The duplicate column is ignored. 2. Duplicate element. The duplicate element is ignored. 3. Invalid indicator in ROWS or BOUNDS chapter. 4. Invalid combination of indicators in BOUNDS chapter. 5. Columns out of sort in BOUNDS chapter.
KIOER	1. An irrecoverable input/output error has occurred. 2. Insufficient storage allocated for internal files.

OPTIMIZATION PHASE

The optimization phase contains three procedures, OPTIMIZE, INVERT, and CRASH. An outline of each is given in Table 11 below.

Table 11. Optimization Procedures

Procedure	Purpose
OPTIMIZE	Attempts to find an optimal, feasible solution to the existing matrix.
INVERT	Restates the product form of the inverse in terms of the minimum number of transformation required to state the basis.
CRASH	Attempts to find a better initial basis.

OPTIMIZE The OPTIMIZE procedure attempts to find an optimal feasible solution to the linear programming model. If the model has no feasible solution or the solution is unbounded, OPTIMIZE causes the KNFS or KUBS interrupts to occur.

While the model is infeasible, OPTIMIZE uses a composite pricing (PI) vector. (Infeasibility is defined as the amount by which a basis variable is below its lower bound or above its upper bound.) The function of the composite PI vector is either to maintain or to move toward optimality while achieving feasibility. CR cell FCMPDJ is the compositing factor which determines the balance between the drive for optimality and/or feasibility. As an example, a value of 0.5 for FDMPDJ implies a balanced driving force between optimality and feasibility, while a value of 0.0 implies total disregard for optimality. When a balanced driving force is requested, OPTIMIZE systematically reduces FCMPDJ by 0.125 if the drive for feasibility is insufficient.

CR variable IIWGHT is used to weight individual infeasibilities. The standard setting for IIWGHT is 0, which implies that all infeasibilities are given equal weight. If IIWGHT is set to -1, individual infeasibilities are weighted by the amount by which they are infeasible. If IIWGHT is set to +1, individual infeasibilities are weighted by the reciprocal of the amount by which they are infeasible.

Setting IIWGHT equal to -1 during part of the first phase of OPTIMIZE (the phase which attempts to eliminate all infeasibilities) may help reduce the number of iterations required to arrive at a feasible solution. However, this may also cause the problem to cycle. Therefore, it is recommended that the use of IIWGHT = -1 be limited to a given number of iterations or to a time period. This is done by initializing CR variables IFREQA or ITIME and setting IIWGHT to zero or to +1 for the remainder of this phase of OPTIMIZE.

CR variable FEPSILON may be used to perturb zero RHS elements on degenerate problems. For "less-than" constraints, zero RHS elements are replaced with FEPSILON. For "greater-than" constraints, zero RHS elements are replaced with -FEPSILON.

Problems for which the OPTIMIZE iteration log shows a zero ACTIVITY value for a large number of iterations may benefit from such perturbation. This is effected by the following control program statements.

```
FEPSILON = 1.0D-5
CALL OPTIMIZE
FEPSILON = 0.0
CALL OPTIMIZE
```

The communication region variables utilized by OPTIMIZE are listed below. Of all the variables in the list, only ARHS, AOBJ, and FOBJWT must be initialized by the user prior to calling OPTIMIZE.

CR Variable	Explanation
ARHS	Name of the right-hand side.
AOBJ	Name of the objective row.
FOBJWT	The weight given to the objective function. Must be +1.0 for minization, -1.0 for maximization.
FCMPDJ	Factor used in determining effective DJ when infeasible, as in DJE = FCMPDJ * DJ + (1.0 -FCMPDJ) * DJI where DJE is the effective DJ of the column. DJ is the true DJ of the column. DJI is the DJ based on infeasibility removal qualities of column.
INCAND	Number of profitable candidates from which one is selected during pricing of the matrix. For example, if INCAND is 5, then from each group of five profitable columns, the most profitable is selected. If INCAND is zero, the system will attempt to choose the optimum set.
IIWGHT	Infeasibility weighting switch, according to codes shown below. Code — Meaning -1 — Weight by amount of infeasibility. 0 — All infeasibilities given equal weight. +1 — Weight by reciprocal of amount of infeasibility.
FEPSILON	The value used to replace zero right-hand-side elements of inequalities on degenerate problems. If the constraint is of the less-than type, a zero RHS element is replaced with FEPSILON. If the constraint is of the greater-than type, a zero RHS element is replaced with -FEPSILON.
FDJZT	DJ zero tolerance. If the absolute value of the reduced cost (DJ) is less than FDJZT, it is considered zero.
FINFZT	Infeasibility zero tolerance. If the absolute value of the amount of infeasibility is less FINFZT, the variable is considered feasible.
FMPIVT	Minimum pivot tolerance. During any optimization procedure (here, INVERT is not considered an optimization procedure), an element is not considered as potentially pivotal unless its absolute value is greater than FMPIVT.

CR Variable	Explanation
ILOGC	Iteration logging frequency on console typewriter.
ILOGP	Iteration logging frequency for standard printing device.
ILOGSS	On/Off switch for printing column selection messages during pricing of matrix.
IFREQI	Iteration frequency interrupt for inversion. The KINV interrupt will occur every IFREQI iterations (IFREQI ≥ 0).
IFREQA	Iteration frequency interrupt. If IFREQA is 0, no interrupt will occur. Otherwise, the KFREQA will occur every IFREQA iterations.
ITIME	The length of time, in minutes, before the KTIME interrupt will occur. The KTIME interrupt does not occur if KTIME is set to zero. Whenever the KTIME interrupt occurs, KTIME is set to zero. Time for KTIME is measured from the time of the last initialization of ITIME.
INVTIME	Switch controlling the KINV interrupt timing routine in the OPTIMIZE procedure. If INVTIME is 0, the timing routine is active and causes KINV interrupt at times such that the total optimization time tends to be minimum. If INVTIME is -1, the timing routine is not active.

The following interrupts may occur within OPTIMIZE.

Interrupt	Causes
KMAJER	1. AOBJ or ARHS undefined. 2. No matrix to optimize.
KIOER	1. Unrecoverable I/O error. 2. INVERSE file capacity exceeded.
KNFS	No feasible solution.
KUBS	Unbounded solution.
KINV	1. Inversion frequency (IFREQI) to be satisfied. 2. Correcting numerical errors. 3. Inverse exceeding file storage. 4. Clock control active. Corrective action requires calling the INVERT procedure.
KFREQA	User iteration frequency (IFREQA) satisfied.
KTIME	User-specified time increment reached.

Some possible difficulties that may occur during optimization, and some suggested cures are given below.

DEGENERACY

If many RHS coefficients are zero, the problem may be degenerate. Degenerate problems are characterized by an inability to reduce infeasibilities beyond a certain number during phase one, or an excessive number of iterations to arrive at the optimal solution.

The cure is crashing before calling for OPTIMIZE. Use of the MODIFY parameter in the call for CRASH is recommended. However, since this causes modification of the matrix data, one may have to save (using the SAVE procedure) the current matrix before calling for CRASH (MODIFY), preserve the optimal basis after optimization (BASISOUT), reload the original matrix by means of RESTORE, reload the optimal basis (BASISIN), and invert to the optimal basis (INVERT). This in effect cancels any changes made by CRASH to the matrix and allows subsequent execution of PARARHS or the use of an alternate RHS vector.

Another cure is to use RHS perturbation (FEPSILON).

PIVOT REJECTIONS

Exception messages printed by the OPTIMIZE and INVERT procedures indicate pivot rejections. Subsequently, the problem may become pseudo-infeasible, or pseudo-unbounded, or may become pseudo-optimal during phase two of OPTIMIZE. Also, the numerical accuracy may be impaired.

Generally, occasional pivot rejections during the OPTIMIZE procedure have no adverse effects. Pivot rejections during INVERT may result in some of the abnormalities listed above.

The following actions may correct pivot rejections:

1. Raise the value of the FABSZT and/or of the FRELZT tolerances: this tends to eliminate small terms from the matrix, thus making it more unlikely for a pivot to be small enough to be rejected. During computations, round-off errors may cause certain zero elements in the transformed matrix to be computed as very small values. Hence, the FABSZT and FRELZT tolerances should be set large enough so that resulting pseudo-values will not be chosen as pivot terms. Care must be taken not to use too large a value, since this could eliminate valid elements.

2. Lower the value of FMPIVT and FMINVT: during OPTIMIZE and INVERT, pivoting on very small elements may cause loss of numerical accuracy. To avoid this, elements

smaller than FMPIVT and FMINVT are rejected as pivot elements. Values that are too large for these tolerances may result in ignoring valid pivot terms, thereby causing unboundness or preventing feasibility.

3. Eliminate poor scaling of the matrix: scaling is adequate when the matrix coefficients are within two or three orders of magnitude of each other.

INVERT The INVERT procedure establishes the product-form inverse for the currently specified basis. To minimize the number of elements in the inverse and, therefore, reduce numerical rounding error and computation time, INVERT uses the most modern techniques in triangularization and sub-triangularization. INVERT may be called either explicitly by the user or as the result of the KINV interrupt.

Periodic calls to INVERT from OPTIMIZE help preserve numerical accuracy and reduce total optimization time. Such calls are automatically executed at suitable time intervals. Setting CR variable INVTIME to a negative value inhibits these automatic calls.

CR variable IFREQI, if set to a positive nonzero value, controls the maximum number of iterations that can occur between occurrences of the KINV interrupt. Exceptional conditions, such as the INVERSE procedure exceeding file storage, or loss of accuracy during OPTIMIZE, PARARHS, or PARAOBJ procedures, may also cause the KINV interrupt to occur.

In general, operating with INVTIME = 0 and IFREQI = 0 gives the best speed and accuracy. CR region variable FMINVT is used by INVERT as the minimum pivot tolerance. Elements are not considered pivotal if their value is smaller than FMINVT. FMINVT should be initialized to a value smaller than the value used for FMPIVT, the minimum pivot tolerance for OPTIMIZE.

The following interrupts may occur within INVERT.

Interrupt	Causes
KMAJER	1. No matrix defined.
	2. No basis to invert to.
KIOER	Irrecoverable input/output error.

CRASH The CRASH procedure attempts to find an initial basis structure that reduces infeasibility, reduces degeneracy, and that contains variables that must be basic at solution. In addition, any row that has no feasible solution is pointed out and a KNFS interrupt occurs.

In the following LP equation,

$$\sum A_{ij} X_j \pm S_i = RHS_i$$

the sign of the slack coefficient S_i is positive for equations of the type "less than" or "equal to", and negative for equations of the type "greater than". Both A_{ij} and S_i are referred to as elements. RHS_i is the right-hand-side coefficient.

The following messages may be printed during CRASH.

ROW xxxxxxxx DOMINATING. ROW SET NON-RESTRAINING (FREE).

This message is produced when row xxxxxxxx has a zero RHS and either no plus elements or no negative elements. Since this equation constrains all of the columns having elements in it to zero, CRASH will also fix all those columns at lower bound. This is equivalent to having specified the row as N (nonrestraining) in the ROWS chapter during INPUT.

SLACK ON ROW xxxxxxxx SET FREE.

This message is produced when the slack for row xxxxxxxx is the only plus element in the row. Therefore, the slack for this row must be basic. This is equivalent to having specified the row as N (nonrestraining) in the ROWS chapter during INPUT.

COLUMN yyyyyyyy SET FREE IN ROW xxxxxxxx.

This message is produced if the element in column yyyyyyyy is the only plus element in equality row xxxxxxxx and the RHS for this row is positive or zero, or if the element in column yyyyyyyy is the only minus element if row xxxxxxxx and the RHS for this row is zero. Column yyyyyyyy is entered into the basis in row xxxxxxxx. This is equivalent to having specified the column as FR (free) in the BOUNDS chapter during INPUT.

COLUMN yyyyyyyy FIXED AT LOWER BOUND.

This message is produced whenever a column has an element in a dominating row implying that it must be nonbasic. This is equivalent to having specified the column as FX (fixed at lower bound) in the BOUNDS chapter during INPUT.

A summary line is printed stating the number of rows set free (slack on rows must be basic), the number of columns set free (columns that must be basic), the number of fixed columns (columns that must be nonbasic), and the number of rows that have no feasible solution.

INVERT is automatically called by CRASH to invert to the basis described by CRASH.

If it is desired to have the free and fixed status applied to the MATRIX, the parameter MODIFY on the call for CRASH will effect this.

Crashing often results in a significant speed increase in the OPTIMIZE procedure if the problem is degenerate and MODIFY is specified. The CRASH execution time is generally negligible compared with the OPTIMIZE time.

If the right-hand-side parametric procedure is to be used later in the run, or if a successive case is run which is obtained from the current case by use of the REVISE procedure or by using other right-hand-sides, and the

MODIFY parameter is specified, the following sequence of operations is necessary.

1. Save the problem before calling for CRASH (call SAVE).
2. Save the optional basis after reaching the solution (CALL BASISOUT, FILE, 'filename').
3. Restore the original matrix (call RESTORE).
4. Restore the optimal basis (CALL BASISIN, FILE, 'filename').

Note that if parametric programming is to be used later in the run or other right-hand-sides are to be used, MODIFY should not be used since the free and fixed status assigned by CRASH will not be valid for another right-hand-side or for PARARHS.

The optional parameter for CRASH is given below.

Parameter	Explanation
MODIFY	Indicates that the free and fixed status of variables is to be made permanent in the MATRIX.

The following communication region variables must be initialized by the user prior to the call for CRASH.

CR Variable	Explanation
ARHS	Name of the right-hand-side.
AOBJ	Name of the cost row.

The following interrupts may occur within CRASH.

Interrupt	Causes
KMAJER	1. AOBJ or ARHS undefined. 2. No matrix to optimize.
KIOER	1. Irrecoverable input/output error. 2. File capacity exceeded.
KNFS	No feasible solution.

OUTPUT PHASE

The output phase contains five procedures, OUTPUT, SOLUTION, ERRORS, CONDITION, and GET. An outline of each is given in Table 12.

Table 12. Output Procedures

Procedure	Purpose
OUTPUT	Displays the matrix in various forms.
SOLUTION	Reports the solution values.
ERRORS	Examines errors in the solution.
CONDITION	Displays the condition of various FMPS regions and files.
GET	Retrieves solution information in the control language.

OUTPUT The OUTPUT procedure displays the entire matrix or a selected subset on the standard printing device, or files on the internal communications device. OUTPUT displays the entire original matrix in tabular form on the standard printing device. Referring to the LP equation formulations below,

$$A_{ij} X_i \pm S_i = RHS$$

$$C_i X_j \longrightarrow \text{Maximum}$$

The OUTPUT procedure displays the values of the following elements:

Coefficients A_{ij}

Coefficient S_i (value of 1 for the slack variable)

Right-Hand-Side values RHS

Cost coefficient C_i

The options of OUTPUT (described in Table 13) control the following display options:

1. Grouping of the coefficients: the coefficients can be grouped and displayed for each variable (matrix column), or for each equation (matrix row), or can be displayed on the printer form in such a way that they form the entire matrix when the printer pages are separated and reassembled together in a certain manner. The grouping by rows is generally the most compact way of displaying large LP matrices. The grouping in tableau format is only practical for small problems (less than 200 variables).

2. Representation of the coefficient values (numerical value) or symbol for order of magnitude.

3. Applicability of selection lists: output may be made to include or exclude all coefficients for specified rows or for rows the names of which match specified row masks or both, or for specified columns or for columns the names of which match specified column masks. If desired, row and column selection lists may be used in conjunction with each other to abstract further the printed output. Two special selection lists, LISTI and LISTU can also be used in this connection. LISTI identifies the set of all infeasible equations (rows) and LISTU identifies the set of all unbounded variables (columns) at the time of the call for OUTPUT.

4. Whether to display the original or current coefficients: referring to the simplex tableau, the original coefficients are the Coefficients, Right-Hand-Side Coefficients, Slack Coefficients, and objective Function coefficients for the initial tableau (all slack basis). Contrasted with this, the "current" coefficients are those for the simplex tableau corresponding to the current basis.

Output Medium: the report prepared by OUTPUT is directed to the standard printing device.

Table 13. Parameters for OUTPUT

Parameter	Output Device (PRINTER)	Function of Parameter
CURRENT	Optional	The requested elements of the matrix are premultiplied by the inverse to bring them up to date with the current basis.
CODED	Optional	Provides a condensed, coded picture of matrix tableau.
BYROWS	Optional	The nonzero elements of the row along with the names of the column in which they reside are displayed. (Matrix displayed row by row.)
BYCOLS	Optional	The nonzero elements of the column along with the names of the rows in which they reside are displayed. (Matrix displayed column by column.)
COUNTS		The name, type, and element count of each row, column, and RHS is printed according to the following codes. The type for a row is printed: Row Type Meaning N Nonrestraining E Equality G Greater than GR Greater than with a range L Less than LR Less than with a range The type for a column or RHS is printed: Row Type Meaning FX Fixed FR Free LO Lower bounded UP Upper bounded LU Lower and upper bounded
MATRIX		Outputs the matrix in card image form on the card punch or to a CARD communication file if the FILE, 'filename' parameters are specified. The contents of CR variable ADATA will be placed in columns 15 to 22 of the generated NAME card.
ROWS	Optional	Indicates that row selection or exception lists are to be used.
COLS	Optional	Indicates that column selection or exception lists are to be used.
EXCEPT	Optional	Indicates that the following parameter is a list reference and items in list are to be excepted from output.
LISTR	Optional	Used in connection with ROWS parameter to specify that LISTR contains the row selection or exception list.
LISTC	Optional	Used in connection with COLS parameter to specify that LISTC contains the column selection or exception list.

Table 13. Parameters for OUTPUT (cont.)

Parameter	Output Device (PRINTER)	Function of Parameter
LISTI	Optional	Used in connection with ROWS parameter to specify that the row selection list is composed of all infeasible rows.
LISTU	Optional	Used in connection with COLS parameter to specify that the column selection list is composed of unbounded columns.
FILE		Indicates that requested output be written on internal communication file (as well as printed).
'filename'		Used in connection with FILE parameter to specify 'filename' of internal communication file.

Notes:

Either BYROWS or BYCOLS must be specified, but not both.

Element values displayed are the original ones as loaded by INPUT unless the parameter CURRENT is specified.

Unless BYROWS or BYCOLS is specified, the matrix is displayed in tableau format.

Parameter ROWS, if specified, must always be part of one of the following parameter sequences:

```
ROWS, LISTR
ROWS, LISTI
ROWS, EXCEPT, LISTR
ROWS, EXCEPT, LISTI
```

This parameter specifies that only those elements in the rows specified in LISTR or LISTI are to be output or to be excluded from output.

Parameter COLS, if specified, must always be part of one of the following parameter sequences:

```
COLS, LISTC
COLS, LISTU
COLS, EXCEPT, LISTC
COLS, EXCEPT, LISTU
```

This parameter specifies that elements in the columns specified in LISTC or LISTU are to be output or excluded from output.

The following control program statements are useful in determining the cause of infeasibility or unboundedness if it occurs during CRASH, OPTIMIZE, PARAOBJ, or PARARHS:

```
C      INITIALIZE UNBOUNDEDNESS INTERRUPT
       CELL TO TRANSFER TO 500
       ASSIGN 500 TO KUBS
C      INITIALIZE INFEASIBILITY INTERRUPT CELL
       CELL TO TRANSFER TO 510
       ASSIGN 510 TO KNFS
       .
       .
       .
C      ENTRY FOR UNBOUNDED PROBLEM INTERRUPT
  500  CALL OUTPUT (BYCOLS, COLS, LISTU)
  503  CALL SOLUTION
       STOP
C      ENTRY FOR INFEASIBLE PROBLEM INTERRUPT
  510  CALL OUTPUT (BYROWS, ROWS, LISTI)
       GO TO 505
       .
       .
       .
```

In case of unboundedness, the matrix columns for the unbounded variables are output.

In case of infeasibility, the matrix rows for the infeasible constraints are output.

The following example illustrates the use of OUTPUT to display the original form of the elements in the rows specified in LISTR but not in the columns specified in LISTC.

```
CALL OUTPUT (BYROWS, ROWS, LISTR, COLS,
EXCEPT, LISTC)
```

The following interrupts may occur within OUTPUT

Interrupt	Causes
KMAJER	1. No matrix has been processed by INPUT. 2. There is no file with the name 'filename'.
KMINER	1. Null selection list. 2. Invalid parameters. 3. Illogical combination of parameters
KIOER	Irrecoverable input/output error.

SOLUTION The OPTIMIZE procedure does not automatically print the solution values when an optimal solution is reached. Its only purpose is to produce the optimal basis. Calling for the SOLUTION procedure allows the user to output the actual solution report.

The same mode of operation applies for parametric programming on the Right-Hand-Side and Cost row. Parametric procedures PARARHS and PARAOBJ create the basis for various values of the parameter FTHETAR and FTHETAC but do not print the solutions, this requires a call to SOLUTION.

Keeping the solution output function separate from the optimization or parametric procedures allows greater flexibility in the use of these procedures. Also, since the solution is called from the control program, tests may be programmed in the control program, using the IF statement to print the solution only under certain conditions. Additionally, several solution reports may be created for a given problem using different selection lists.

SOLUTION may also be used after a call to RESTORE, thereby printing the solution for a problem previously saved on a RESTART file, or after the sequence CALL BASISIN, CALL INVERT to output the solution pertaining to a user-specified basis.

The normal mode of SOLUTION is to print the solution on the standard printing device. If the optional parameter FILE is specified, the specified information is also placed on communication file 'filename'. In this case, the RCHAPTER and/or CCHAPTER parameters must be used to specify the columns of output to be filed.

SOLUTION output is prepared in two chapters, ROWS and COLUMNS. The ROWS chapter contains information on the selected rows in the matrix. The report contains nine columns of information. Table 14 describes each of the nine columns for the ROWS chapter. The COLUMNS chapter contains information on the selected columns in the matrix. The columns report contains eight columns which are described in Table 15.

If the FILE option is used, it is possible to file the data columns selectively in each chapter as well as select which rows and columns to output. Each data column has been assigned a number. Tables 14 and 15 list the numbers as well as the headings in each chapter.

The data columns are selected for filing by using the keyword parameters RCHAPTER and CCHAPTER, each followed by the numbers of the data columns to be filed.

Table 14. ROWS Chapter Column Description

Column	Heading	Description of Information in Column
1	NUMBER	The internal serial number associated with the row.
2	ROW	The name of the row (slack).
3	AT	A two-character code indicating status of row. Code — Meaning BS — Slack variable in basis and feasible. ** — Slack variable in basis and infeasible. EQ — Artificial slack variable, nonbasic. UL — Row at upper limit. LL — Row at lower limit.
4	ACTIVITY	Activity of row, that is, the original right-hand-side minus the activity of the slack.
5	SLACK ACTIVITY	Activity of slack variable.
6	LOWER LIMIT	Lowest activity that row may have.
7	UPPER LIMIT	Highest activity that row may have.
8	DUAL ACTIVITY	Otherwise known as simplex multiplier, or PI value for row.
9	SLACK PRICE	Slack price if specified during input. If slack is priced, reduced cost of slack is equal to the DUAL ACTIVITY + or - the SLACK PRICE, where + or - refers to minimizing or maximizing, respectively.

Table 15. COLUMNS Chapter Column Description

Column	Heading	Description of Information in Column
1	NUMBER	The internal serial number associated with column.
2	COLUMN	The name of the column.
3	AT	A two-character code indicating status of column. Code — Meaning BS — Column in basis and feasible. ** — Column in basis and infeasible. FR — Column basic and free. EQ — Column nonbasic and fixed. UL — Column nonbasic at upper limit. LL — Column nonbasic at lower limit.
4	ACTIVITY	The value of the column in the solution.
5	INPUT COST	The objective function coefficient of column.
6	LOWER LIMIT	Lowest activity column may have.
7	UPPER LIMIT	Highest activity column may have.
8	REDUCED COST	The DJ of the column. The rate of change in the objective value per unit change of the column. Note that the reduced cost of an upper-bounded variable at upper bound will be negative. It may also be negative on a fixed variable.

Chapter 2 describes the means of accessing the filed solution and the structure of each record.

The example shown below illustrates some uses of SOLUTION.

```
CALL SOLUTION (ROWS,LISTR,COLS,LISTC,
FILE,'SOLFILE',RCHAPTER,2,5,8,CCHAPTER,
2,4,8)
```

In the example, SOLUTION is used to perform the following tasks.

1. File the output on communication file 'SOLFILE' as well as on the printer.
2. File only the rows specified in row selection list LISTR.
3. File only the columns specified in column selection list LISTC.
4. File only the row name, slack activity, and dual activity columns of the ROWS chapter. (All columns appear on the printer report.)
5. File only the column name, activity, and reduced cost columns of the COLUMNS chapter. (All columns appear on the printer report.)

The optional parameters available to SOLUTION are given below.

Parameter	Explanation
ROWS	Indicates that row selection or exception list follows.
COLS	Indicates that column selection or exception list follows.
EXCEPT	Indicates that following list reference is exception list.
LISTR	Used in connection with ROWS to specify row selection or exception list.
LISTC	Used in connection with COLS to specify column selection or exception list.
FILE	Indicates that requested output be written on internal communication file 'filename'.
'filename'	Used in connection with FILE to specify 'filename'.
RCHAPTER	Indicates ROWS chapter data column selection numbers follow.
CCHAPTER	Indicates COLUMNS chapter data column selection numbers follow.

The following interrupts may occur within SOLUTION.

Interrupt	Causes
KMAJER	1. No matrix defined. 2. There is no file with name 'filename'.

Interrupt	Causes
	3. Data column selection indicated but specifications missing.
KMINER	1. Invalid parameter.
	2. Illogical combination of parameters.
KIOER	Irrecoverable input/output error.

ERRORS The ERRORS procedure substitutes the current primal and dual solutions into the original primal and dual problems and computes and outputs any rounding error that exists to the standard printing device. Any error less than the tolerance FABSZT is considered zero, and no line of print will occur.

The output is prepared in two sections. The first section contains the dual errors and consists of the following information.

1. Name of the basis variable.
2. Magnitude of error.

The second section contains the primal errors and consists of the following information.

1. Name of the row.
2. Right-hand-side value of row.
3. Magnitude of error.

The following interrupts may occur in ERRORS.

Interrupt	Causes
KMAJER	No matrix defined.
KIOER	Irrecoverable input/output error.

CONDITION The CONDITION procedure outputs to the standard printing device the following information:

1. Contents of communication region.
2. Current status of all active files.

GET The GET procedure allows the user to retrieve information about a row or column, and to alter his strategy in the control language. All or any part of the following items may be accessed on a call for GET.

Code	Meaning
UB	Upper bound
LB	Lower bound
CJ	Objective function coefficient
BI	Activity level
DJ	Reduced cost
ZJ	PI value

The general form of a call for GET is

```
CALL GET (NAME,op,FWxx,OP,FWxx,—. —)
```

where

NAME is the name of a row or column.

op is one of the codes listed above.

FWxx is a user working cell.

In addition to placing requested information in the specified working cells, GET also prints information on the standard printing device. The following example illustrates the use of GET to obtain the activity in FW01, the input cost in FW02, and the upper bound in column RUNCRUDE in FW03.

```
CALL GET ('RUNCRUDE',BI,FW01,CJ,
FW02,UB,FW03)
```

PRESERVATION/RESTORATION PHASE

The preservation/restoration phase contains four procedures, BASISOUT, SAVE, BASISIN, and RESTORE. An outline of each is given in Table 16 below.

Table 16. Preservation/Restoration Procedures

Procedure	Purpose
BASISOUT	Preserves the basis structure.
SAVE	Preserves the contents of data areas and files.
BASISIN	Restores a basis structure.
RESTORE	Restores the contents of data areas and files.

BASISOUT The BASISOUT procedures punches or files (FILE parameter) the current basis structure and bounds status. The punched or filed data deck is preceded by a NAME card which contains (in columns 15 to 22) the contents of CR cell ADATA. In addition, the data deck is followed by an ENDATA card.

The data deck produced by BASISOUT is in the correct format to be used as input data to the BASISIN procedure.

Chapter 5 describes the format of data cards produced by BASISOUT and required as input by BASISIN.

Optional parameters for BASISOUT are:

Parameter	Explanation
FILE	Indicates that the output is to be written on communication file 'filename'. If FILE is not specified, the output will be produced on the standard punch device.
'filename'	The symbolic name of a communication file.

The following interrupts may occur within BASISOUT:

Interrupt	Causes
KMAJER	1. No matrix defined.
	2. 'filename' undefined.
	3. Invalid parameter.
KIOER	Irrecoverable input/output error.

SAVE The SAVE procedure saves the contents of the communication region, the various internal work areas, and all internal files (MATRIX, INVERSE, etc.) on the tape file RESTART. Only one problem may be saved on the RESTART tape. Any number of SAVEs may be made to the same restart tape, but the last one overlays previous ones. If several SAVE files are desired, the tape unit for RESTART may be changed in the control program by a new ATTACH statement preceding the SAVE. Note that user working-storage and communication files are not saved.

The following interrupts may occur within SAVE.

Interrupt	Causes
KMAJER	1. RESTART file undefined.
	2. RESTART file not on a tape unit.
KIOER	Irrecoverable input/output error.

BASISIN The BASISIN procedure either inputs a new basis, or modifies the existing basis. Provision is made to allow both the specification of variables to be entered into the basis and the removal of variables at upper or lower bound. In addition, the user may specify which nonbasic variables are to be placed at upper or lower bound.

If the MODIFY parameter is used, the current basis will be used to process the input. Chapter 5 describes the format of the input cards. If the MODIFY parameter is not used, an all-slack basis will be used to process the input, and all variables will initially be set at lower bound.

A call for the INVERT procedure must be made following the BASISIN procedure.

The optional parameters for BASISIN are given below.

Parameter	Explanation
MODIFY	Indicates that the input data is to be processed against the current basis structure (instead of the slack basis).
FILE	Indicates that the input is on file 'filename' instead of the normal card reading device.
'filename'	The symbolic name of the input file.

The following interrupts may occur within BASISIN.

Interrupt	Causes
KMAJER	1. Invalid parameter.
	2. 'filename' undefined.
KIOER	Irrecoverable input/output error.

RESTORE The RESTORE procedure restores the data areas and internal files saved by SAVE from file RESTART. Note that any internal file restored by RESTORE must be defined prior to the call for RESTORE.

The following interrupts may occur within RESTORE.

Interrupt	Causes
KMAJER	1. RESTART file undefined.
	2. Internal file undefined.
	3. RESTART file not on a tape unit.
	4. Insufficient core available for restoring data areas.
KIOER	Irrecoverable input/output error.

7. SEPARABLE PROGRAMMING OPERATING MODE

Use and operation of procedures in the separable programming (SEP) operating mode will be described in this chapter. A general description of this operating mode is provided followed by descriptions of specific procedures. The procedures are presented in four logical phases.

1. Input
2. Optimization
3. Output
4. Preservation and Restoration

GENERAL DESCRIPTION OF SEP MODE

Separable programming provides the FMPS user with the capability of handling certain types of nonlinear functions.

The nonlinearities must comply with the following important restrictions:

1. A nonlinear function in n variables must be "separable" into the sum of n functions, each in terms of only one of these variables, as in

$$y = f(x_n) = f_1(x_1) + f_2(x_2) + \ldots + f_n(x_n)$$

2. Each of the n functions must be representable by a piece-wise linear approximation of that function. The graph of the function in Figure 7 is shown in solid lines, a piece-wise linear approximation of the function is shown in broken line.

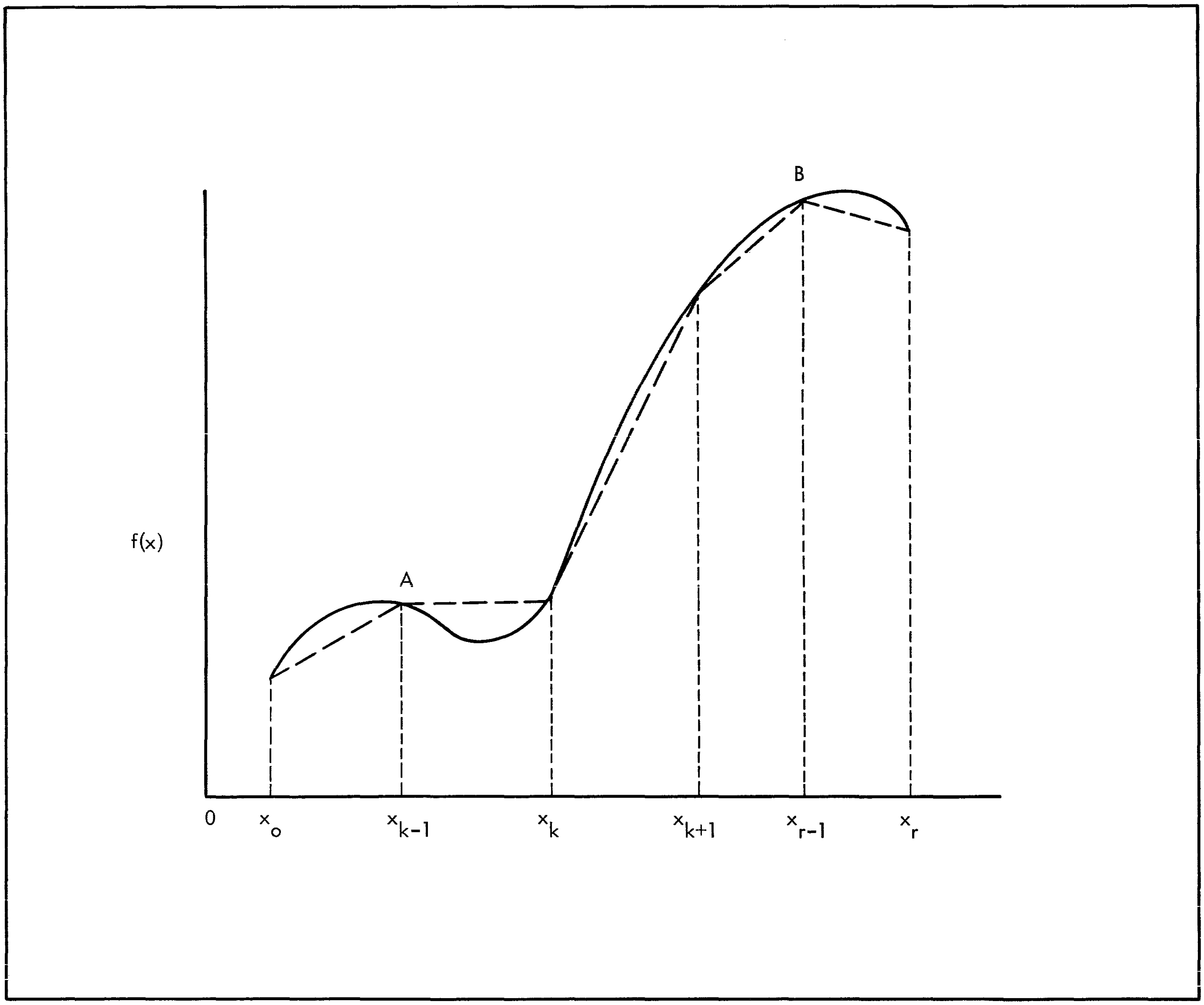


Figure 7. Piece-Wise Linear Approximation to a Separable Function

SEP ALGORITHM

A full description of the delta-method algorithm, together with a discussion of methods available to ensure that the problem complies with the above conditions, is found in Non-Linear and Dynamic Programming by G. Hadley.[†] Some details of this algorithm are outlined below.

1. Each variable x participating in a nonlinear function f(x) has associated with it a set of special variables. These special variables depict the piece-wise linear approximation to f(x); each special variable represents the distance progressed along some particular section of the piece-wise linear approximation. That is, dx_k is the kth of r special variables used to approximate f(x). It may be written as

$$dx_k = \frac{x - x_{k-1}}{x_k - x_{k-1}}$$

   where x_{k-1} and x_k are successive intercepts on the x coordinate (see Figure 7).

2. Each of the special variables has a lower bound of zero and an upper bound of 1. Their order specifies a direction along the x coordinate.

3. A special variable may become basic only if one of the adjacent variables is basic or the preceding variable is at upper bound. A bound shift is allowed only if the preceding variable is at upper bound. No two special variables in the same set may be basic at a given iteration.

4. The activity of the variable approximated is given by a grid equation of the form

$$x = x_0 + \underset{k=1}{\overset{r}{\text{sum}}} \Delta x_k \; . \; dx_k$$

   (See "Applicability of the SEP Algorithm" below.)

5. Any subset of the objective function and the problem constraints may be separable functions. A variable x may appear linearly in some functions and as a set of special variables approximating it in other functions. The user must only observe the requirements for establishing interrelationship.

PIECE-WISE LINEAR APPROXIMATION

Figure 7 shows a piece-wise linear approximation to some function f(x). This function is to be included in a set of equations for optimization. The function may be part of the objective or of some constraint. Note that the function is defined only over certain limits of x, that is,

$$x_0 \le x \le x_r$$

Special variables dx_1, dx_2, ... dx_r are now defined. These variables collectively form the set of special variables required to approximate f(x). The special variable dx_1 defines the interval between the two x intercepts x_0 and x_1; dx_2, the next interval between x_1 and x_2, and so on. The relationship is given by

$$x = x_0 + dx_1 (x_1 - x_0) + dx_2(x_1 - x_2) + \ldots + dx_r(x_r - x_{r-1})$$

or, simply,

$$x = x_0 + \underset{k=1}{\overset{r}{\text{sum}}} \Delta x_k \; . \; dx_k$$

where

$$0 \le dx_k \le 1$$

Δx_k are user-defined intervals along the x axis.

The Δx_k may be as small or as large as required, and may vary as necessary to obtain the user-required degree of approximation to any section of f(x).

The value of f(x) at x_0 is $f(x_0)$, at x_1 it is $f(x_1)$, and so on to $f(x_r)$ at x_r. Defining

$$\Delta f(x_k) = f(x_k) - f(x_{k-1}),$$

the relationship for f(x) along the first interval of the piece-wise linear approximation is obtained by equating

$$f(x) = f(x_0) + \Delta f(x_1) \; . \; dx_1$$

where

$$0 \le dx_1 \le 1$$

$$dx_2 = dx_3 = \ldots = dx_r = 0$$

This relationship can be extended to any point on the approximation, as in

$$f(x) = f(x_0) + \underset{k=1}{\overset{r}{\text{sum}}} \Delta f(x_k) \; . \; dx_k$$

This is a linear relationship in dx_k. If the dx_k are variables in the linear program, then this function may be included in the linear program as long as the following restriction is observed:

for

$$0 \le dx_k \le 1 \qquad \begin{array}{l} dx_0 = dx_1 = \ldots dx_{k-1} = 1 \\ dx_{k+1} = \ldots = dx_r = 0 \end{array}$$

[†] G. Hadley, Non-Linear and Dynamic Programming. Reading, Massachusetts: Addison-Wesley Publishing Company, 1964, Chapter 4.

The variable dx is the only variable in the set that may be basic. All other variables in the set are at upper or lower bound.

APPLICABILITY OF THE SEP ALGORITHM

There are two points, A and B, on the piece-wise linear approximation (Figure 7) from which the value of f(x) decreases irrespective of the direction along the x coordinate. Assuming that f(x) is an objective to be maximized, it is apparent that starting from [x, f(x)], the point A would be reached and the optimum would be indicated. However, A represents only a local optimum. The global optimum is point B. By starting at x_r, and proceeding in the opposite direction, point B is attained. The use of the SETBOUND procedure can assist in finding the global optimum in such cases, but there is no guarantee that an optimum attained using separable programming is the global optimum unless all functions have the appropriate properties of convexity and concavity.

The problem of local optima is also raised by separable nonconvex constraints. If the objective for the problem for which Figure 7 represents a constraint was z = x, then, depending on the direction in which x is moving, the algorithm may decide that A or B is the optimum.

EXAMPLES USING SEPARABLE PROGRAMMING

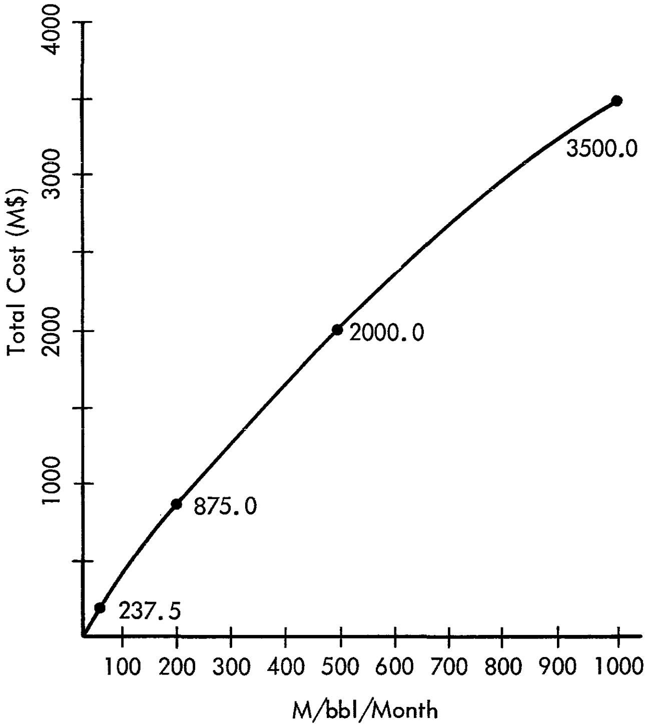
The following two problems illustrate the use of separable programming to model nonlinearities in the objective function and in a constraint.

NONLINEAR OBJECT FUNCTION

Volume-related discounts on a certain petrochemical feedstock are to be applied to the objective function according to the following table:

Volume, Mbbl/Month	$/bbl
0 - 50	$4.75
50 - 200	$4.25
200 - 500	$3.75
500 - 1000	$3.00

The total cost of feed, which is the amount by which the objective function should be decremented, varies with volume according to the following polygonal curve.



The pseudo costs associated with the four special variables entered into the problem are the difference in total cost found on this curve divided by the range of volume associated with the special variable. Those differences are $237.5, $637.5, $1125.0 and $1500 respectively. The matrix tableau would appear as follows.

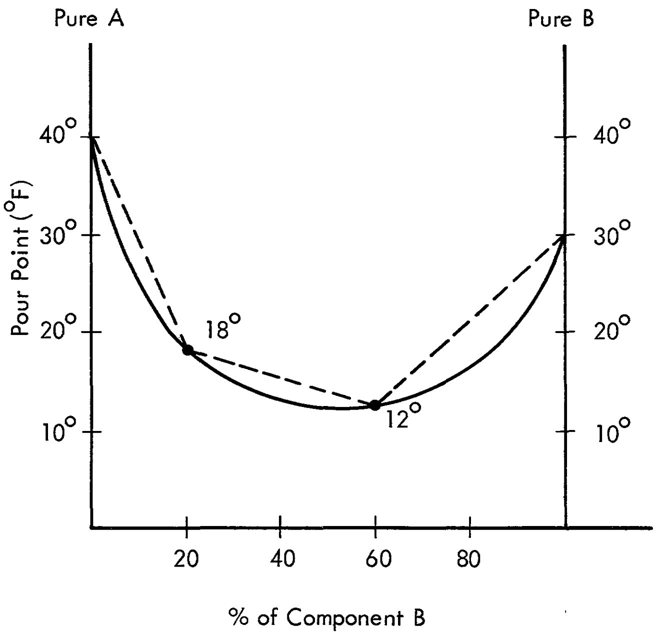
	Purchase Feedstock			
	SPVAR1 -237.5	SPVAR2 -637.5	SPVAR3 -1125.0	SPVAR4 -1500.0
Feedstock Material Balance	-50	-150	-300	-500

Note that the scaling of the special variables must be done manually and will affect all coefficients of the feed vector.

NONLINEAR CONSTRAINT

This example illustrates the use of separate programming to model a nonconvex specification row. Two products, A and B, are to be blended to meet a maximum pour-point specification of 20°F.

The Pour Point versus Mix Curve is illustrated below. To prepare the curve for modeling, an arbitrary choice of ranges is made for the separable segments. In this case, ranges are 0-20%, 20-60%, 60-100% of Component B.



It is assumed that we wish to make 10 Mbbls of the blend. One vector is used to represent 100% A, and three "delta" vectors are used to represent the addition of Component B, as shown in the following tableau.

Separable Set (Unscaled)					
	100A 0B	80A 20B	40A 60B	0A 100B	RHS
Upper Bound Row	10	10	10	10	
Material Balance on A	+1.0	-0.2	-0.4	-0.4	
Material Balance on B		+0.2	+0.4	+0.4	
Pour Point Maximum Specification	40°	-22°	-10°	+18°	≤200°
Note: Pour Point Maximum Specification is equal to specification multiplied by total volume, as in 20° x 10° = ≤200°.					

Since the input requires the separable set to be scaled to have upper bound of 1, multiply each vector by 10. This results in the final tableau below as entered in the problem.

Separable Set (Scaled)					
	100A 0B	80A 20B	40A 60B	0A 100B	RHS
Upper Bound Row	1	1	1	1	
Material Balance on A	+10.0	-2.0	-4.0	-4.0	
Material Balance on B		+2.0	+4.0	+4.0	
Pour Point Maximum Specification	400°	-220°	+100°	+180°	≤200°

The separable programming operating mode requires different internal treatments of the work matrix than the linear programming operating mode. There, it is necessary to set the mode of operation at the beginning of a run by means of the ENTER procedure.

The procedures in the separable programming operating mode are presented in four logical phases.

1. Input
2. Optimization
3. Output
4. Preservation and Restoration

Each phase will be explained in detail. Note that many procedures in the separable programming operating mode are identical to corresponding procedures in the linear programming operating mode. Descriptions of these procedures are repeated in this section for user convenience. A note at the beginning of each procedure indicates whether or not the procedure is identical to the corresponding linear programming procedure.

INPUT PHASE

The input phase consists of two procedures, INPUT, and REVISE. An outline of each is given in Table 17 below.

Table 17. SEP Input Procedures

Procedure	Purpose
INPUT	Accepts the initial statement of the SEP problem
REVISE	Makes revisions to the SEP problem

INPUT Except for the restrictions and conditions described in the following paragraphs, the INPUT procedure for the separable programming operating mode is the same as the INPUT procedure for the linear programming operating mode.

The INPUT procedure specifies a separable programming problem to FMPS. INPUT processes input data (in standard data card format only) and converts it into a compact internal representation on internal file MATRIX. The following internal files (see Table 7) must be defined before the call to INPUT.

1. MATRIX
2. INVERSE
3. UTIL1
4. UTIL2

Also, if INPUT's data are on file, the user's communication file must also be defined.

The data deck setup for the input procedure is shown in Chapter 5.

The special variables may appear in any row in the problem. They are identified as such in the COLUMNS chapter, and this identification is the only difference between separable and linear programming data. The 'MARKER' parameters are used to bracket each set of special variables. (The single quotation marks are included in the keywords.) There are two types of 'MARKER' cards distinguished by the keywords 'SEPORG' or 'SEPEND' in columns 40 to 47 of the 'MARKER' data card. The format of a 'MARKER' data card is shown below.

Columns	Description
1-4	Blank.
5-12	Unique column name.
13-14	Blank.
15-22	'MARKER'
23-39	Blank.
40-47	'SEPORG' or 'SEPEND'
48-72	Blank.

All of the special variables in a set must be contained between two 'MARKER' cards. A set may be embedded anywhere within the body of the matrix columns. The beginning of a new set is recognized when a 'SEPORG' type of 'MARKER' card is read. The name of the set is the name in columns 5 to 12 of the 'SEPORG' card which precedes the set. The end of a set is recognized when either a 'SEPEND' or 'SEPORG' type of 'MARKER' card with a unique name in columns 5 to 12 is processed. Contiguous sets do not require a 'SEPEND' type of 'MARKER' card as a separator.

Data cards describing the special vectors in a set have the same format as normal linear variables. The order of appearance of the variables in a set defines the required sequence $dx_1, \ldots, dx_r$.

Each of the separable special variables must have an upper bound of 1. This bound is automatically assigned to each of the special variables. The user may, if he so desires, include these bounds in the BOUNDS chapter. However, if any other bound besides this preempted bound is assigned, it will be registered as a minor error.

The following CR variables must be initialized before the call for INPUT.

CR Variable	Explanation
ADATA	Contains the name of the data deck for data reading procedures such as INPUT, REVISE, etc. Also used by data outputting procedures such as BASISOUT to name output data deck.
APBNAME	The name to be assigned to the SEP problem.

Optional parameters for INPUT are

Parameter	Explanation
FILE	Indicates that the input data is to be found on file 'filename'. If the parameter is not used, INPUT data is assumed to be on the standard card input device.
'filename'	The symbolic name of the communication file on which the input data resides.

The following interrupts may occur with INPUT.

Interrupt	Causes
KMAJER	1. Invalid parameter. 2. Input data not found. 3. Minimum required input not found (ROWS, COLUMNS, and RHS). 4. Undefined files. 5. Rows chapter exceeds available memory. 6. FILE 'filename' undefined. 7. Invalid 'MARKER' card.

Interrupt	Causes
KMINER	1. Duplicate columns. The duplicate column is ignored.
	2. Duplicate element. The duplicate element is ignored.
	3. Invalid indicator in ROWS or BOUNDS chapter.
	4. Invalid combination of indicators in BOUNDS chapter.
	5. Columns out of sort in BOUNDS chapter.
	6. Illegal bound for a special variable. The illegal bound is ignored.
KIOER	An irrecoverable input/output error has occurred.

REVISE This procedure is identical to the corresponding procedure in the linear programming mode.

The REVISE procedure modifies a matrix according to the input data from the standard card input device or from an internal communication file. Any element of the matrix can be modified, deleted, or inserted. REVISE requires that the matrix to be revised be currently input and that all of the standard FMPS internal files be defined. Communication region variable ADATA contains the name of the REVISE data deck or identification record name if data are on file. New sets of special variables must be bracketed by the required 'MARKER' cards.

It is mandatory (unless a slack starting basis is desired) that a BASISIN procedure and an INVERT procedure follow REVISE to resume from an advanced base.

The data card format is the same as for INPUT. Refer to Chapter 5 for information about data deck setup.

Optional parameters for REVISE are given below.

Parameter	Explanation
FILE	Indicates that the input data for REVISE is on the file 'filename'.
'filename'	The symbolic name of the communication file on which the input data resides.

The following interrupts may occur within REVISE.

Interrupt	Causes
KMAJER	1. Invalid parameter.
	2. Input data not found.
	3. Undefined files.
KMAJER (cont.)	4. ROWS chapter exceeds available memory.
	5. No matrix exists to REVISE.
	6. Invalid 'MARKER' card.
KMINER	1. Duplicate columns. The duplicate column is ignored.
	2. Duplicate element. The duplicate element is ignored.
	3. Invalid indicator in ROWS or BOUNDS chapter.
	4. Invalid combination of indicators in BOUNDS chapter.
	5. Columns out of sort in BOUNDS chapter.
	6. Illegal bound for a special variable. The illegal bound is ignored.
KIOER	An irrecoverable input/output error has occurred.

SEP OPTIMIZATION PHASE

The optimization phase contains three procedures in the separable programming operating mode, OPTIMIZE, INVERT, and SETBOUND. An outline of each is given in Table 18 below.

Table 18. SEP Optimization Procedures

Procedure	Purpose
OPTIMIZE	Attempts to find optimal, feasible solution to the existing matrix while ensuring that the special variables comply with their basic entry rules.
INVERT	Restates the product form of the inverse in terms of the minimum number of transformations required to state the basis.
SETBOUND	Tries different solution paths by setting the special variables in specified sets to bound.

OPTIMIZE OPTIMIZE is similar to the LP OPTIMIZE, except that in the SEP operating mode, the CR variable INCAND is not available for user setting.

The OPTIMIZE procedure attempts to find a feasible optimal solution to the separable programming matrix using the

SEP algorithm. If the matrix has no solution, or if the solution is unbounded, OPTIMIZE will cause the KNFS or KUBS interrupts to occur.

While the model is infeasible, OPTIMIZE uses a composite pricing (PI) vector. The function of the composite PI vector is either to maintain or to move toward optimality while achieving feasibility. Communication region cell FCMPDJ is the compositing factor which determines the balance between the drive for optimality and/or feasibility. As an example, a value of 0.5 for FCMPDJ implies a balanced driving force between optimality and feasibility while a value of 0.0 implies total disregard for optimality. When a balanced driving force is requested, OPTIMIZE systematically reduces FCMPDJ by 0.125 if the drive for feasibility is insufficient. FCMPDJ will be reduced if only one candidate from the selected subset is chosen to enter the basis, and the sum of infeasibilities is not decreasing.

Communication region variable IIWGHT is used to weight individual infeasibilities. The standard setting for IIWGHT is 0, which implies all infeasibilities are given equal weight. If IIWGHT is set to -1, individual infeasibilities are weighted by the amount by which they are infeasible. If IIWGHT is set to +1, individual infeasibilities are weighted by the reciprocal of the amount by which they are infeasible.

The communication region variables utilized by OPTIMIZE are listed below. Of all the cells in the list, only ARHS, AOBJ, and FOBJWT must be initialized by the user prior to calling OPTIMIZE.

CR Variable	Explanation
ARHS	Name of the right-hand side.
AOBJ	Name of the objective row.
FOBJWT	The weight given to the objective function. Must be +1.0 for minimization, -1.0 for maximization.
FCMPDJ	Factor used in determining effective DJ when infeasible, as in DJE = FCMPDJ * DJ + (1.0-FCMPDJ) * DJI where DJE is the effective DJ of the column. DJ is the true DJ of the column. DJI is the DJ based on infeasibility removal qualities of the column.
IIWGHT	Infeasibility weighting switch, according to codes shown below.

CR Variable	Explanation	
IIWGHT (cont.)	Code	Meaning
	-1	Weight by amount of infeasibility.
	0	All infeasibilities given equal weight.
	+1	Weight by reciprocal of amount of infeasibility.
FDJZT	DJ zero tolerance. If the absolute value of the reduced cost (DJ) is less than FDJZT, it is considered zero.	
FINFZT	Infeasibility zero tolerance. If the absolute value of the amount of infeasibility is less than FINFZT, the variable is considered feasible.	
FMPIVT	Minimum pivot tolerance. During any optimization procedure (here, INVERT is not considered an optimization procedure), an element is not considered as potentially pivotal unless its absolute value is greater than FMPIVT.	
ILOGP	Iteration logging frequency for console printer.	
ILOGSS	On/Off switch for printing column selection messages during pricing of matrix.	
IFREQI	Iteration frequency interrupt for inversion. The KINV interrupt will occur every IFREQI iterations (IFREQI > 0).	
IFREQA	Iteration frequency interrupt. If IFREQA is 0, no interrupt will occur. Otherwise, the KFREQA interrupt will occur every IFREQA iterations.	
ITIME	The length of time, in minutes, before the KTIME interrupt will occur. The KTIME interrupt does not occur if ITIME is set to zero. Whenever the KTIME interrupt occurs, ITIME is set to zero. Time for KTIME is measured from the time of the last initializatior of ITIME.	

The following interrupts may occur within OPTIMIZE.

Interrupt	Causes
KMAJER	1. AOBJ or ARHS undefined.
	2. No matrix to optimize.

Interrupt	Causes
KIOER	1. Irrecoverable input/output error. 2. File capacity exceeded.
KNFS	No feasible solution.
KUBS	Unbounded solution.
KINV	1. Inversion frequency (IFREQI) satisfied. 2. Correcting numerical errors. 3. Inverse exceeding file storage. Corrective action requires calling the INVERT procedure.
KFREQA	User iteration frequency (IFREQA) satisfied.
KTIME	User-specified time increment reached.

INVERT This procedure is identical to the corresponding procedure in the linear programming mode.

The INVERT procedure establishes the product-form inverse for the currently specified basis. To minimize the number of elements in the inverse and, therefore, reduce numerical rounding error and computation time, INVERT uses the most modern techniques in triangularization and subtriangularization. INVERT may be either called explicitly by the user or called as the result of the KINV interrupt.

Periodic calls to INVERT from OPTIMIZE help preserve numerical accuracy and reduce total optimization time. Such calls are automatically executed at suitable time intervals. Setting CR variable INVTIME to a negative value inhibits these automatic calls.

CR variable IFREQI, if set to a positive nonzero value, controls the maximum number of iterations that can occur between occurrences of the KINV interrupt. Exceptional conditions such as the INVERSE procedure exceeding file storage, or loss of accuracy during OPTIMIZE, PARARHS, or PARAOBJ procedures may also cause the KINV interrupt to occur.

In general, operating with INVTIME = 0 and IFREQI = 0 gives the best speed and accuracy. CR region variable FMINVT is used by INVERT as the minimum pivot tolerance. Elements are not considered pivotal if their value is smaller than FMINVT. FMINVT should be initialized to a value smaller than the value used for FMPIVT, the minimum pivot tolerance for OPTIMIZE.

The following interrupts may occur within INVERT.

Interrupt	Causes
KMAJER	1. No matrix defined. 2. No basis to invert to.
KIOER	Irrecoverable input/output error.

SETBOUND The SETBOUND procedure may be called at any stage of problem solution, provided that a matrix exists on the file MATRIX. Due to the possibility of obtaining a local optimum to a problem (depending on the solution path taken), it is of interest to examine the solutions obtained by proceeding along different paths. SETBOUND provides this capability.

Independent of problem status, SETBOUND will set all the special variables in the sets specified to upper bound.

The two possible calls to SETBOUND are

CALL SETBOUND

and

CALL SETBOUND (LISTC)

The first of these calls will result in all the special variables in all the sets being set to upper bound.

The second call will result in all the special variables in those sets listed in a previously loaded column selection list (see LOADLIST) being set to upper bound. The sets required are specified by including the column name given on the 'SEPORG' type of 'MARKER' card in the list of names in the column selection list.

For example, if a set of special variables is preceded in the INPUT data by a card with the format outlined below,

Columns	Description
5-12	FIRSTSET
13-14	Blank.
15-22	'MARKER'
23-39	Blank.
40-47	'SEPORG'

and the name FIRSTSET is included in the LOADLIST data, then the call

CALL SETBOUND (LISTC)

will set all the special variables in the set bracketed by the above and the next 'MARKER' card to upper bound. All other special variables will remain at their previous bound setting.

Note that if LISTC is specified and no list is set up, then all special variables will be set to bound.

Optional parameters for SETBOUND are given below.

Parameter	Explanation
LISTC	Indicates that a previously established column selection list should be searched for the set names of the variables to change bound.

The following interrupts may occur within SETBOUND.

Interrupt	Causes
KMAJER	No matrix setup.
KMINER	No selection list setup and optional parameter specified.
KIOER	An irrecoverable input/output error has occurred.

OUTPUT PHASE

The output phase contains four procedures, OUTPUT, SOLUTION, ERRORS, and CONDITION.

An outline of each is given in Table 19 below.

Table 19. SEP Output Procedures

Procedure	Purpose
OUTPUT	Displays the matrix in various forms.
SOLUTION	Reports the solution values.
ERRORS	Examines the errors in the solution.
CONDITION	Displays the condition of various FMPS regions and files.

Note that, except where explicitly noted, the 'MARKER's are not included in any of the output generated by the following procedures.

OUTPUT This procedure is identical to the corresponding procedure in the linear programming operating mode.

The OUTPUT procedure displays the entire matrix of a selected subset on the standard printing device, or files on the internal communications device. OUTPUT displays the entire original matrix in tabular form on the standard printing device.

Parameters for OUTPUT make it possible to:

1. Display updated elements.
2. Select specific rows and/or columns.
3. Output nonzero elements only.
4. File results.

Table 13 in Chapter 6 contains a complete list of parameters for OUTPUT.

The filed output consists of two logical records. The first, the identification record, is labeled OUTPUT and is followed by the second record containing the selected data. Chapter 2 describes the basic means of accessing the filed records in FORTRAN and lists the detailed structure of each record.

The following interrupts may occur within OUTPUT.

Interrupt	Causes
KMAJER	1. No matrix has been processed by INPUT.
	2. There is no file with the name 'filename'.
KMINER	1. Null selection list.
	2. Invalid parameter(s).
	3. Illogical combination of parameters.
KIOER	Irrecoverable input/output error.

The following example illustrates the use of OUTPUT to display the original form of the elements in the rows specified in LISTR but not in the columns specified in LISTC.

```
CALL OUTPUT (BYROWS,ROWS,LISTR,COLS,
     EXCEPT,LISTC)
```

SOLUTION SOLUTION output for the separable programming operating mode is prepared in three sections: the IDENTIFIER section, the ROWS section, and the COLUMNS section. The IDENTIFIER section is for display of problem status and indicates the operating mode. The ROWS and the COLUMNS sections are the same as for the linear programming operating mode with one addition in the COLUMNS section. The column names of the 'MARKER' cards will be included in the column name list in the position they had in the INPUT data column order. These names mark off each set of special variables, and have no entries against them. If the user requires the activity of the variable x approximated by the $dx_1 \ldots, dx_r$, he must include the grid equation (see "SEP Algorithm", above) in the problem.

The SOLUTION procedure prepares the current solution of the separable programming matrix for display. The normal mode of SOLUTION is to print the solution on the standard printing device. If the optional parameter FILE is used, the specified information is placed on internal communication file 'filename'.

SOLUTION output is prepared in three chapters for the separable programming operating mode. The first, the IDENTIFIER chapter, is for display of problem status. The second, the ROWS chapter, contains information on the selected rows in the matrix. The report contains nine columns of information. The COLUMNS chapter contains information on the selected columns in the matrix. The COLUMNS report contains eight columns.

If the FILE option is used, it is possible to file the data columns selectively in each chapter, as well as to select which rows and columns to output. Each data column has been assigned a number.

Table 14 in Chapter 6 describes the nine columns of the row report. Table 15 in the same chapter describes the 8 columns of the columns report. These tables also indicate the number and the heading assigned to each data column.

The data columns are selected for filing by using the keyword parameters RCHAPTER and CCHAPTER, each followed by the numbers of the data columns to be filed.

Chapter 2 describes the means of accessing the filed solution and the structure of each record.

The example shown below illustrates some uses of SOLUTION.

```
1  CALL SOLUTION  (ROWS, LISTR, COLS, LISTC,
         FILE, 'SOLFILE', RCHAPTER,2,5,8,CCHAPTER,
         2,4,8)
```

In the example, SOLUTION is used to perform the following tasks:

1. File the output on communication file 'SOLFILE' as well as on the printer.
2. File only the rows specified in row selection list LISTR.
3. File only the columns specified in column selection list LISTC.
4. File only the row name, slack activity, and dual activity columns of the ROWS chapter. All columns appear on the printed report.
5. File only the column name, activity, and reduced cost columns of the columns chapter. All columns appear on the printed report.

The optional parameters available to SOLUTION are given below.

Parameter	Explanation
ROWS	Indicates that row selection or exception list follows.
COLS	Indicates that column selection or exception list follows.
EXCEPT	Indicates that following list reference is exception list.
LISTR	Used in connection with ROWS to specify row selection or exception list.
LISTC	Used in connection with COLS to specify column selection or exception list.
FILE	Indicates that requested output be written on internal communication file 'filename'.
'filename'	Used in connection with FILE to specify 'filename'.
RCHAPTER	Indicates ROWS chapter data column selection numbers follow.
CCHAPTER	Indicates COLUMNS chapter data column selection numbers follow.

The following interrupts may occur within SOLUTION.

Interrupt	Causes
KMAJER	1. No matrix defined. 2. There is no file with name 'filename'. 3. Data column selection indicated but specifications missing.
KMINER	1. Invalid parameter. 2. Illogical combination of parameters.
KIOER	Irrecoverable input/output error.

ERRORS This procedure is identical to the corresponding procedure in the linear programming operating mode.

The ERRORS procedure substitutes the current primal and dual solutions into the original primal and dual problems and computes and outputs any rounding error that exists to the standard printing device. Any error less than the tolerance FABSZT is considered zero, and no line of print will occur.

The output is prepared in two sections. The first section contains the dual errors and consists of the following information.

1. Name of the basis variable.
2. Magnitude of error.

The second section contains the primal errors and consists of the following information.

1. Name of the row.
2. Right-hand-side value of row.
3. Magnitude of error.

The following interrupts may occur in ERRORS.

Interrupt	Causes
KMAJER	No matrix defined.
KIOER	Irrecoverable input/output error.

CONDITION This procedure is identical to the corresponding procedure in the linear programming operating mode.

The CONDITION procedure outputs to the standard printing device the following information.

1. Contents of communication region.
2. Current status of all active files.
3. Current status of all assigned input/output devices.
4. Amount of storage (words) in use by each file.
5. Maximum amount of storage used in the run by each file.

SEP PRESERVATION/RESTORATION PHASE

The preservation/restoration phase contains four procedures, BASISOUT, SAVE, BASISIN, and RESTORE. An outline of each is given in Table 20 below.

Table 20. SEP Preservation/Restoration Procedures

Procedure	Purpose
BASISOUT	Preserves the basis structure.
SAVE	Preserves the contents of data areas and files.
BASISIN	Restores a basis structure.
RESTORE	Restores the contents of data areas and files.

These procedures are identical to the corresponding procedures in the linear programming operating mode.

BASISOUT The BASISOUT procedure punches or files (FILE parameter) the current basis structure and bounds status. The punched or filed data deck is preceded by a NAME card which contains (in columns 15 to 20) the contents of CR cell ADATA. In addition, the data deck is followed by an ENDATA card.

The data deck produced by BASISOUT is in the correct format to be used as input data to the BASISIN procedure. Chapter 5 describes the format of data cards produced by BASISOUT and required as input by BASISIN.

The optional parameters for BASISOUT are

Parameter	Explanation
FILE	Indicates that the output is to be written on communication file 'filename'. If FILE is not specified, the output will be produced on the standard punch device.
'filename'	The symbolic name of a communication file.

The following interrupts may occur within BASISOUT.

Interrupt	Causes
KMAJER	1. No matrix defined. 2. 'filename' undefined. 3. Invalid parameter.
KIOER	Irrecoverable input/output error.

SAVE The SAVE procedure saves the contents of the communication region, the various internal work areas, and all internal files (MATRIX, INVERSE, etc.) on the tape file RESTART. Note that user working-storage, and communication files are not saved.

The following interrupts may occur within SAVE.

Interrupt	Causes
KMAJER	1. RESTART file undefined. 2. RESTART file not on a tape unit.
KIOER	Irrecoverable input/output error.

BASISIN The BASISIN procedure either inputs a new basis or modifies the existing basis. Provision is made to allow both the specification of variables to be entered into the basis and the removal of variables at upper or lower bound. In addition, the user may specify which nonbasic variables are to be placed at upper or lower bound.

If the MODIFY parameter is used, the current basis will be used to process the input. Chapter 5 contains the format of the input cards. If the MODIFY parameter is not used, an all-slack basis will be used to process the input and all variables will initially be set at lower bound. A call for the INVERT procedure must be made following the BASISIN procedure.

The optional parameters for BASISIN are given below.

Parameter	Explanation
MODIFY	Indicates that the input data is to be processed against the current basis structure (instead of the slack basis).
FILE	Indicates that the input is on file 'filename' instead of the normal card reading device.
'filename'	The symbolic name of the input file.

The following interrupts may occur within BASISIN.

Interrupt	Causes
KMAJER	1. Invalid parameter. 2. 'filename' undefined.
KIOER	Irrecoverable input/output error.

Note that basis specifications which conflict with the rules for basic and upper bounded variable (see "SEP Algorithm", above) selection will be resolved by ignoring invalid specifications.

RESTORE The RESTORE procedure restores the data areas and internal files saved by SAVE from file RESTART. Note that any internal file restored by RESTORE must be defined prior to the call for RESTORE.

The following interrupts may occur within RESTORE.

Interrupt	Causes
KMAJER	1. RESTART file undefined. 2. Internal file undefined. 3. RESTART file not on a tape unit. 4. Insufficient core available for restoring data areas.
KIOER	Irrecoverable input/output error.

8. OPERATING PROCEDURES

This chapter includes a description of the BPM control cards necessary for FMPS runs, and the relationship between BPM !ASSIGN control cards and FMPS control language CALL DEVICE statements. Also included are guidelines for the efficient use of FMPS. The user should reference the SIGMA 5/7 Batch Processing Monitor Reference Manual for complete discussion of BPM control cards. Error messages and error types are given in Appendix B.

BPM CONTROL COMMANDS USED IN FMPS RUNS

Figure 8 illustrates the general deck sequence for an FMPS run. The run deck always starts with a set of BPM control cards. Following the !DATA control card are the user's FMPS control language program terminated by an END statement and input data decks. Each input data deck is preceded by a NAME card and followed by an ENDATA card.

ASSIGN AND CALL DEVICE INTERACTION

The interrelationships between !ASSIGN control card parameters and the arguments in the CALL DEVICE control language statement are shown in the following examples.

In the command

```
     CALL DEVICE('EXAMPLE',TAPE,'E')
```

the keyword TAPE dictates an !ASSIGN control card which establishes a RAD file, labeled or unlabeled tape, and specifies that file or tape organization be consecutive-sequential (see Table 21).

In the command

```
     CALL DEVICE('EXAMPLE2', DISC, 'C')
```

the keyword DISC dictates an !ASSIGN control card which establishes a RAD file, and specifies that file organization be keyed direct-access (see Table 22).

The user should note that the compiled FMPS control language statements are written to a file or tape using the F:1 DCB. A BPM !ASSIGN control card must be in each run deck for F:1, and the organization must be consecutive-sequential. The control language compiler within FMPS simulates the following pair of control language statements.

```
     CALL DEVICE('PREPDEVI', TAPE, 'A')
     CALL ATTACH('PREPOUT', 'PREPDEVI')
```

The (INOUT) clause should be included in !ASSIGN control cards for all FMPS internal files and user communication files to assure the ability to read and write the file.

Should the user wish to save the RESTART tape after using the CALL SAVE procedure in an FMPS run, the (SAVE) clause should be included on the !ASSIGN control card associated with the tape.

Note that all FMPS internal files and user FORTRAN communication files are binary files; the !ASSIGN control card should have the (BIN) clause included.

Table 21. Consecutive-Sequential File Assignments

FMPS Control Language Statement CALL DEVICE('EXAMPLE', TAPE, 'E')	
Acceptable BPM !ASSIGN Control Cards	
RAD File	!ASSIGN F:5, (FILE, EXAMP), (CONSEC), (SEQUEN)...
Labeled Tape	!ASSIGN F:5, (LABEL, EXAMP), (CONSEC), (SEQUEN)...
Unlabeled Tape	!ASSIGN F:5, (DEVICE, 9T), (CONSEC), (SEQUEN)...

Table 22. Direct-Access File Assignments

FMPS Control Language Statement CALL DEVICE('EXAMPLE2', DISC, 'C')	
Acceptable BPM !ASSIGN Control Card	
RAD File	!ASSIGN F:3, (FILE, EXAM2), (KEYED), (DIRECT)...

EFFICIENT USE OF FMPS

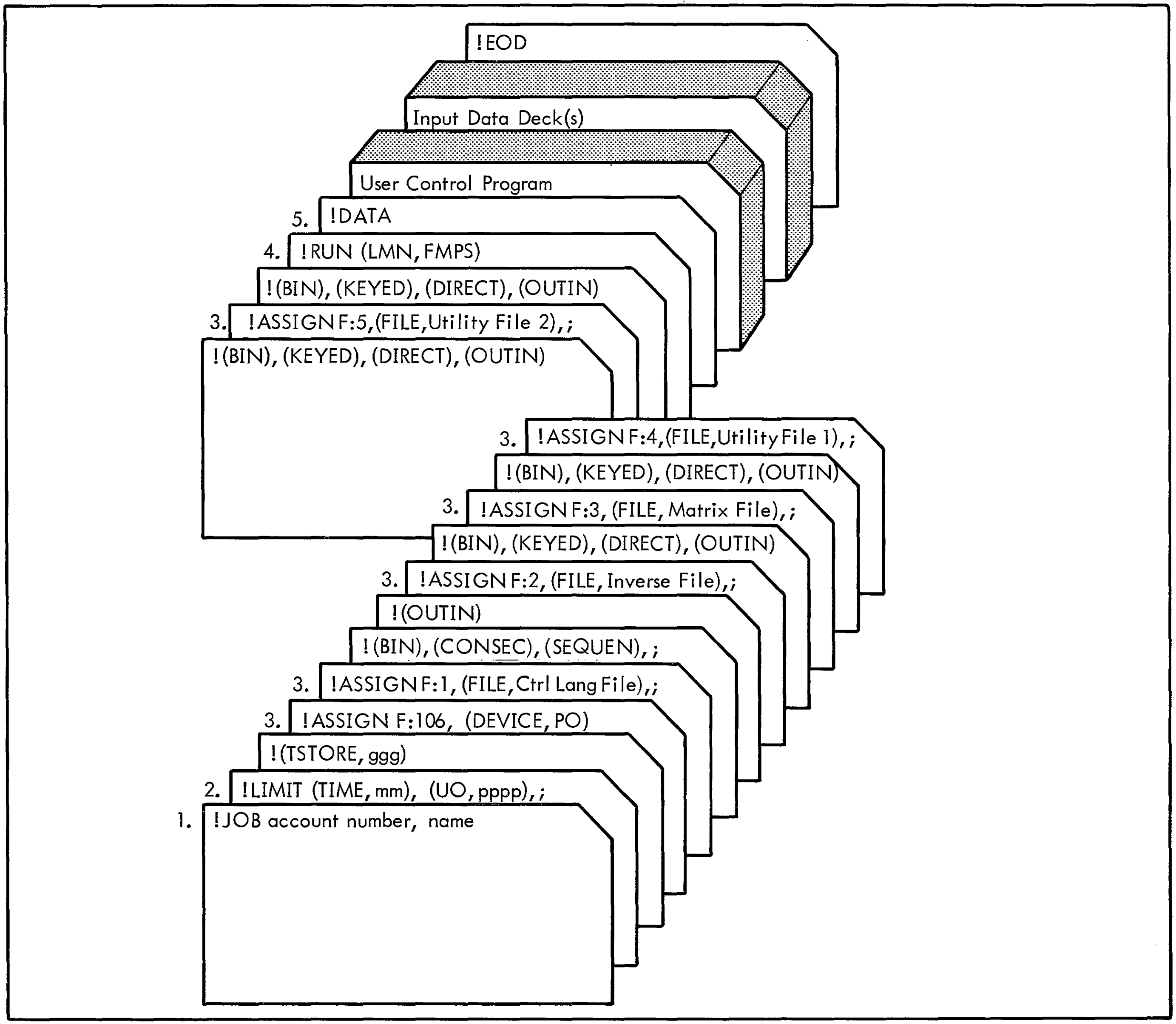
ORGANIZING THE CONTROL PROGRAM

For simplicity and in order to avoid sequence errors, it is recommended that the control program always start with the following statement order:

```
     CALL ENTER
     ASSIGN statements for KMAJER and KMINER
     CALL DEVICE
     CALL ATTACH
```

If standard tolerance settings are to be used, the user need only be concerned with the following initializations:

CR Variable	Explanation
ADATA	Initialize prior to the call for any procedure requiring input data, or producing output on



```
   !EOD
   Input Data Deck(s)
   User Control Program
5. !DATA
4. !RUN (LMN,FMPS)
   !(BIN), (KEYED), (DIRECT), (OUTIN)
3. !ASSIGN F:5,(FILE,Utility File 2),;
   !(BIN), (KEYED), (DIRECT), (OUTIN)
3. !ASSIGN F:4,(FILE,Utility File 1),;
   !(BIN), (KEYED), (DIRECT), (OUTIN)
3. !ASSIGN F:3, (FILE, Matrix File),;
   !(BIN), (KEYED), (DIRECT), (OUTIN)
3. !ASSIGN F:2, (FILE, Inverse File),;
   !(OUTIN)
   !(BIN), (CONSEC), (SEQUEN),;
3. !ASSIGN F:1, (FILE,Ctrl Lang File),;
3. !ASSIGN F:106, (DEVICE, PO)
   !(TSTORE, ggg)
2. !LIMIT (TIME, mm), (UO, pppp),;
1. !JOB account number, name
```

Figure 8. General FMPS Deck Structure

Card types and their uses are explained below.

Card Type	Parameter	Purpose
1.	!JOB	Identifies the account number and the user for the job.
2.	!LIMIT	Sets the maximum execution time, number of printer pages and number of temporary RAD granules to be in effect for the run. This card is required only when the user expects the job to exceed the default BPM limits defined during BPM system generation.
3.	!ASSIGN	Mandatory for the five standard FMPS files and also for any additional files or tapes the job will require (for example RESTART). If the CALL BASISOUT procedure is to be used, the assign card for F:106 must be included. Note that all the standard FMPS files may be assigned to either RAD or tape; however, for improved execution speed they should be assigned to RAD as keyed direct-access files. The control language file (F:1) should always be a RAD file and must have consecutive-sequential organization.
4.	!RUN	Causes BPM to load FMPS into core and commence execution.
5.	!DATA	Signals BPM that following cards are user data decks to be read by FMPS.

CR Variable	Explanation
ADATA (cont.)	tape or cards, except for SAVE, RESTART, and INPUT when SHARE is specified.
AOBJ, ARHS	Initialize these two cells early in the control program since they are used by many procedures.
FOBJWT	Initialize at -1.0 for maximization, or 1.0 for minimization.

It is always necessary to initialize the KINV interrupt cell and to program a sequence of action for that interrupt because the KINV interrupt may occur for reasons beyond the user's control (such as the occurrence of excessive numerical errors). Also, the KINV interrupt may be activated by the timing routine built into the OPTIMIZE procedure, whenever more frequent calls for INVERT would help reduce the time per iteration within the OPTIMIZE procedure.

The SAVE procedure can be used for two purposes:

1. To preserve the problem status on tape in order to be able to restart from an advanced basis if it is necessary to discontinue the run, or if hardware errors occur.

2. To create a working copy of a problem in a compact format on magnetic tape; for instance, calling the SAVE procedure after reading a large matrix from cards allows use of the RESTART tape rather than the cards at a later time.

Execution of the SAVE procedure several times during one run causes the latest status to be preserved on tape.

Whenever a call for SAVE is executed, any information written on tape by previous calls for SAVE is overlaid by the new information being written. When restarting a run by means of the RESTART procedure, care must be used in the sequence of control program statements. Any statements that modify the communication region (CR) must appear after the call for RESTART, since execution of the RESTART procedure initializes the CR to the status at the time the problem was saved. For this reason, it is recommended that the CALL RESTART statement be placed immediately after the calls for DEVICE and for ATTACH.

MULTIPLE ATTACHMENTS OF RESTART TAPE

It is sometimes desired to use different tapes for RESTART and SAVE. In this case, it is permissible to ATTACH the RESTART file several times as in the following sequence.

```
CALL DEVICE('MATRIXIN', TAPE,'F')
CALL DEVICE('MATRXOUT', TAPE, 'G')
CALL ATTACH(RESTART, 'MATRIXIN')
CALL RESTART
    :
CALL ATTACH(RESTART, 'MATRXOUT')
    :
CALL SAVE
```

In the above sequence, the problem is restarted using RESTART tape F; following the call for RESTART, tape G is attached to the RESTART file, so that any information saved during subsequent calculations is written on that tape, rather than on tape F.

APPENDIX A. PARAMETRIC PROGRAMMING

This appendix describes three post-optimal procedures, RANGE, PARAOBJ, and PARARHS, that are available as options to FMPS. An outline of each is given in Table 23 below. Note that post-optimal procedures are available only in the linear programming operating mode.

Table 23. Parametric Programming Procedures

Procedure	Purpose
RANGE	Generates and outputs an analysis of the current LP solution.
PARAOBJ	Performs parametric programming on the objective row after optimality.
PARARHS	Performs parametric programming on the RHS after primal and dual optimality.

After an optimal solution has been obtained, the procedures RANGE, PARAOBJ, and PARARHS may be used to determine the sensitivity of the optimal solution in regard to RHS and objective function values. The RHS range computes how far the activity level of a given nonbasic variable can be changed in either direction, while holding all other nonbasic variables at the current activity level, before the optimal basis for the current RHS will change. The COST range computes how far the cost coefficient of a given basic variable can be changed in either direction, while holding the cost coefficients of all other variables constant, before the optimal basis for the current cost coefficients will change. Parametric programming is an extension of RANGES, and is used to determine how the optimal basis will change when more than one coefficient moves over a special range of values. Before performing parametric procedures, a change row or column must have been defined. Depending upon which parametric procedure is requested, a matrix cost row or RHS is changed continuously until the specified maximum change has been obtained. The cost row or RHS is called a composite because it consists of the original elements plus a given amount of a change element. The function of parametric procedures is to retain optimality and feasibility as the problem continues to change.

RANGE The RANGE procedure generates and outputs an analysis of the current LP solution.

RANGE will produce two different types of reports depending upon the optional parameters. The first parameter, BASIC, generates a report of 11 columns for the variables currently basic or at intermediate levels. The second parameter, NONBASIC, creates another report of 12 columns for the variables currently nonbasic or at limit levels. Tables 24 and 25 list column numbers as well as headings in each level. If neither BASIC nor NONBASIC is specified, both outputs will be given.

The optional parameters available to RANGE are given below.

Parameter	Explanation
BASIC	Indicates that output is to include only those columns currently in the basis.
NONBASIC	Indicates that output is to include only those constraint rows whose slack variables are currently nonbasic and those columns currently nonbasic.
ROWS	Indicates that row selection or exception list parameter follows.
COLS	Indicates that column selection or exception list parameter follows
EXCEPT	Indicates that following list reference is for exception list.
LISTR	Used in connection with ROWS to specify row selection or exception list.
LISTC	Used in connection with COLS to specify column selection or exception list.

The following interrupts may occur within RANGE.

Interrupt	Causes
KMAJER	No matrix defined.
KMINER	1. Invalid parameter. 2. Illogical combination of parameters.
KINV	1. Solution is primal or dual feasible. Typical response to this interrupt would be: CALL INVERT CALL OPTIMIZE RETURN
KIOER	Irrecoverable input/output error.

Table 24. Output for Basic Variables

Column	Heading	Description of Information in Column
1	NUMBER	The internal number associated with the BASIC variable.
2	NAME	Name of the basic variable.
3	AT	A two-character code indicating the status of the BASIC variable. Code Meaning BS Basic variable ** Separator used to distinguish slack from nonslack
4	ACTIVITY	Activity of the basic variable.
5	INPUT COST	Input cost specified by the user.
6	LOWER PROCESS	The name of the variable that would change its status (enter the basis) if the cost coefficient of the basic variable in column 2 was decreased by more than the amount in column 7.
7	LOWER INCREMENT	The maximum amount of cost coefficient decrease for the basic variable in column 2 which would not change the status of any variable. If the cost coefficient is changed beyond this amount, the variable in column 6 would change its status.
8	LOWER AT	The current status (at upper limit [UL] or at lower limit [LL]) associated with the process specified in column 6.
9	UPPER PROCESS	The name of the variable that would change its status (enter the basis) if the cost coefficient of the basic variable in column 2 was increased by more than the amount in column 10.
10	UPPER INCREMENT	The maximum amount of the cost coefficient increase for the basic variable which would not change the status of any variable. If the cost coefficient was changed beyond this amount, the status of the variable in 9 would be changed.
11	UPPER AT	The current status (at upper limit [UL] or at lower limit [LL]) associated with the variable in column 9.

Table 25. Output for Nonbasic Variables

Column	Heading	Description of Information in Column
1	NUMBER	The internal number associated with the NONBASIC variable.
2	NAME	Name of the nonbasic variable.
3	AT	A two-character code indicating the status of the NONBASIC variable. Code Meaning EQ Artificial variable. UL Row at upper limit for slack variable, or column at upper limit for nonslack variable. LL Row at lower limit for slack variable, or column at lower limit for nonslack variable. ** Separator to distinguish slack variables from nonslack variables.
4	LOWER LIMIT	The lower bound on row activity for slack variables. The lower bound on column activity for nonslack variables.
5	UPPER LIMIT	The upper bound on row activity for slack variables. The upper bound on column activity for nonslack variables.
6	REDUCED COST	The DJ of the variable in column 2.

Table 25. Output for Nonbasic Variables (cont.)

Column	Heading	Description of Information in Column
7	LOWER PROCESS	The name of the basic variable that would leave the basis if the original activity level of the variable in column 2 was decreased beyond the amount in column 8.
8	LOWER INCREMENT	The maximum amount of original activity decrease of the variable in column 2 which would not change the status of any variable. If the activity level decreased beyond this amount, the basic variable in column 7 would leave the basis. (The lower limit of the variable is ignored.)
9	LOWER AT	A two-character code indicating the status at which the BASIC variable in column 7 would leave the basis. Code — Meaning UL — Variable leaves basis at upper limit. LL — Variable leaves basis at lower limit.
10	UPPER PROCESS	The name of the basic variable that would leave the basis if the original activity level of the variable in column 2 decreased beyond the amount in column 11.
11	UPPER INCREMENT	The maximum amount of original activity increase of the variable in column 2 which would not change the status of any variable. If the activity level was increased beyond this amount, the basic variable in column 10 would leave the basis. (The upper limit of the variable is ignored.)
12	UPPER AT	A two-character code indicating the status at which the BASIC variable in column 10 would leave the basis. Code — Meaning UL — Variable leaves basis at upper limit. LL — Variable leaves basis at lower limit.

PARAOBJ The PARAOBJ procedure is used to perform parametric programming on the objective row after an LP problem has reached optimally. From any LP program a series of related problems can be defined by replacing the objective row with the original row plus a multiple of a change objective row. This multiple, FTHETAC, is the parameter commonly known as THETA. In PARAOBJ, each value of FTHETAC defines a different problem with different cost coefficients. The function of this procedure is to trace the whole series of solutions, varying FTHETAC from zero up to a maximum parameter of FTHETACM defined by the user. FTHETAC is gradually increased while the solution is kept primal and dual feasible by changing the basis when necessary. Solution printout may be obtained optionally at a basis change or at a chosen interval of FTHETAC.

PARAOBJ produces an iteration lob at each basis change which is identical to that of OPTIMIZE with the exception of the THETA column which represents the current value of the parameter.

The following parameters must be defined, in addition to those parameters requested by OPTIMIZE procedure, before PARAOBJ procedure is called.

Parameter	Explanation
APOBJ	Contains name of objective function row.
FTHETAC	Initial value of THETA for PARAOBJ.
FTHETACM	Maximum value of THETA for PARAOBJ.
FTHETACP	The incremental value for THETA during PARAOBJ for which the KSOLTN interrupt will occur.

PARAOBJ will terminate at one of the following three conditions.

1. The parameter is at its maximum value of FTHETACM. The message

```
'MAXIMUM OF PARAMETER OF THETA AT
.XXXXXX'
```

is printed and FTHETAC is set to FTHETACM.

2. The problem becomes unbounded at the current value of the parameter and no further basis change will occur. The message

```
'PREMATURE MAXIMUM OF THETA AT
.XXXXXX'
```

is printed and FTHETAC retains the current value.

3. The parameter has reached a value beyond which it can be increased indefinitely without any basis change to maintain optimality. The message

```
'NO MAXIMUM FOR PARAMETER OF THETA AT
.XXXXXX'
```

is printed and FTHETAC is set to FTHETACM.

The following interrupts may occur within PARAOBJ.

Interrupt	Causes
KMAJER	1. AOBJ, ARHS or APOBJ undefined. 2. No matrix to parameterize.
KINV	1. Problem is initially primal or dual infeasible. 2. Problem has lost primal or dual feasibility due to numerical error. 3. Inversion frequency satisfied. 4. Inverse exceeding file storage. Normal interrupt response for KINV would be: CALL INVERT CALL OPTIMIZE RETURN
KSOLTN	Solution printing is requested. A typical response to this interrupt would be: CALL SOLUTION RETURN
KIOER	1. Irrecoverable input/output error. 2. File capacity exceeded.
KFREQA	User iteration frequency (IFREQA) satisfied.
KTIME	User-specified time increment reached.

PARARHS The PARARHS procedure is used to perform parametric programming on the RHS after a problem has reached primal and dual optimality. From any LP problem a series of related problems can be defined by replacing the RHS with the original RHS plus a multiple of a change RHS. This multiple, FTHETAR, is the parameter commonly known as THETA. In PARARHS each value of FTHETAR defines a different LP problem with a different RHS. The function of this procedure is to trace the whole series of solutions by varying FTHETAR from zero up to a maximum parameter of FTHETAM defined by the user. FTHETAR is gradually increased while the solution is kept primal and dual feasible by changing the basis when necessary. Solution printouts may be obtained optionally at basis changes or at a chosen interval of FTHETAR.

PARARHS produces an iteration log at each basis change which is identical to that of OPTIMIZE with the exception of the THETA column representing the current value of FTHETAR.

The following parameters must be defined before PARARHS is called.

Parameter	Explanation
APRHS	Name of the parametric RHS.
FTHETAR	Initial value of THETA for PARARHS.
FTHETARM	Maximum value of THETA for PARARHS.
FTHETARP	The incremental value for THETA during PARARHS for which the KSOLTN interrupt will occur.

PARARHS will terminate for one of the following three conditions.

1. The parameter is at its maximum value of FTHETARM. The message

```
'MAXIMUM OF PARAMETER OF THETA AT
.XXXXXX'
```

is printed and FTHETAR is set to FTHETARM.

2. The problem becomes infeasible at the current value of parameter and no further basis change can occur. The message.

```
'PREMATURE MAXIMUM OF THETA AT  .XXXXXX'
```

is printed and FTHETAR retains the current value.

3. The parameter has reached a value beyond which it can be increased indefinitely without any basis change to maintain feasibility. The message

```
'NO MAXIMUM FOR PARAMETER OF THETA AT
.XXXXXX'
```

is printed and FTHETAR is set to FTHETARM.

The following interrupts may occur within PARARHS.

Interrupt	Causes
KMAJER	1. AOBJ, ARHS or APRHS undefined. 2. No matrix to parameterize.
KINV	1. Problem initially primal or dual infeasible. 2. Problem has lost primal or dual feasibility due to numerical error.

Interrupt	Causes
	3. Inversion frequency satisfied. Normal interrupt response for KINV would be: CALL INVERT CALL OPTIMIZE RETURN
KSOLTN	Solution printing is requested. A typical response to this interrupt would be: CALL SOLUTION RETURN
KIOER	1. Irrecoverable input/output error. 2. File capacity exceeded.
KFREQA	User iteration frequency (IFREQA) satisfied.
KTIME	User-specified time increment reached.

APPENDIX B. FMPS ERROR MESSAGES

CONTROL LANGUAGE COMPILER DIAGNOSTICS

The following list specifies the error messages that can be produced by the control language compiler at compile time. Any error during compilation precludes execution of the control program. Note that all error lines are prefixed with

ERROR*****.

Computer diagnostics are listed below. Note that in the INVALID PARAMETER message, aaaaaaaa contains the name, in from one to eight characters, of the incorrect parameter.

ILLEGAL STATEMENT

STATEMENT NUMBER MUST BE NUMERIC

ASSIGN STATEMENT MUST REFER TO INTERRUPT CELL

REQUIRED FIELD MISSING

THE STATEMENT NUMBER OF A GO TO STATEMENT MUST BE NUMERIC OR KTYPE

ARGUMENT ON LEFT OF EQUAL SIGN MUST BE EITHER USER OR COMMON STORAGE VARIABLE

EQUAL SIGN MISSING

INVALID PARAMETER aaaaaaaa

MISSING LEFT PARENTHESIS

LOGICAL OPERATOR MUST BE ENCLOSED IN PERIODS

ILLEGAL LOGICAL OPERATOR

MISSING RIGHT PARENTHESIS

INVALID PROCEDURE NAME

UNDEFINED STATEMENT NUMBER

DUPLICATE STATEMENT NUMBER

NOT ENOUGH CORE AVAILABLE TO PROCESS THIS MANY STATEMENTS

MISSING TERMINAL QUOTE

INPUT/OUTPUT ERROR TYPES

The following table describes the input/output error messages that can occur during an FMPS run.

Table 26. Input/Output Error Types

Error Type	Description
1.	A file is referenced but no ATTACH was made.
2.	No DEVICE is attached to a file.
3.	Device read error.
4.	Device write error.
5.	Volume of storage for device exceeded during a write operation.
6.	Attempt to write on a file in read or closed status.
7.	Attempt to read on file in write or closed status.
8.	Attempt to read beyond written information.
9.	Dynamic core pointer for a file buffer points to an illegal core area.
10.	Undefined type of device, i.e., device not DISC or TAPE.
11.	Insufficient core available to create even one file buffer.

APPENDIX C. FMPS SAMPLE RUNS

```
JØB 326,SDMD
LIMIT (TIME,90),(LØ,1000),(UØ,1000),(DØ,1000)
ASSIGN F:106,(DEVICE,CPA04)
ASSIGN F:1,(FILE,CLANG),(BIN),(WRITE,ALL),(CØNSEC),(SEQUEN),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:2,(FILE,UTIL1),(BIN),(WRITE,ALL),(KEYED),(DIRECT),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:3,(FILE,UTIL2),(BIN),(WRITE,ALL),(KEYED),(DIRECT),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:4,(FILE,MTRX),(BIN),(WRITE,ALL),(DIRECT),(KEYED),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:5,(FILE,IVSE),(BIN),(WRITE,ALL),(DIRECT),(KEYED),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:6,(DEVICE,9T),(INØUT),(INSN,026),(BIN),(WRITE,ALL),(SAVE)
RUN (LMN,FMPS)
DATA
```

```
C      THIS IS A CØMMENT (PUNCHED C IN CØL 1)
C
C         DEFINE HEADING AND ENTER L.P. MØDE
C
       TITLE SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN
C
C   THIS BENCHMARK HAS BEEN PURPØSELY MADE QUITE CØMPLEX TØ DEMØNSTRATE
C    MANY ØF THE ØPTIØNS AVAILABLE IN FMPS. USUALLY, CØNTRØL PRØGRAMS
C       ARE MUCH SIMPLER AND THE STANDARD ØPTIØNS ARE USED.
C
       CALL ENTER(LP)
C
C
C       INITIALIZE MAJØR ERRØR INTERRUPT VARIABLE
       ASSIGN 300 TØ KMAJER
C       INITIALIZE MINØR ERRØR INTERRUPT VARIABLE
       ASSIGN 300 TØ KMINER
C       SET TIME LIMIT ØF 5 MINUTES FRØM EXECUTIØN ØF THIS STATEMENT
       ITIME = 5
C       INITIALIZE TIME-ØUT INTERRUPT VARIABLE
       ASSIGN 45 TØ KTIME
C       SPECIFY FØUR SYMBØLIC UNITS (WØRKING FILES) ØN RAD
C
       CALL DEVICE('FILE1',DISC,'B')
       CALL DEVICE('FILE2',DISC,'C')
       CALL DEVICE('FILE3',DISC,'D')
       CALL DEVICE('FILE4',DISC,'E')
C
C      SPECIFY A SYMBØLIC UNIT ØN TAPE (LØGICAL NUMBER A)
        CALL DEVICE('TAPEA',TAPE,'F')
C
C      ATTACH THE FØUR STANDARD L.P. FMPS FILES TØ THE
C      PREVIØUSLY DEFINED FØUR SYMBØLIC UNITS (RAD)
C
        CALL ATTACH(MATRIX,'FILE1')
       CALL ATTACH(INVERSE,'FILE2')
       CALL ATTACH(UTIL1,'FILE3')
       CALL ATTACH(UTIL2,'FILE4')
C
C      ATTACH THE RESTART FILE TØ LØGICAL TAPE A PREVIØUSLY DEFINED
C
       CALL ATTACH(RESTART,'TAPEA')
C
C      NØTE FØR THE ABØVE-MATRIX,INVERSE,UTIL1,UTIL2, AND RESTART
C      ARE INTERNAL FILES WHICH MUST ALWAYS BE ATTACHED
C      EXCEPT RESTART IF NØ SAVING ØR RESTARTING IS PRØGRAMMED
C
C      SELECT DESIRED INPUT DATA RECØRD AND SPECIFY PRØBLEM NAME
C
       ADATA = 'ALLØYS'
       APØNAME = 'FUSIØN'
C
C      LØAD INPUT MATRIX FRØM CARDS, USING RECØRD 'ALLØYS'
```

```
C
       CALL INPUT
C
C      CALL INPUT(FILE,FILENAME) WØULD RESULT IN SEARCHING INPUT FILE
C      CALLED FILENAME FØR RECØRD ALLØYS AND LØADING IT AS INPUT MATRIX
C      IN THIS CASE ØNE SHØULD FIRST DEFINE THE FILE AND ATTACH IT
C      BY MEANS ØF DEVICE AND ATTACH CALLS.
C
C      IDENTIFY RIGHT-HAND-SIDE CØLUMN AND CØST RØW TØ BE USED
C
       ARHS = 'ALØY1'
       AØBJ = 'VALUE'
C
C
C
C
C         V A R I Ø U S  Ø P T I Ø N S  T Ø  D I S P L A Y  M A T R I X
C
C
C        DISPLAY ØRIGINAL MATRIX IN STANDARD FØRMAT
```

```
C
      CALL OUTPUT
C
C     DISPLAY ORIGINAL MATRIX IN CODED FORM
C
      CALL OUTPUT(CODED)
C
C     DISPLAY ORIGINAL MATRIX IN ROW ORDER
C
      CALL OUTPUT(BYROWS)
C
C     DISPLAY ORIGINAL MATRIX IN COLUMN ORDER
C
      CALL OUTPUT(BYCOLS)
C
C
C
C      E X A M P L E   O F   S O L U T I O N
C
C
C
C     VARIOUS INITIALIZATIONS FOR SOLUTION (OPTIMIZE)
C
C     SET TO INVERT NO LESS FREQUENTLY THAN AT INTERVALS OF 4 ITERATIONS
C
      IFREQI = 4
C
C     ASSIGN WEIGHT OF 1.0 TO OBJECTIVE ROW
C     (1.0 RESULTS IN MINIMIZATION, -1.C IN MAXIMIZATION)
C
      FOBJWT = 1.0
C
C     SET TO PRINT ITERATION LOG EACH ITERATION (PRINTER OUTPUT)
```

```
C
      ILOGP = 1
C     SET PRICING TO BE MADE FROM GROUPS OF TWO PROFITABLE VARIABLES
C
      INCAND = 2
C
C     SET INVERSION INTERRUPT CELL TO TRANSFER TO STATEMENT 200
C
      ASSIGN 200 TO KINV
C
C     NOW SET MINOR ERROR INTERRUPT TO CAUSE CREATION OF RESTART TAPE
C     IF IT WERE TO OCCUR DURING THE OPTIMIZE PHASE
      ASSIGN 400 TO KMINER
C
C     SET OPTIMIZE TO DISREGARD OPTIMALITY DURING PHASE ONE
      FCMPDJ = 0.0
C
C     SOLVE L.P. MATRIX
C
      CALL OPTIMIZE
C
C     PRESERVE BASIS OF OPTIMAL SOLUTION
      CALL SAVE
C     PRINT SOLUTION VALUES (COLUMNS AND ROWS)
      CALL SOLUTION
C
C     PRINT PRIMAL AND DUAL ERRORS
C
      CALL ERRORS
C
C
C     E X A M P L E   O F   R A N G E   C A L C U L A T I O N S
C
C
      CALL RANGE
C
C
C     E X A M P L E   O F   C O S T   P A R A M E T R I C S
C
C
C
C     SET INITIAL AND MAXIMUM THETA VALUES FOR COST PARAMETRICS
      FTHETAC = 0.0
      FTHETACM = 10.
C
C     SET TO PRINT SOLUTIONS AT THETA INTERVALS OF .05
      FTHETACP = .05
C     IDENTIFY COST PARAMETRIC ROW (THE ONE TO BE MULTIPLIED BY THETA)
      APOBJ = 'DELCST'
C     INITIALIZE SOLUTION REQUEST INTERRUPT VARIABLE
      ASSIGN 600 TO KSOLTN
      ASSIGN 700 TO KINV
C     EXECUTE PARAMETRIC COST RUN
      CALL PARAOBJ
```

```
      CALL SOLUTION
C
C
C     E X A M P L E   O F   R H S   P A R A M E T R I C   R U N
C
C
C     RESTORE OPTIMAL BASIS
      CALL RESTORE
C     SET INITIAL AND MAXIMUM THETA VALUES FOR RHS PARAMETRICS
      FTHETAR = 0.0
      FTHETARM = 10.0
```

```
C     SET TØ PRINT SØLUTIØN AT THETA INTERVALS ØF 1.0
      FTHETARP = 1.0
C     IDENTIFY RHS PARAMETRIC ØLUMN (THE ØNE TØ BE MULTIPLIED BY THETA)
      APRHS = 'DELPRØDC'
C     EXECUTE PARAMETRIC RHS RUN
      CALL PARARHS
      CALL SØLUTIØN
C
C
      STØP
C
C
C     THE FØLLØWING STATEMENTS CØNTRØL THE RESPØNSE TØ INTERRUPTS
C
C
C     ENTER HERE FØR TIME-ØUT INTERRUPT
C     PRESERVE PRØBLEM STATUS ØN RESTART TAPE
   45 CALL SAVE
C     TERMINATE RUN
      STØP
C     ENTER HERE WHEN INVERSIØN INTERRUPT ØCCURS
  200 CALL INVERT
C     RETURN TØ PRØCEDURE THAT CAUSED THE KINV INTERRUPT
      RETURN
C     ENTER HERE IN CASE ØF MAJØR ØR MINØR ERRØRS
C     DISPLAY CØMMUNICATIØN REGIØN VARIABLES AND FILE STATUS
  300 CALL CØNDITIØN
C     TERMINATE FMPS EXECUTIØN
      STØP
C     ENTER HERE FØR MINØR ERRØR INTERRUPT DURING ØPTIMIZE PHASE
C     DISPLAY FMPS STATUS
  400 CALL CØNDITIØN
C     DØ SAME AS IF TIMEØUT ØCCURED
      GØ TØ 45
C     ENTER HERE WHEN SØLUTIØN PRINT-ØUT IS REQUESTED (BASIS CHANGE
C     ØR SØLUTIØN PRINT-ØUT INTERVAL ØF THETA SATISFIED)
C     PRINT SØLUTIØN
  600 CALL SØLUTIØN
C     PRINT VALUE ØF ITERATIØN CØUNT
      WRITE ITCNT
C     RETURN TØ PARAMETRICS
      RETURN
```

```
C     ENTER HERE IF NUMERICAL ACCURACY CAUSES INFEASIBILITY DURING PARAMETRICS
  700 CALL INVERT
      CALL ØPTIMIZE
      RETURN
C     END ØF CØNTRØL PRØGRAM
      END
```

```
NAME           ALLØYS
RØWS
 N  VALUE
 E  YIELD
 L  FE
 L  MN
 L  CU
 L  MG
 G  AL
 L  SI
  N DELCST
CØLUMNS
    BIN1       VALUE       0.03000
    BIN1       YIELD       1.00000
    BIN1       FE          0.15000
    BIN1       CU          0.03000
    BIN1       MN          0.02000
    BIN1       MG          0.02000
    BIN1       AL          0.70000
    BIN1       SI          0.02000
    BIN1       DELCST        -10.0
    BIN2       VALUE       0.08000
    BIN2       YIELD       1.00000
    BIN2       FE          0.04000
    BIN2       CU          0.05000
    BIN2       MN          0.04000
    BIN2       MG          0.03000
    BIN2       AL          0.75000
    BIN2       SI          0.06000
    BIN3       VALUE       0.17000
    BIN3       YIELD       1.00000
    BIN3       FE          0.02000
    BIN3       CU          0.08000
    BIN3       MN          0.01000
    BIN3       AL          0.80000
    BIN3       SI          0.08000
    BIN4       VALUE       0.12000
    BIN4       YIELD       1.00000
    BIN4       FE          0.04000
    BIN4       CU          0.02000
    BIN4       MN          0.02000
    BIN4       AL          0.75000
    BIN4       SI          0.12000
    BIN5       VALUE       0.15000
    BIN5       YIELD       1.00000
    BIN5       FE          0.02000
    BIN5       CU          0.06000
    BIN5       MN          0.02000
    BIN5       MG          0.01000
```

```
BIN5       AL        0.80000
BIN5       SI        0.02000
ALUM       VALUE     0.21000
ALUM       YIELD     1.00000
```

```
    ALUM       FE         0.01000
    ALUM       CU         0.01000
    ALUM       AL         0.97000
    ALUM       SI         0.01000
    SILCØN     VALUE      0.38000
    SILCØN     YIELD      1.00000
    SILCØN     FE         0.03000
    SILCØN     SI         0.97000
RHS
    ALØY1      YIELD      2000.00000
    ALØY1      FE           60.00000
    ALØY1      CU          100.00000
    ALØY1      MN           40.00000
    ALØY1      MG           30.00000
    ALØY1      AL         1500.00000
    ALØY1      SI          300.00000
    DELPRØDC   YIELD      20000.0
RANGES
    AL1        SI           50.00000
BØUNDS
 UP PRØD1      BIN1        200.00000
 UP PRØD1      BIN2       2500.00000
 LØ PRØD1      BIN3        400.00000
 UP PRØD1      BIN3        800.00000
 LØ PRØD1      BIN4        100.00000
 UP PRØD1      BIN4        700.00000
 UP PRØD1      BIN5       1500.00000
ENDATA
```

```
11:34 FEB 12,'69 ID=0000
JØB 326,SDMD
LIMIT (TIME,90),(LØ,1000),(UØ,1000),(DØ,1000)
ASSIGN F:106,(DEVICE,CPA04)
ASSIGN F:1,(FILE,CLANG),(BIN),(WRITE,ALL),(CØNSEC),(SEQUEN),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:2,(FILE,UTIL1),(BIN),(WRITE,ALL),(KEYED),(DIRECT),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:3,(FILE,UTIL2),(BIN),(WRITE,ALL),(KEYED),(DIRECT),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:4,(FILE,MTRX),(BIN),(WRITE,ALL),(DIRECT),(KEYED),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:5,(FILE,IVSE),(BIN),(WRITE,ALL),(DIRECT),(KEYED),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:6,(DEVICE,9T),(INØUT),(INSN,026),(BIN),(WRITE,ALL),(SAVE)
RUN (LMN,FMPS)
```

```
12FEB69                                                                                   0.   0.   1.


INTERNAL STATEMENT NUMBER    0     TIME = 11:34
        C      THIS IS A CØMMENT (PUNCHED C IN CØL 1)
        C
        C         DEFINE HEADING AND ENTER L.P. MØDE
        C
   1 **        TITLE SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN
        C
        C   THIS BENCHMARK HAS BEEN PURPØSELY MADE CUITE CØMPLEX TØ DEMØNSTRATE
        C    MANY ØF THE ØPTIØNS AVAILABLE IN FMPS. USUALLY, CØNTRØL PRØGRAMS
        C       ARE MUCH SIMPLER AND THE STANDARD ØPTIØNS ARE USED.
        C
   2 **        CALL ENTER(LP)
        C
        C
        C       INITIALIZE MAJØR ERRØR INTERRUPT VARIABLE
   3 **        ASSIGN 300 TØ KMAJER
        C       INITIALIZE MINØR ERRØR INTERRUPT VARIABLE
   4 **        ASSIGN 300 TØ KMINER
        C       SET TIME LIMIT ØF 5 MINUTES FRØM EXECUTIØN ØF THIS STATEMENT
   5 **        ITIME = 5
        C       INITIALIZE TIME-ØUT INTERRUPT VARIABLE
   6 **        ASSIGN 45 TØ KTIME
        C       SPECIFY FØUR SYMBØLIC UNITS (WØRKING FILES) ØN RAD
        C
   7 **        CALL DEVICE('FILE1',DISC,'B')
   8 **        CALL DEVICE('FILE2',DISC,'C')
   9 **        CALL DEVICE('FILE3',DISC,'D')
  10 **        CALL DEVICE('FILE4',DISC,'E')
        C
        C      SPECIFY A SYMBØLIC UNIT ØN TAPE (LØGICAL NUMBER A)
  11 **         CALL DEVICE('TAPEA',TAPE,'F')
        C
        C      ATTACH THE FØUR STANDARD L.P. FMPS FILES TØ THE
        C      PREVIØUSLY DEFINED FØUR SYMBØLIC UNITS (RAD)
        C
  12 **         CALL ATTACH(MATRIX,'FILE1')
  13 **        CALL ATTACH(INVERSE,'FILE2')
  14 **        CALL ATTACH(UTIL1,'FILE3')
  15 **        CALL ATTACH(UTIL2,'FILE4')
        C
        C      ATTACH THE RESTART FILE TØ LØGICAL TAPE A PREVIØUSLY DEFINED
        C
  16 **        CALL ATTACH(RESTART,'TAPEA')
```

```
      C
      C     NOTE FOR THE ABOVE-MATRIX,INVERSE,UTIL1,UTIL2, AND RESTART
      C     ARE INTERNAL FILES WHICH MUST ALWAYS BE ATTACHED
      C     EXCEPT RESTART IF NO SAVING OR RESTARTING IS PROGRAMMED
      C
      C     SELECT DESIRED INPUT DATA RECORD AND SPECIFY PROBLEM NAME
      C
```

```
12FEB69                                                                  0.   0.   2.

   17 **    ADATA = 'ALLOYS'
   18 **    APBNAME = 'FUSION'
      C
      C     LOAD INPUT MATRIX FROM CARDS, USING RECORD 'ALLOYS'
      C
   19 **    CALL INPUT
      C
      C     CALL INPUT(FILE,FILENAME) WOULD RESULT IN SEARCHING INPUT FILE
      C     CALLED FILENAME FOR RECORD ALLOYS AND LOADING IT AS INPUT MATRIX
      C     IN THIS CASE ONE SHOULD FIRST DEFINE THE FILE AND ATTACH IT
      C     BY MEANS OF DEVICE AND ATTACH CALLS.
      C
      C     IDENTIFY RIGHT-HAND-SIDE COLUMN AND COST ROW TO BE USED
      C
   20 **    ARHS = 'ALOY1'
   21 **    AOBJ = 'VALUE'
      C
      C
      C
      C
      C       V A R I O U S   O P T I O N S   T O   D I S P L A Y   M A T R I X
      C
      C
      C      DISPLAY ORIGINAL MATRIX IN STANDARD FORMAT
      C
   22 **    CALL OUTPUT
      C
      C     DISPLAY ORIGINAL MATRIX IN CODED FORM
      C
   23 **    CALL OUTPUT(CODED)
      C
      C     DISPLAY ORIGINAL MATRIX IN ROW ORDER
      C
   24 **    CALL OUTPUT(BYROWS)
      C
      C     DISPLAY ORIGINAL MATRIX IN COLUMN ORDER
      C
   25 **    CALL OUTPUT(BYCOLS)
      C
      C
      C
      C      E X A M P L E   O F   S O L U T I O N
      C
      C
      C
      C     VARIOUS INITIALIZATIONS FOR SOLUTION (OPTIMIZE)
      C
      C     SET TO INVERT NO LESS FREQUENTLY THAN AT INTERVALS OF 4 ITERATIONS
      C
   26 **    IFREQI = 4
```

```
12FEB69                                                                  0.   0.   3.

      C
      C     ASSIGN WEIGHT OF 1.0 TO OBJECTIVE ROW
      C     (1.0 RESULTS IN MINIMIZATION, -1.0 IN MAXIMIZATION)
      C
   27 **    FOBJWT = 1.0
      C
      C     SET TO PRINT ITERATION LOG EACH ITERATION (PRINTER OUTPUT)
      C
   28 **    ILOGP = 1
      C     SET PRICING TO BE MADE FROM GROUPS OF TWO PROFITABLE VARIABLES
      C
   29 **    INCAND = 2
      C
      C     SET INVERSION INTERRUPT CELL TO TRANSFER TO STATEMENT 200
      C
   30 **    ASSIGN 200 TO KINV
      C
      C     NOW SET MINOR ERROR INTERRUPT TO CAUSE CREATION OF RESTART TAPE
      C     IF IT WERE TO OCCUR DURING THE OPTIMIZE PHASE
   31 **    ASSIGN 400 TO KMINER
      C
      C     SET OPTIMIZE TO DISREGARD OPTIMALITY DURING PHASE ONE
   32 **    FCHPDJ = 0.0
      C
      C     SOLVE L.P. MATRIX
      C
   33 **    CALL OPTIMIZE
      C
      C     PRESERVE BASIS OF OPTIMAL SOLUTION
   34 **    CALL SAVE
      C     PRINT SOLUTION VALUES (COLUMNS AND ROWS)
   35 **    CALL SOLUTION
      C
      C     PRINT PRIMAL AND DUAL ERRORS
```

```
         C
  36 **          CALL ERRORS
         C
         C
         C       E X A M P L E   O F   R A N G E   C A L C U L A T I O N S
         C
         C
  37 **          CALL RANGE
         C
         C
         C       E X A M P L E   O F   C O S T   P A R A M E T R I C S
         C
         C
         C       SET INITIAL AND MAXIMUM THETA VALUES FOR COST PARAMETRICS
  38 **          FTHETAC = 0.0
  39 **          FTHETACM = 10.
```

```
12FEB69                                                                          0.    0.    4.

         C
         C       SET TO PRINT SOLUTIONS AT THETA INTERVALS OF .05
  40 **          FTHETACP = .05
         C       IDENTIFY COST PARAMETRIC ROW (THE ONE TO BE MULTIPLIED BY THETA)
  41 **          APOBJ = 'DELCST'
         C       INITIALIZE SOLUTION REQUEST INTERRUPT VARIABLE
  42 **          ASSIGN 600 TO KSOLTN
  43 **          ASSIGN 700 TO KINV
         C       EXECUTE PARAMETRIC COST RUN
  44 **          CALL PARAOBJ
  45 **          CALL SOLUTION
         C
         C
         C       E X A M P L E   O F   R H S   P A R A M E T R I C   R U N
         C
         C
         C       RESTORE OPTIMAL BASIS
  46 **          CALL RESTORE
         C       SET INITIAL AND MAXIMUM THETA VALUES FOR RHS PARAMETRICS
  47 **          FTHETAR = 0.0
  48 **          FTHETARM = 10.0
         C       SET TO PRINT SOLUTION AT THETA INTERVALS OF 1.0
  49 **          FTHETARP = 1.0
         C       IDENTIFY RHS PARAMETRIC OLUMN (THE ONE TO BE MULTIPLIED BY THETA)
  50 **          APRHS = 'DELPRODC'
         C       EXECUTE PARAMETRIC RHS RUN
  51 **          CALL PARARHS
  52 **          CALL SOLUTION
         C
         C
  53 **          STOP
         C
         C
         C       THE FOLLOWING STATEMENTS CONTROL THE RESPONSE TO INTERRUPTS
         C
         C
         C       ENTER HERE FOR TIME-OUT INTERRUPT
         C       PRESERVE PROBLEM STATUS ON RESTART TAPE
  54 **       45 CALL SAVE
         C       TERMINATE RUN
  55 **          STOP
         C       ENTER HERE WHEN INVERSION INTERRUPT OCCURS
  56 **      200 CALL INVERT
         C       RETURN TO PROCEDURE THAT CAUSED THE KINV INTERRUPT
  57 **          RETURN
         C       ENTER HERE IN CASE OF MAJOR OR MINOR ERRORS
         C       DISPLAY COMMUNICATION REGION VARIABLES AND FILE STATUS
  58 **      300 CALL CONDITION
         C       TERMINATE FMPS EXECUTION
  59 **          STOP
```

```
12FEB69                                                                          0.    0.    5.

         C       ENTER HERE FOR MINOR ERROR INTERRUPT DURING OPTIMIZE PHASE
         C       DISPLAY FMPS STATUS
  60 **      400 CALL CONDITION
         C       DO SAME AS IF TIMEOUT OCCURED
  61 **          GO TO 45
         C       ENTER HERE WHEN SOLUTION PRINT-OUT IS REQUESTED (BASIS CHANGE
         C       OR SOLUTION PRINT-OUT INTERVAL OF THETA SATISFIED)
         C       PRINT SOLUTION
  62 **      600 CALL SOLUTION
         C       PRINT VALUE OF ITERATION COUNT
  63 **          WRITE ITCNT
         C       RETURN TO PARAMETRICS
  64 **          RETURN
         C       ENTER HERE IF NUMERICAL ACCURACY CAUSES INFEASIBILITY DURING PARAMETRICS
  65 **      700 CALL INVERT
  66 **          CALL OPTIMIZE
  67 **          RETURN
         C       END OF CONTROL PROGRAM
  68 **          END
INTERNAL STATEMENT NUMBER   0     TIME = 11:35
```

```
12FEB69

INTERNAL STATEMENT NUMBER  1    TIME = 11:35
INTERNAL STATEMENT NUMBER  2    TIME = 11:35
INTERNAL STATEMENT NUMBER  3    TIME = 11:35
INTERNAL STATEMENT NUMBER  4    TIME = 11:35
INTERNAL STATEMENT NUMBER  5    TIME = 11:35
INTERNAL STATEMENT NUMBER  6    TIME = 11:35
INTERNAL STATEMENT NUMBER  7    TIME = 11:35
INTERNAL STATEMENT NUMBER  8    TIME = 11:35
INTERNAL STATEMENT NUMBER  9    TIME = 11:35
INTERNAL STATEMENT NUMBER 10    TIME = 11:35
INTERNAL STATEMENT NUMBER 11    TIME = 11:35
INTERNAL STATEMENT NUMBER 12    TIME = 11:35
INTERNAL STATEMENT NUMBER 13    TIME = 11:35
INTERNAL STATEMENT NUMBER 14    TIME = 11:35
INTERNAL STATEMENT NUMBER 15    TIME = 11:35
INTERNAL STATEMENT NUMBER 16    TIME = 11:35
INTERNAL STATEMENT NUMBER 17    TIME = 11:35
INTERNAL STATEMENT NUMBER 18    TIME = 11:35
INTERNAL STATEMENT NUMBER 19    TIME = 11:35

BUFFER SIZES (BYTES) ARE.. MATRIX =   648 INVERSE = 10240

MATRIX STATISTICS
RØWS.........      9
CØLUMNS......      8
RHS..........      2
DENSITY......  70.37
ELEMENTS.....     57
LARGEST......  0.2000000+05
SMALLEST.....  0.1000000-01
MAJØR ERRØRS       0
MINØR ERRØRS       0
INTERNAL STATEMENT NUMBER  20    TIME = 11:35
INTERNAL STATEMENT NUMBER  21    TIME = 11:35
INTERNAL STATEMENT NUMBER  22    TIME = 11:35
```

```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                              0.   2.   1.

                                                    ØRIGINAL MATRIX

 LØWER BØUND        0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000
 UPPER BØUND                                                                                                  50.00000

                   VALUE        YIELD        FE           MN           CU           MG           AL           SI
 VALUE       N      1.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000
 YIELD       E      0.00000      1.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000
 FE          L      0.00000      0.00000      1.00000      0.00000      0.00000      0.00000      0.00000      0.00000
 MN          L      0.00000      0.00000      0.00000      1.00000      0.00000      0.00000      0.00000      0.00000
 CU          L      0.00000      0.00000      0.00000      0.00000      1.00000      0.00000      0.00000      0.00000
 MG          L      0.00000      0.00000      0.00000      0.00000      0.00000      1.00000      0.00000      0.00000
* AL         G      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000     -1.00000      0.00000
 SI          L      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      1.00000
 DELCST      N      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000
```

```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                              0.   3.   1.

                                                    ØRIGINAL MATRIX

 LØWER BØUND        0.00000      0.00000      0.00000    400.00000    100.00000      0.00000      0.00000      0.00000
 UPPER BØUND                   200.00000   2500.00000    800.00000    700.00000   1500.00000

                   DELCST       BIN1         BIN2         BIN3         BIN4         BIN5         ALUM         SILCØN
 VALUE       N      0.00000      0.03000      0.08000      0.17000      0.12000      0.15000      0.21000      0.38000
 YIELD       E      0.00000      1.00000      1.00000      1.00000      1.00000      1.00000      1.00000      1.00000
 FE          L      0.00000      0.15000      0.04000      0.02000      0.04000      0.02000      0.01000      0.03000
 MN          L      0.00000      0.02000      0.04000      0.01000      0.02000      0.02000      0.00000      0.00000
 CU          L      0.00000      0.03000      0.05000      0.08000      0.02000      0.06000      0.01000      0.00000
 MG          L      0.00000      0.02000      0.03000      0.00000      0.00000      0.01000      0.00000      0.00000
* AL         G      0.00000      0.70000      0.75000      0.80000      0.75000      0.80000      0.97000      0.00000
 SI          L      0.00000      0.02000      0.06000      0.08000      0.12000      0.02000      0.01000      0.97000
 DELCST      N      1.00000    -10.00000      0.00000      0.00000      0.00000      0.00000      0.00000      0.00000
```

```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                              0.   4.   1.

                                                    ØRIGINAL MATRIX

 LØWER BØUND        0.00000      0.00000
 UPPER BØUND

                   ALØY1        DELPRØDC
 VALUE       N      0.00000      0.00000
 YIELD       E   2000.00000  20000.00000
 FE          L     60.00000      0.00000
 MN          L     40.00000      0.00000
 CU          L    100.00000      0.00000
 MG          L     30.00000      0.00000
* AL         G   1500.00000      0.00000
 SI          L    300.00000      0.00000
 DELCST      N      0.00000      0.00000
```

```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                          0.   5.   1.

INTERNAL STATEMENT NUMBER  23    TIME = 11:35
```

```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                          0.   6.   1.


                                         ORIGINAL PICTURE

  LOWER BOUND             . . . . . . . . . . . C B . . . . .
  UPPER BOUND                           B   C D C C D

                          V Y F M C M A S D B B B B B A S A D
                          A I E N U G L I E I I I I I L I L E
                          L E             L N N N N N U L O L
                          U L             C 1 2 3 4 5 M C Y P
                          E D             S             O 1 R
                                          T             N   O
                                                            D
                                                            C

  VALUE          N        1 . . . . . . . . U U T T T T T . .
  YIELD          E        . 1 . . . . . . . 1 1 1 1 1 1 1 D E
  FE             L        . . 1 . . . . . . T U U U U U U B .
  MN             L        . . . 1 . . . . . U U U U U . . B .
  CU             L        . . . . 1 . . . . U U U U U U . B .
  MG             L        . . . . . 1 . . . U U . . U . . B .
  AL             G        . . . . . .=1 . . T T T T T T . D .
  SI             L        . . . . . . . 1 . U U U T U U T C .
  DELCST         N        . . . . . . . . 1.A . . . . . . . .
```

```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                          0.   7.   1.


SYMBL        SUMMARY OF MATRIX

                  RANGE                         COUNT (INCL.RHS)

  Z            LESS THAN              .000001      0

  Y        .000001  THRU              .000009      0

  X        .000010                    .000099      0

  W        .000100                    .000999      0

  V        .001000                    .009999      0

  U        .010000                    .099999     27

  T        .100000                    .999999     14

  1       1.000000                   1.000000     16

  A       1.000001                  10.000000      1

  B      10.000001                 100.000000      4

  C     100.000001               1,000.000000      1

  D   1,000.000001              10,000.000000      2

  E  10,000.000001             100,000.000000      1

  F 100,000.000001           1,000,000.000000      0

  G         GREATER THAN     1,000,000.000000      0
```

```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                          0.   8.   1.

INTERNAL STATEMENT NUMBER  24    TIME = 11:35
```

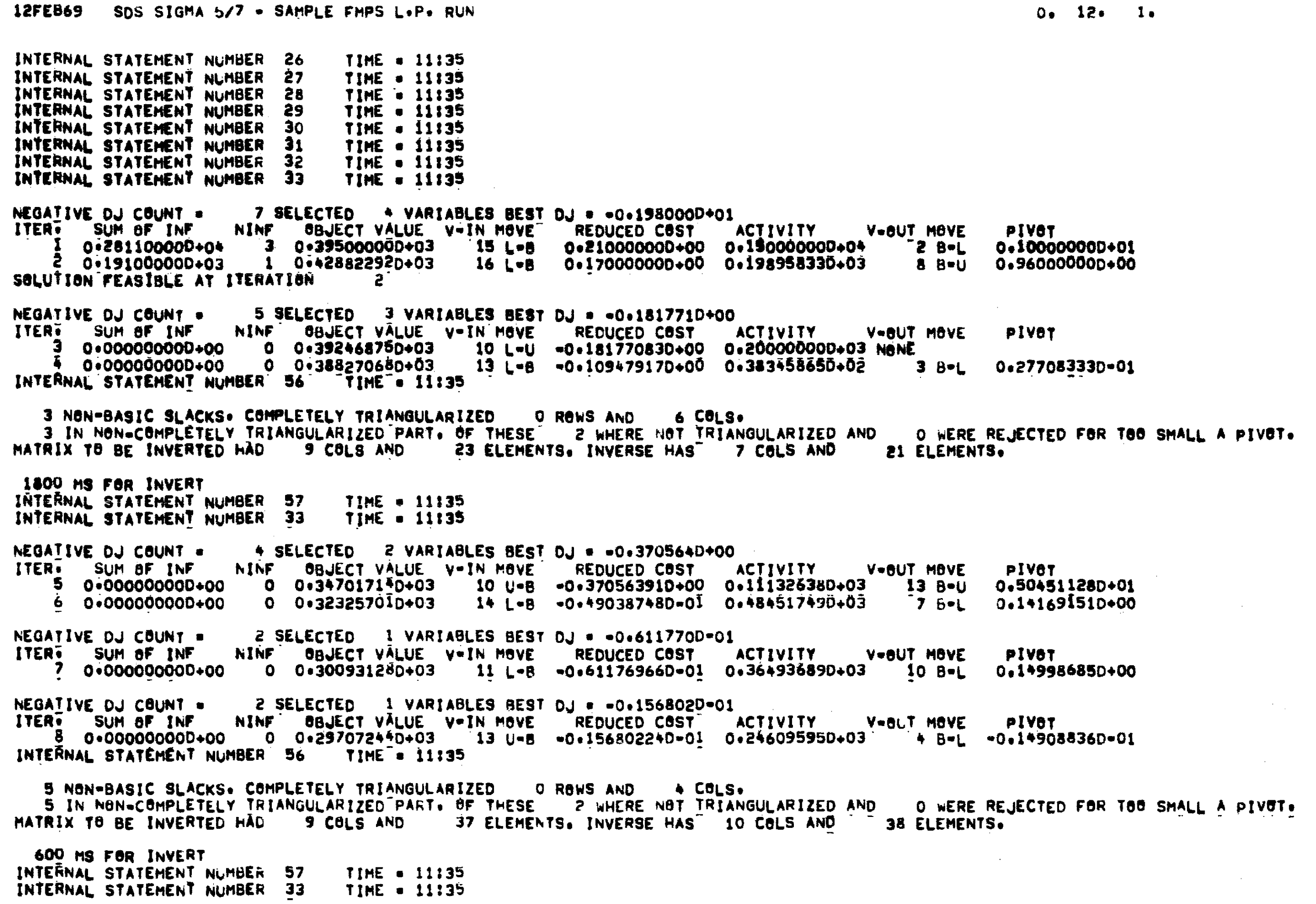
```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                          0.   9.   1.


ROW VALUE      NUMBER     1
    1.0000(VALUE    )      0.0300(BIN1     )      0.0800(BIN2     )      0.1700(BIN3     )      0.1200(BIN4     )      0.1500(BIN5     )
    0.2100(ALUM     )      0.3800(SILCON   )

ROW YIELD      NUMBER     2
    1.0000(YIELD    )      1.0000(BIN1     )      1.0000(BIN2     )      1.0000(BIN3     )      1.0000(BIN4     )      1.0000(BIN5     )
    1.0000(ALUM     )      1.0000(SILCON   )   2000.0000(ALOY1    )  20000.0000(DELPRODC)

ROW FE         NUMBER     3
    1.0000(FE       )      0.1500(BIN1     )      0.0400(BIN2     )      0.0200(BIN3     )      0.0400(BIN4     )      0.0200(BIN5     )
    0.0100(ALUM     )      0.0300(SILCON   )     60.0000(ALOY1    )

ROW MN         NUMBER     4
    1.0000(MN       )      0.0200(BIN1     )      0.0400(BIN2     )      0.0100(BIN3     )      0.0200(BIN4     )      0.0200(BIN5     )
   40.0000(ALOY1    )
```

```
ROW CU         NUMBER     5
    1.0000(CU        )    0.0300(BIN1      )     0.0500(BIN2      )     0.0800(BIN3      )     0.0200(BIN4      )     0.0600(BIN5      )
    0.0100(ALUM      )  100.0000(ALOY1     )

ROW MG         NUMBER     6
    1.0000(MG        )    0.0200(BIN1      )     0.0300(BIN2      )     0.0100(BIN5      )    30.0000(ALOY1     )

ROW AL         NUMBER     7
   -1.0000(AL        )    0.7000(BIN1      )     0.7500(BIN2      )     0.8000(BIN3      )     0.7500(BIN4      )     0.8000(BIN5      )
    0.9700(ALUM      ) 1500.0000(ALOY1     )

ROW SI         NUMBER     8
    1.0000(SI        )    0.0200(BIN1      )     0.0600(BIN2      )     0.0800(BIN3      )     0.1200(BIN4      )     0.0200(BIN5      )
    0.0100(ALUM      )    0.9700(SILCON    )   300.0000(ALOY1     )

ROW DELCST     NUMBER     9
    1.0000(DELCST    )  -10.0000(BIN1      )
```

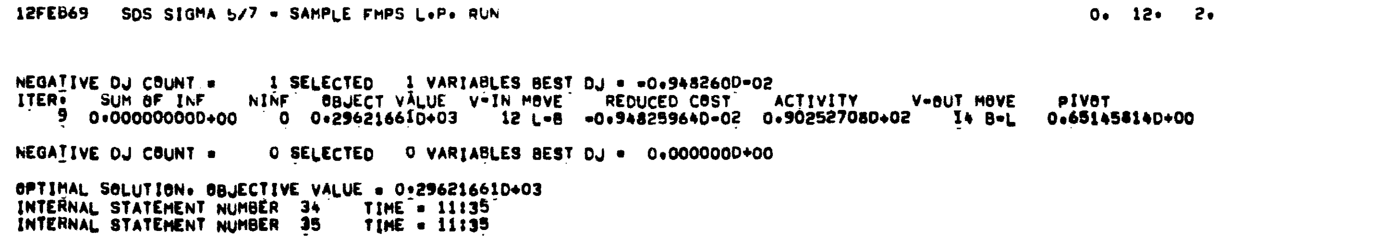
```
12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                      0.  10.   1.

INTERNAL STATEMENT NUMBER  25     TIME = 11:35
```

```
12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                      0.  11.   1.


COLUMN VALUE      NUMBER       1 LOWER        0.0000 UPPER .NONE.
    1.0000(VALUE     )

COLUMN YIELD      NUMBER       2 LOWER        0.0000 UPPER        0.0000
    1.0000(YIELD     )

COLUMN FE         NUMBER       3 LOWER        0.0000 UPPER .NONE.
    1.0000(FE        )

COLUMN MN         NUMBER       4 LOWER        0.0000 UPPER .NONE.
    1.0000(MN        )

COLUMN CU         NUMBER       5 LOWER        0.0000 UPPER .NONE.
    1.0000(CU        )

COLUMN MG         NUMBER       6 LOWER        0.0000 UPPER .NONE.
    1.0000(MG        )

COLUMN AL         NUMBER       7 LOWER        0.0000 UPPER .NONE.
   -1.0000(AL        )

COLUMN SI         NUMBER       8 LOWER        0.0000 UPPER       50.0000
    1.0000(SI        )

COLUMN DELCST     NUMBER       9 LOWER        0.0000 UPPER .NONE.
    1.0000(DELCST    )

COLUMN BIN1       NUMBER      10 LOWER        0.0000 UPPER      200.0000
    0.0300(VALUE     )     1.0000(YIELD     )     0.1500(FE        )     0.0200(MN        )     0.0300(CU        )     0.0200(MG        )
    0.7000(AL        )     0.0200(SI        )   -10.0000(DELCST    )

COLUMN BIN2       NUMBER      11 LOWER        0.0000 UPPER     2500.0000
    0.0800(VALUE     )     1.0000(YIELD     )     0.0400(FE        )     0.0400(MN        )     0.0500(CU        )     0.0300(MG        )
    0.7500(AL        )     0.0600(SI        )

COLUMN BIN3       NUMBER      12 LOWER      400.0000 UPPER      800.0000
    0.1700(VALUE     )     1.0000(YIELD     )     0.0200(FE        )     0.0100(MN        )     0.0800(CU        )     0.8000(AL        )
    0.0800(SI        )

COLUMN BIN4       NUMBER      13 LOWER      100.0000 UPPER      700.0000
    0.1200(VALUE     )     1.0000(YIELD     )     0.0400(FE        )     0.0200(MN        )     0.0200(CU        )     0.7500(AL        )
    0.1200(SI        )

COLUMN BIN5       NUMBER      14 LOWER        0.0000 UPPER     1500.0000
    0.1500(VALUE     )     1.0000(YIELD     )     0.0200(FE        )     0.0200(MN        )     0.0600(CU        )     0.0100(MG        )
    0.8000(AL        )     0.0200(SI        )

COLUMN ALUM       NUMBER      15 LOWER        0.0000 UPPER .NONE.
    0.2100(VALUE     )     1.0000(YIELD     )     0.0100(FE        )     0.0100(CU        )     0.9700(AL        )     0.0100(SI        )
```

```
12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                      0.  11.   2.


COLUMN SILCON     NUMBER      16 LOWER        0.0000 UPPER .NONE.
    0.3800(VALUE     )     1.0000(YIELD     )     0.0300(FE        )     0.9700(SI        )

COLUMN ALOY1      NUMBER      17 LOWER        0.0000 UPPER        0.0000
 2000.0000(YIELD     )    60.0000(FE        )    40.0000(MN        )   100.0000(CU        )    30.0000(MG        )  1500.0000(AL        )
  300.0000(SI        )

COLUMN DELPRODC NUMBER        18 LOWER        0.0000 UPPER        0.0000
20000.0000(YIELD     )
```

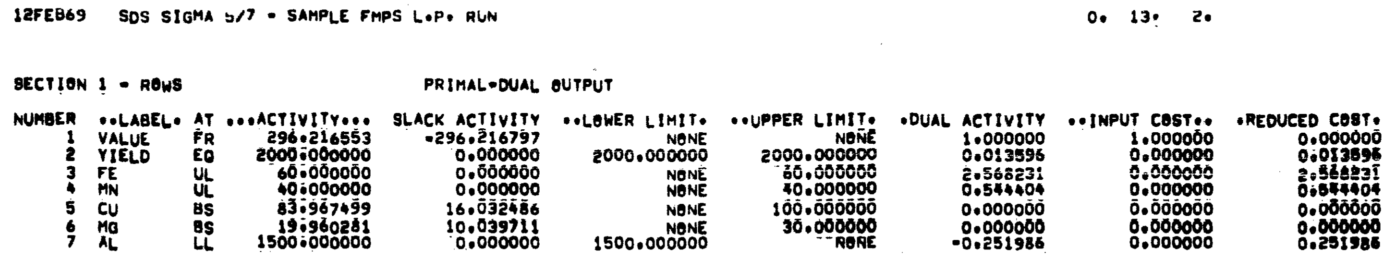


```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                              0.  12.   1.


INTERNAL STATEMENT NUMBER  26    TIME = 11:35
INTERNAL STATEMENT NUMBER  27    TIME = 11:35
INTERNAL STATEMENT NUMBER  28    TIME = 11:35
INTERNAL STATEMENT NUMBER  29    TIME = 11:35
INTERNAL STATEMENT NUMBER  30    TIME = 11:35
INTERNAL STATEMENT NUMBER  31    TIME = 11:35
INTERNAL STATEMENT NUMBER  32    TIME = 11:35
INTERNAL STATEMENT NUMBER  33    TIME = 11:35

NEGATIVE DJ CØUNT =     7 SELECTED   4 VARIABLES BEST DJ = -0.198000D+01
ITER.   SUM ØF INF     NINF   ØBJECT VALUE  V-IN MØVE    REDUCED CØST    ACTIVITY      V-ØUT MØVE    PIVØT
   1  0.281100000D+04    3  0.395000000D+03   15 L-B   0.210000000D+00  0.190000000D+04    2 B-L   0.100000000D+01
   2  0.191000000D+03    1  0.428822920D+03   16 L-B   0.170000000D+00  0.198958330D+03    8 B-U   0.960000000D+00
SØLUTIØN FEASIBLE AT ITERATIØN      2

NEGATIVE DJ CØUNT =     5 SELECTED   3 VARIABLES BEST DJ = -0.181771D+00
ITER.   SUM ØF INF     NINF   ØBJECT VALUE  V-IN MØVE    REDUCED CØST    ACTIVITY      V-ØUT MØVE    PIVØT
   3  0.000000000D+00    0  0.392468750D+03   10 L-U  -0.181770830D+00  0.200000000D+03 NØNE
   4  0.000000000D+00    0  0.388270680D+03   13 L-B  -0.109479170D+00  0.383458650D+02    3 B-L   0.277083330D-01
INTERNAL STATEMENT NUMBER  56    TIME = 11:35

   3 NØN-BASIC SLACKS. CØMPLETELY TRIANGULARIZED    0 RØWS AND    6 CØLS.
   3 IN NØN-CØMPLETELY TRIANGULARIZED PART. ØF THESE    2 WHERE NØT TRIANGULARIZED AND    0 WERE REJECTED FØR TØØ SMALL A PIVØT.
MATRIX TØ BE INVERTED HAD    9 CØLS AND     23 ELEMENTS. INVERSE HAS   7 CØLS AND     21 ELEMENTS.

 1800 MS FØR INVERT
INTERNAL STATEMENT NUMBER  57    TIME = 11:35
INTERNAL STATEMENT NUMBER  33    TIME = 11:35

NEGATIVE DJ CØUNT =     4 SELECTED   2 VARIABLES BEST DJ = -0.370564D+00
ITER.   SUM ØF INF     NINF   ØBJECT VALUE  V-IN MØVE    REDUCED CØST    ACTIVITY      V-ØUT MØVE    PIVØT
   5  0.000000000D+00    0  0.347017140D+03   10 U-B  -0.370563910D+00  0.111326380D+03   13 B-U   0.504511280D+01
   6  0.000000000D+00    0  0.323257010D+03   14 L-B  -0.490387480D-01  0.484517490D+03    7 B-L   0.141691510D+00

NEGATIVE DJ CØUNT =     2 SELECTED   1 VARIABLES BEST DJ = -0.611770D-01
ITER.   SUM ØF INF     NINF   ØBJECT VALUE  V-IN MØVE    REDUCED CØST    ACTIVITY      V-ØUT MØVE    PIVØT
   7  0.000000000D+00    0  0.300931280D+03   11 L-B  -0.611769660D-01  0.364936890D+03   10 B-L   0.149986850D+00

NEGATIVE DJ CØUNT =     2 SELECTED   1 VARIABLES BEST DJ = -0.156802D-01
ITER.   SUM ØF INF     NINF   ØBJECT VALUE  V-IN MØVE    REDUCED CØST    ACTIVITY      V-ØUT MØVE    PIVØT
   8  0.000000000D+00    0  0.297072440D+03   13 U-B  -0.156802240D-01  0.246095950D+03    4 B-L  -0.149088360D-01
INTERNAL STATEMENT NUMBER  56    TIME = 11:35

   5 NØN-BASIC SLACKS. CØMPLETELY TRIANGULARIZED    0 RØWS AND    4 CØLS.
   5 IN NØN-CØMPLETELY TRIANGULARIZED PART. ØF THESE    2 WHERE NØT TRIANGULARIZED AND    0 WERE REJECTED FØR TØØ SMALL A PIVØT.
MATRIX TØ BE INVERTED HAD    9 CØLS AND     37 ELEMENTS. INVERSE HAS  10 CØLS AND      38 ELEMENTS.

  600 MS FØR INVERT
INTERNAL STATEMENT NUMBER  57    TIME = 11:35
INTERNAL STATEMENT NUMBER  33    TIME = 11:35
```



```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                              0.  12.   2.


NEGATIVE DJ CØUNT =     1 SELECTED   1 VARIABLES BEST DJ = -0.948260D-02
ITER.   SUM ØF INF     NINF   ØBJECT VALUE  V-IN MØVE    REDUCED CØST    ACTIVITY      V-ØUT MØVE    PIVØT
   9  0.000000000D+00    0  0.296216610D+03   12 L-B  -0.948259640D-02  0.902527080D+02   14 B-L   0.651458140D+00

NEGATIVE DJ CØUNT =     0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00

ØPTIMAL SØLUTIØN. ØBJECTIVE VALUE = 0.296216610D+03
INTERNAL STATEMENT NUMBER  34    TIME = 11:35
INTERNAL STATEMENT NUMBER  35    TIME = 11:35
```

```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                              0.  13.   1.


IDENTIFIER SECTIØN

          PRØBLEM... NAME.. FUSIØN
                     MØDE.. LP
                     CLASS. LP
                     STATUS ØPTIMAL.
          FUNCTIØNAL NAME.. VALUE
                     ØBJECT MINIMIZE
                     VALUE.       296.216553
          RESTRAINT. NAME.. ALØY1
          ITERATIØN. CØUNT.         9
```



```
12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                              0.  13.   2.


SECTIØN 1 - RØWS                       PRIMAL-DUAL ØUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  SLACK ACTIVITY  ..LØWER LIMIT.  ..UPPER LIMIT.  .DUAL ACTIVITY  ..INPUT CØST..  .REDUCED CØST.
     1  VALUE    FR     296.216553     -296.216797            NØNE            NØNE        1.000000        1.000000        0.000000
     2  YIELD    EQ    2000.000000        0.000000     2000.000000     2000.000000        0.013596        0.000000        0.013596
     3  FE       UL      60.000000        0.000000            NØNE       60.000000        2.568231        0.000000        2.568231
     4  MN       UL      40.000000        0.000000            NØNE       40.000000        0.544404        0.000000        0.544404
     5  CU       BS      83.967499       16.032486            NØNE      100.000000        0.000000        0.000000        0.000000
     6  MG       BS      19.960281       10.039711            NØNE       30.000000        0.000000        0.000000        0.000000
     7  AL       LL    1500.000000        0.000000     1500.000000            NØNE       -0.251986        0.000000        0.251986
     8 SI       LL    250.000000        50.000000      250.000000      300.000000       -0.485199        0.000000       -0.485199
     9 DELCST   FR      0.000000         0.000000            NONE            NONE        0.000000        0.000000        0.000000
________________________________________________________________________________________________________________________________

12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                          0.  13.   3.


SECTION 2 - COLUMNS                    PRIMAL-DUAL OUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  ..INPUT COST..  ..LOWER LIMIT.  ..UPPER LIMIT.  .REDUCED COST.
    10  BIN1     LL       0.000000        0.030000        0.000000      200.000000        0.253624
    11  BIN2     BS     665.342773        0.080000        0.000000     2500.000000        0.000000
    12  BIN3     BS     490.252686        0.170000      400.000000      800.000000        0.000000
    13  BIN4     BS     424.187500        0.120000      100.000000      700.000000        0.000000
    14  BIN5     LL       0.000000        0.150000        0.000000     1500.000000        0.014556
    15  ALUM     BS     299.638916        0.210000        0.000000            NONE        0.000000
    16  SILCON   BS     120.577606        0.380000        0.000000            NONE        0.000000
    18  DELPRODC EQ       0.000000        0.000000        0.000000        0.000000      271.913330
________________________________________________________________________________________________________________________________

12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                          0.  14.   1.


INTERNAL STATEMENT NUMBER  36     TIME = 11:35
________________________________________________________________________________________________________________________________

12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                          0.  16.   1.


PRIMAL ERRORS

NUMBER ..NAME..       ERROR              RHS
     1 VALUE     0.790265630-10  0.000000000D+00
     3 FE        0.218136620-11  0.600000000D+02
     4 MN        0.151345600-11  0.400000000D+02
     5 CU        0.293667310-10  0.100000000D+03
     6 MG        0.236610730-11  0.300000000D+02
     7 AL        0.172238560-10  0.150000000D+04
     8 SI        0.612203620-10  0.300000000D+03

MAXIMUM ERRORS
DUAL... 0.000000000D+00
PRIMAL. 0.790265630-10
INTERNAL STATEMENT NUMBER  37     TIME = 11:36
________________________________________________________________________________________________________________________________

12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                          0.  17.   1.


                                     RANGES FOR VARIABLES AT LIMIT LEVEL

                                                                   ----------LOWER----------- ----------UPPER-----------
NUMBER AT ..NAME.. .LOWER LIMIT.. .UPPER LIMIT.. .REDUCED COST. PROCESS. ..INCREMENT... AT PROCESS. ..INCREMENT... AT
     2 EQ YIELD      2000.000000    2000.000000       0.013596 BIN3         -4.931356 LL CU             14.034788 LL
     3 UL FE                NONE      60.000000       2.568231 BIN4         -4.109840 LL BIN3            2.699783 LL
     4 UL MN                NONE      40.000000       0.544404 BIN4         -5.576643 UL BIN3            1.686909 LL
     7 LL AL         1500.000000           NONE       0.251986 CU          -14.215750 LL BIN3            4.921259 LL
     8 LL SI          250.000000     300.000000       0.485198 CU          -14.671292 LL BIN3            5.060728 LL
       **
    10 LL BIN1          0.000000     200.000000       0.253624 BIN4        -28.824753 UL BIN4           33.880386 LL
    14 LL BIN5          0.000000    1500.000000       0.014556 BIN3       -201.787399 UL BIN3           58.795853 LL
________________________________________________________________________________________________________________________________

12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                          0.  18.   1.


                               RANGES FOR VARIABLES AT INTERMEDIATE LEVEL

                                                      ----------LOWER----------- ----------UPPER-----------
NUMBER AT ..NAME.. ...ACTIVITY... ..INPUT COST.. PROCESS. ..INCREMENT... AT PROCESS. ..INCREMENT... AT
     1 BS VALUE      -296.216797       1.000000                     NONE    FE             1.000000 LL
     5 BS CU           16.032486       0.000000 BIN5           -0.214742 LL MN             0.306131 LL
     6 BS MG           10.039711       0.000000 MN             -0.287567 LL BIN1           1.796180 LL
     9 BS DELCST        0.000000       0.000000 BIN1           -0.025362 LL                    NONE
       **
    11 BS BIN2        665.342773       0.080000 BIN1           -0.062777 LL MN             0.008627 LL
    12 BS BIN3        490.252686       0.170000 MN             -0.010175 LL BIN5           0.009483 LL
    13 BS BIN4        424.187500       0.120000 MN             -0.011007 LL BIN1           0.026506 LL
    15 BS ALUM        299.638916       0.210000 AL             -0.021152 LL MN             0.016215 LL
    16 BS SILCON      120.577606       0.380000 SI             -0.231724 UL MN             0.086667 LL
________________________________________________________________________________________________________________________________

12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                          0.  19.   1.


INTERNAL STATEMENT NUMBER  38     TIME = 11:36
INTERNAL STATEMENT NUMBER  39     TIME = 11:36
INTERNAL STATEMENT NUMBER  40     TIME = 11:36
INTERNAL STATEMENT NUMBER  41     TIME = 11:36
INTERNAL STATEMENT NUMBER  42     TIME = 11:36
INTERNAL STATEMENT NUMBER  43     TIME = 11:36
INTERNAL STATEMENT NUMBER  44     TIME = 11:36

ITER.   SUM OF INF      NINF   OBJECT  VALUE   V-IN MOVE     REDUCED COST     ACTIVITY       V-OUT MOVE     PIVOT            THETA
   10  0.000000000D+00     0  0.29621661D+03    10 L-B    0.25362455D+00  0.33880400D+02     13 B-L   0.95685921D+01  0.25362455D-01
   11  0.000000000D+00     0  0.29264818D+03     4 L-B    0.54440433D+00  0.43709135D+01      5 B-L   0.25343143D+01  0.35894891D-01
   12  0.000000000D+00     0  0.28975136D+03    14 L-B    0.89927007D-02  0.53425573D+03     11 B-L   0.11561365D+01  0.41024457D-01
```

```
INTERNAL STATEMENT NUMBER  65    TIME = 11:36

   5 NON-BASIC SLACKS. COMPLETELY TRIANGULARIZED    0 ROWS AND    4 COLS.
   5 IN NON-COMPLETELY TRIANGULARIZED PART. OF THESE    3 WHERE NOT TRIANGULARIZED AND    0 WERE REJECTED FOR TOO SMALL A PIVOT.
MATRIX TO BE INVERTED HAD    9 COLS AND     38 ELEMENTS. INVERSE HAS  10 COLS AND     39 ELEMENTS.

  600 MS FOR INVERT
INTERNAL STATEMENT NUMBER  66    TIME = 11:36

NEGATIVE DJ COUNT =     0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00

OPTIMAL SOLUTION. OBJECTIVE VALUE = 0.28975136D+03
INTERNAL STATEMENT NUMBER  67    TIME = 11:36
INTERNAL STATEMENT NUMBER  44    TIME = 11:36

ITER.   SUM OF INF     NINF   OBJECT  VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT           THETA
   13  0.00000000D+00    0   0.25748148D+03    5 L-B   0.10461957D+00  0.13398107D+02    12 B-L   0.24484934D+02  0.62518519D-01
   14  0.00000000D+00    0   0.24574468D+03    7 L-B   0.37037037D-01  0.87680851D+02    10 B-U  -0.49434857D+00  0.70010638D-01
NO MAXIMUM PARAMETER AT THETA=  0.7001060D-01
INTERNAL STATEMENT NUMBER  45    TIME = 11:36
________________________________________________________________________________________________________________________________

12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                                   0.  20.   1.


IDENTIFIER SECTION

          PROBLEM... NAME.. FUSION
                     MODE.. LP
                     CLASS. LP
                     STATUS OPTIMAL.
          FUNCTIONAL NAME.. VALUE
                     OBJECT MINIMIZE
                     VALUE.   -19614.234375
          RESTRAINT. NAME.. ALOY1
          ITERATION. COUNT.         14
          PARAMETRIC MODE.. COST
                     NAME.. DELCST
                     VALUE.         10.000000
________________________________________________________________________________________________________________________________

12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                                   0.  20.   2.


SECTION 1 - ROWS                        PRIMAL-DUAL OUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  SLACK ACTIVITY  ..LOWER LIMIT.  ..UPPER LIMIT.  .DUAL ACTIVITY  ..INPUT COST..  .REDUCED COST.
     1  VALUE    FR    385.765869     -385.766113            NONE            NONE        1.000000        1.000000        0.000000
     2  YIELD    EQ   2000.000000        0.000000     2000.000000     2000.000000       -0.270000        0.000000       -0.270000
     3  FE       UL     60.000000        0.000000            NONE       60.000000        6.308510        0.000000        6.308510
     4  MN       BS     12.170213       27.829773            NONE       40.000000        0.000000        0.000000        0.000000
     5  CU       BS     56.468079       43.531906            NONE      100.000000        0.000000        0.000000        0.000000
     6  MG       BS      5.085106       24.914886            NONE       30.000000        0.000000        0.000000        0.000000
     7  AL       BS   1587.680664       87.680847     1500.000000            NONE        0.000000        0.000000        0.000000
     8  SI       LL    250.000000       50.000000      250.000000      300.000000       -0.308511        0.000000       -0.308511
     9  DELCST   FR  -2000.000000     1999.999756            NONE            NONE       10.000000       10.000000        0.000000
________________________________________________________________________________________________________________________________

12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                                   0.  20.   3.


SECTION 2 - COLUMNS                     PRIMAL-DUAL OUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  ..INPUT COST..  ..LOWER LIMIT.  ..UPPER LIMIT.  .REDUCED COST.
    10  BIN1     UL    200.000000      -99.970001        0.000000      200.000000      -99.299896
    11  BIN2     LL      0.000000        0.080000        0.000000     2500.000000        0.043830
    12  BIN3     LL    400.000000        0.170000      400.000000      800.000000        0.001489
    13  BIN4     LL    100.000000        0.120000      100.000000      700.000000        0.065319
    14  BIN5     BS    108.510635        0.150000        0.000000     1500.000000        0.000000
    15  ALUM     BS    995.744629        0.210000        0.000000            NONE        0.000000
    16  SILCON   BS    195.744675        0.380000        0.000000            NONE        0.000000
    18  DELPRODC EQ      0.000000        0.000000        0.000000        0.000000    -5400.003906
________________________________________________________________________________________________________________________________

12FEB69   SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                                   0.  21.   1.

INTERNAL STATEMENT NUMBER  46    TIME = 11:36
INTERNAL STATEMENT NUMBER  47    TIME = 11:36
INTERNAL STATEMENT NUMBER  48    TIME = 11:36
INTERNAL STATEMENT NUMBER  49    TIME = 11:36
INTERNAL STATEMENT NUMBER  50    TIME = 11:36
INTERNAL STATEMENT NUMBER  51    TIME = 11:36

ITER.   SUM OF INF     NINF   OBJECT  VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT           THETA
   10  0.00000000D+00    0   0.29602579D+03   14 L-B   0.14555957D-01  0.00000000D+00     5 B-L  -0.67783394D-01  0.70173942D-03
   11  0.00000000D+00    0   0.30315439D+03    7 L-B   0.97997444D-02  0.00000000D+00    15 B-L  -0.63272262D+01  0.22399783D-02
   12  0.00000000D+00    0   0.31701877D+03   15 L-B   0.15488215D-02  0.00000000D+00    12 B-L  -0.11380471D+01  0.51365188D-02
INTERNAL STATEMENT NUMBER  65    TIME = 11:36

   5 NON-BASIC SLACKS. COMPLETELY TRIANGULARIZED    0 ROWS AND    4 COLS.
   5 IN NON-COMPLETELY TRIANGULARIZED PART. OF THESE    2 WHERE NOT TRIANGULARIZED AND    0 WERE REJECTED FOR TOO SMALL A PIVOT.
MATRIX TO BE INVERTED HAD    9 COLS AND     37 ELEMENTS. INVERSE HAS  11 COLS AND     39 ELEMENTS.

  600 MS FOR INVERT
INTERNAL STATEMENT NUMBER  66    TIME = 11:36
```

```
NEGATIVE DJ COUNT =      0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00

OPTIMAL SOLUTION. OBJECTIVE VALUE = 0.31701877D+03
INTERNAL STATEMENT NUMBER  67     TIME = 11:36
INTERNAL STATEMENT NUMBER  51     TIME = 11:36

ITER.   SUM OF INF      NINF   OBJECT  VALUE  V-IN MOVE     REDUCED COST     ACTIVITY       V-OUT MOVE     PIVOT            THETA
   13  0.000000000D+00    0   0.44783129D+03     4 L-B   0.90828402D+00  0.000000000D+00     13 B-L  -0.74852071D+02  0.32162577D-01
   14  0.000000000D+00    0   0.89044149D+03     5 L-B   0.40711462D+00  0.00000000D+00      11 B-L  -0.92885375D+01  0.11938830D+00
PREMATURE MAXIMUM AT THETA=  0.119388D+00
INTERNAL STATEMENT NUMBER  52     TIME = 11:36
```

```
12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                                   0.  22.   1.


IDENTIFIER SECTION

             PROBLEM... NAME.. FUSION
                        MODE.. LP
                        CLASS. LP
                        STATUS OPTIMAL.
             FUNCTIONAL NAME.. VALUE
                        OBJECT MINIMIZE
                        VALUE.        890.441406
             RESTRAINT. NAME.. ALOY1
             ITERATION. COUNT.          14
             PARAMETRIC MODE.. RHS
                        NAME.. DELPRODC
                        VALUE.          0.119388
```

```
12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                                   0.  22.   2.


SECTION 1 - ROWS                          PRIMAL-DUAL OUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  SLACK ACTIVITY  ..LOWER LIMIT.  ..UPPER LIMIT.  .DUAL ACTIVITY  ..INPUT COST..  .REDUCED COST.
     1  VALUE    FR    890.441406     -890.441650            NONE            NONE        1.000000        1.000000        0.000000
     2  YIELD    EQ   2000.000000       0.000000      2000.000000     2000.000000       -0.270000        0.000000       -0.270000
     3  FE       UL     60.000000       0.000000             NONE       60.000000        6.308510        0.000000        6.308510
     4  MN       BS     17.521271      22.478714             NONE       40.000000        0.000000        0.000000        0.000000
     5  CU       BS    100.000000       0.000000             NONE      100.000000        0.000000        0.000000        0.000000
     6  MG       BS      5.760638      24.239349             NONE       30.000000        0.000000        0.000000        0.000000
     7  AL       BS   3905.159424    2405.159424      1500.000000            NONE        0.000000        0.000000        0.000000
     8  SI       LL    250.000000      50.000000       250.000000      300.000000       -0.308511        0.000000       -0.308511
     9  DELCST   FR      0.000000       0.000000             NONE            NONE        0.000000        0.000000        0.000000
```

```
12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                                   0.  22.   3.


SECTION 2 - COLUMNS                       PRIMAL-DUAL OUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  ..INPUT COST..  ..LOWER LIMIT.  ..UPPER LIMIT.  .REDUCED COST.
    10  BIN1     LL      0.000000        0.030000        0.000000      200.000000        0.700106
    11  BIN2     LL      0.000000        0.080000        0.000000     2500.000000        0.043830
    12  BIN3     LL    400.000000        0.170000      400.000000      800.000000        0.001489
    13  BIN4     LL    100.000000        0.120000      100.000000      700.000000        0.065319
    14  BIN5     BS    576.063721        0.150000        0.000000     1500.000000        0.000000
    15  ALUM     BS   3143.616943        0.210000        0.000000            NONE        0.000000
    16  SILCON   BS    168.085098        0.380000        0.000000            NONE        0.000000
    18  DELPRODC EQ      0.000000        0.000000        0.000000        0.000000    -5400.003906
```

```
 12FEB69    SDS SIGMA 5/7 - SAMPLE FMPS L.P. RUN                                                                  0.  23.   1.

 INTERNAL STATEMENT NUMBER  53     TIME = 11:36
.EXIT.
```

```
 TOTAL JOB TIME                        1.30
  PROCESSOR EXECUTION TIME              .01
  PROCESSOR I/O TIME                    .07
  PROCESSOR OVERHEAD TIME               .07
  USER EXECUTION TIME                   .48
  USER I/O TIME                         .52
  USER OVERHEAD TIME                    .75
 # OF CARDS READ                        316
 # OF CARDS PUNCHED                       0
 # OF PROCESSOR PAGES OUT                 2
 # OF USER PAGES OUT                     35
 # OF DIAGNOSTIC PAGES OUT                0
 # OF SCRATCH TAPES USED                  0
 # OF SAVE TAPES USED                     0
 # OF DISK READS AND WRITES            1594
 # OF DISC READS AND WRITES            2957
 TEMPORARY DISC SPACE USED               17
 PERMANENT DISC SPACE USED                0
 ACCUM. PERM. DISC SPACE USED             0
```

```
 JOB 326,SDMD
 LIMIT (TIME,90),(LO,1000),(UO,1000),(DO,1000)
```

```
ASSIGN F:106,(DEVICE,CPA04)
ASSIGN F:1,(FILE,CLANG),(BIN),(WRITE,ALL),(CONSEC),(SEQUEN),;
(OUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:2,(FILE,UTIL1),(BIN),(WRITE,ALL),(KEYED),(DIRECT),;
(OUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:3,(FILE,UTIL2),(BIN),(WRITE,ALL),(KEYED),(DIRECT),;
(OUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:4,(FILE,MTRX),(BIN),(WRITE,ALL),(DIRECT),(KEYED),;
(OUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:5,(FILE,IVSE),(BIN),(WRITE,ALL),(DIRECT),(KEYED),;
(OUTIN),(RECL,30000),(READ,ALL)
RUN (LMN,FMPS)
DATA
```

```
C
C      DEFINE HEADING AND ENTER SEPERABLE PROGRAMMING MODE
C
       TITLE NON-LINEAR PROBLEM NO 6
       CALL ENTER(SEP)
C
C      SPECIFY FOUR SYMBOLIC UNITS(WORKING FILES) ON RAD
C
       CALL DEVICE('FILE1',DISC,'B')
       CALL DEVICE('FILE2',DISC,'C')
       CALL DEVICE('FILE3',DISC,'D')
       CALL DEVICE('FILE4',DISC,'E')
C
C      ATTACH THE FOUR STANDARD FMPS FILES TO THE FOUR
C      PREVIOUSLY DEFINED SYMBOLIC UNITS(RAD).
C
       CALL ATTACH(MATRIX,'FILE1')
       CALL ATTACH(INVERSE,'FILE2')
       CALL ATTACH(UTIL1,'FILE3')
       CALL ATTACH(UTIL2,'FILE4')
C
C      INITIALIZE INTERRUPT VARIABLES
C
       ASSIGN 100 TO KMAJER
       ASSIGN 200 TO KIOER
       ASSIGN 300 TO KNFS
       ASSIGN 400 TO KUBS
       ASSIGN 500 TO KINV
C
       ADATA = 'NLPSTD01'
C
C      LOAD INPUT MATRIX FROM CARDS, USING RECORD 'NLPSTD01'
C
       CALL INPUT
C
C      IDENTIFY RIGHT-HAND-SIDE COLUMN AND COST ROW TO BE USED
C
       AOBJ = 'OBJT    '
       ARHS = '1RHS    '
C
C      SET TO INVERT NO LESS FREQUENTLY THAN AT INTERVALS OF
C      50 ITERATIONS(NOTE: AUTOMATIC INVERT ON TIME IS BY DEFAULT
C      IN OPERATION.).
C
C   *****
C    NOTE: TO TURN OFF THE AUTOMATIC INVERT ON TIME, THE FOLLOWING
C           STATEMENT SHOULD BE USED.
C      INVTIME = -1
C   *****
C
       IFREQI = 50
C
C      INITIALIZE ITERATION LOGGING FREQUENCY TO PRINT EVERY ITERATION
```

```
C
       ILOGP = 1
C
C      SPECIFY MINIMIZATION
C
       FOBJWT = 1.
C
C      SOLVE SEPERABLE MATRIX
C
       CALL OPTIMIZE
C
C      DISPLAY PROBLEM SOLUTION
C
       CALL SOLUTION
       STOP
C
C      ENTER HERE FOR MAJOR ERROR CONDITIONS
C
   100 CALL CONDITION
       STOP
C
C      ENTER HERE FOR I/O ERROR CONDITION
C
   200 CALL CONDITION
       STOP
C
C      ENTER HERE FOR NO FEASIBLE SOLUTION CONDITION
C
   300 CALL CONDITION
C
C      ENTER HERE FOR UNBOUNDED SOLUTION CONDITION
```

```
C
  400 CALL SØLUTIØN
      STØP
C
C     ENTER HERE FØR INVERSIØN INTERRUPT CØNDITIØN
C
  500 CALL INVERT
      RETURN
      END
```

```
NAME          NLPSTD01
RØWS
 N ØBJT
 E RØW1
 E RØW2
 E RØW3
 E RØW4
 E RØW5
 E RØW6
 E RØW7
 E RØW8
 E RØW9
CØLUMNS
    X5        RØW4        200.
    UBØUND1   'MARKER'                   'SEPØRG'
    U1        ØBJT         -9.
    U1        RØW1         30.
    U2        ØBJT         -3.1
    U2        RØW1         10.
    UBØUND2   'MARKER'                   'SEPØRG'
    U3        ØBJT         -2.8
    U3        RØW2         10.
    U4        ØBJT         -2.9
    U4        RØW2         10.
    U5        ØBJT        -24.
    U5        RØW2         80.
    $3BØUND   'MARKER'                   'SEPØRG'
    3$ 1      RØW1        -.179619
    3$ 1      RØW3        -1.88670
    3$ 1      RØW5        -.50732
    3$ 1      RØW6        -.50732
    3$ 1      RØW7        -.50732
    3$ 1      RØW8        -.50732
    3$ 1      RØW9        -.50732
    3$ 2      RØW1        -.181719
    3$ 2      RØW3        -1.90877
    3$ 2      RØW5        -.502080
    3$ 2      RØW6        -.502080
    3$ 2      RØW7        -.502080
    3$ 2      RØW8        -.502080
    3$ 2      RØW9        -.502080
    3$ 3      RØW1        -.18382
    3$ 3      RØW3        -1.93084
    3$ 3      RØW5        -.496350
    3$ 3      RØW6        -.496350
    3$ 3      RØW7        -.496350
    3$ 3      RØW8        -.496350
    3$ 3      RØW9        -.496350
    3$ 4      RØW1        -.185921
    3$ 4      RØW3        -1.9529
    3$ 4      RØW5        -.490730
    3$ 4      RØW6        -.490730
    3$ 4      RØW7        -.490730
```

```
    3$ 4      RØW8        -.490730
    3$ 4      RØW9        -.490730
    3$ 5      RØW1        -.188022
    3$ 5      RØW3        -1.97497
    3$ 5      RØW5        -.485250
    3$ 5      RØW6        -.485250
    3$ 5      RØW7        -.485250
    3$ 5      RØW8        -.485250
    3$ 5      RØW9        -.485250
    3$ 6      RØW1        -.237982
    3$ 6      RØW3        -2.49974
    3$ 6      RØW5        -.599040
    3$ 6      RØW6        -.599040
    3$ 6      RØW7        -.599040
    3$ 6      RØW8        -.599040
    3$ 6      RØW9        -.599040
    3$ 7      RØW1        -.241133
    3$ 7      RØW3        -2.53422
    3$ 7      RØW5        -.590880
    3$ 7      RØW6        -.590880
    3$ 7      RØW7        -.590880
    3$ 7      RØW8        -.590880
    3$ 7      RØW9        -.590880
    3$ 8      RØW1        -.244547
    3$ 8      RØW3        -2.56870
    3$ 8      RØW5        -.582960
    3$ 8      RØW6        -.582960
    3$ 8      RØW7        -.582960
    3$ 8      RØW8        -.582960
    3$ 8      RØW9        -.582960
    3$ 9      RØW1        -.247829
    3$ 9      RØW3        -2.60318
    3$ 9      RØW5        -.575230
    3$ 9      RØW6        -.575230
    3$ 9      RØW7        -.575230
```

```
3$ 9      RØW8       -.575230
3$ 9      RØW9       -.575230
3$10      RØW1       -.301728
3$10      RØW3       -3.16933
3$10      RØW5       -.680370
3$10      RØW6       -.680370
3$10      RØW7       -.680370
3$10      RØW8       -.680370
3$10      RØW9       -.680370
3$11      RØW1       -.306455
3$11      RØW3       -3.21898
3$11      RØW5       -.669880
3$11      RØW6       -.669880
3$11      RØW7       -.669880
3$11      RØW8       -.669880
3$11      RØW9       -.669880
3$12      RØW1       -.311181
3$12      RØW3       -3.26863
```

```
3$12      RØW5       -.659700
3$12      RØW6       -.659700
3$12      RØW7       -.659700
3$12      RØW8       -.659700
3$12      RØW9       -.659700
3$13      RØW1       -.315908
3$13      RØW3       -3.31828
3$13      RØW5       -.649830
3$13      RØW6       -.649830
3$13      RØW7       -.649830
3$13      RØW8       -.649830
3$13      RØW9       -.649830
3$14      RØW1       -.320635
3$14      RØW3       -3.36793
3$14      RØW5       -.640250
3$14      RØW6       -.640250
3$14      RØW7       -.640250
3$14      RØW8       -.640250
3$14      RØW9       -.640250
3$15      RØW1       -.270807
3$15      RØW3       -2.84453
3$15      RØW5       -.526420
3$15      RØW6       -.526420
3$15      RØW7       -.526420
3$15      RØW8       -.526420
3$15      RØW9       -.526420
3$16      RØW1       -.274089
3$16      RØW3       -2.87901
3$16      RØW5       -.520120
3$16      RØW6       -.520120
3$16      RØW7       -.520120
3$16      RØW8       -.520120
3$16      RØW9       -.520120
$$BOUND   'MARKER'                  'SEPØRG'
4$ 1      RØW2       -.179619
4$ 1      RØW4       -1.88670
4$ 1      RØW5       -.507320
4$ 1      RØW6       -.507320
4$ 1      RØW7       -.507320
4$ 1      RØW8       -.507320
4$ 1      RØW9       -.507320
4$ 2      RØW2       -.181719
4$ 2      RØW4       -1.90877
4$ 2      RØW5       -.502080
4$ 2      RØW6       -.502080
4$ 2      RØW7       -.502080
4$ 2      RØW8       -.502080
4$ 2      RØW9       -.502080
4$ 3      RØW2       -.183820
4$ 3      RØW4       -1.93084
4$ 3      RØW5       -.496350
4$ 3      RØW6       -.496350
4$ 3      RØW7       -.496350
```

```
4$ 3      RØW8       -.496350
4$ 3      RØW9       -.496350
4$ 4      RØW2       -.185921
4$ 4      RØW4       -1.95290
4$ 4      RØW5       -.490370
4$ 4      RØW6       -.490370
4$ 4      RØW7       -.490370
4$ 4      RØW8       -.490370
4$ 4      RØW9       -.490370
4$ 5      RØW2       -.188022
4$ 5      RØW4       -1.97497
4$ 5      RØW5       -.485250
4$ 5      RØW6       -.485250
4$ 5      RØW7       -.485250
4$ 5      RØW8       -.485250
4$ 5      RØW9       -.485250
4$ 6      RØW2       -.237982
4$ 6      RØW4       -2.49974
4$ 6      RØW5       -.599040
4$ 6      RØW6       -.599040
4$ 6      RØW7       -.599040
4$ 6      RØW8       -.599040
4$ 6      RØW9       -.599040
4$ 7      RØW2       -.241133
4$ 7      RØW4       -2.53422
4$ 7      RØW5       -.590880
4$ 7      RØW6       -.590880
4$ 7      RØW7       -.590880
```

```
4S 7        ROW8        -.590880
4S 7        ROW9        -.590880
4S 8        ROW2        -.244547
4S 8        ROW4        -2.56870
4S 8        ROW5        -.582960
4S 8        ROW6        -.582960
4S 8        ROW7        -.582960
4S 8        ROW8        -.582960
4S 8        ROW9        -.582960
4S 9        ROW2        -.247829
4S 9        ROW4        -2.60318
4S 9        ROW5        -.575230
4S 9        ROW6        -.575230
4S 9        ROW7        -.575230
4S 9        ROW8        -.575230
4S 9        ROW9        -.575230
4S10        ROW2        -.301728
4S10        ROW4        -3.16933
4S10        ROW5        -.680370
4S10        ROW6        -.680370
4S10        ROW7        -.680370
4S10        ROW8        -.680370
4S10        ROW9        -.680370
4S11        ROW2        -.306455
4S11        ROW4        -3.21898
```

```
4S11        ROW5        -.669880
4S11        ROW6        -.669880
4S11        ROW7        -.669880
4S11        ROW8        -.669880
4S11        ROW9        -.669880
4S12        ROW2        -.311181
4S12        ROW4        -3.26863
4S12        ROW5        -.659700
4S12        ROW6        -.659700
4S12        ROW7        -.659700
4S12        ROW8        -.659700
4S12        ROW9        -.659700
4S13        ROW2        -.315908
4S13        ROW4        -3.31828
4S13        ROW5        -.649830
4S13        ROW6        -.649830
4S13        ROW7        -.649830
4S13        ROW8        -.649830
4S13        ROW9        -.649830
4S14        ROW2        -.320635
4S14        ROW4        -3.36793
4S14        ROW5        -.640250
4S14        ROW6        -.640250
4S14        ROW7        -.640250
4S14        ROW8        -.640250
4S14        ROW9        -.640250
4S15        ROW2        -.270807
4S15        ROW4        -2.84453
4S15        ROW5        -.526420
4S15        ROW6        -.526420
4S15        ROW7        -.526420
4S15        ROW8        -.526420
4S15        ROW9        -.526420
4S16        ROW2        -.274089
4S16        ROW4        -2.87901
4S16        ROW5        -.520120
4S16        ROW6        -.520120
4S16        ROW7        -.520120
4S16        ROW8        -.520120
4S16        ROW9        -.520120
$SBOUND     'MARKER'                            'SEPORG'
5S 1        ROW5        -2.05043
5S 1        ROW6         .18343
5S 1        ROW7         .016
5S 1        ROW8        -.01524
5S 2        ROW5        -2.69876
5S 2        ROW6         .25653
5S 2        ROW7         .02352
5S 2        ROW8        -.01993
5S 3        ROW5        -2.53943
5S 3        ROW6         .25816
5S 3        ROW7         .02499
5S 3        ROW8        -.01846
```

```
5S 4        ROW5        -2.39774
5S 4        ROW6         .25981
5S 4        ROW7         .02647
5S 4        ROW8        -.01699
5S 5        ROW5        -2.30899
5S 5        ROW6         .266
5S 5        ROW7         .02844
5S 5        ROW8        -.01578
5S 6        ROW5        -2.15475
5S 6        ROW6         .26318
5S 6        ROW7         .02945
5S 6        ROW8        -.01403
5S 7        ROW5        -2.05148
5S 7        ROW6         .26486
5S 7        ROW7         .03093
5S 7        ROW8        -.01257
5S 8        ROW5        -1.95753
5S 8        ROW6         .26656
5S 8        ROW7         .03242
5S 8        ROW8        -.01110
```

```
5S 9       ROW5      -1.87169
5S 9       ROW6       .26827
5S 9       ROW7       .03390
5S 9       ROW8      -.00964
5S10       ROW5      -1.79295
5S10       ROW6       .26999
5S10       ROW7       .03539
5S10       ROW8      -.00818
5S11       ROW5      -1.74951
5S11       ROW6       .27643
5S11       ROW7       .03753
5S11       ROW8      -.00682
5S12       ROW5      -1.65237
5S12       ROW6       .27351
5S12       ROW7       .03840
5S12       ROW8      -.00523
5S13       ROW5      -1.59038
5S13       ROW6       .27527
5S13       ROW7       .03990
5S13       ROW8      -.00377
5S14       ROW5      -1.53279
5S14       ROW6       .27705
5S14       ROW7       .04140
5S14       ROW8      -.00231
5S15       ROW5      -1.47912
5S15       ROW6       .27884
5S15       ROW7       .04290
5S15       ROW8      -.00084
5S16       ROW5      -1.42898
5S16       ROW6       .28064
5S16       ROW7       .04441
5S16       ROW8       .00062
5S17       ROW5      -1.40548
```

```
5S17       ROW6       .28734
5S17       ROW7       .04672
5S17       ROW8       .00212
5S18       ROW5      -1.99022
5S18       ROW6       .42717
5S18       ROW7       .07174
5S18       ROW8       .00589
5S19       ROW5      -1.89986
5S19       ROW6       .43134
5S19       ROW7       .07515
5S19       ROW8       .00918
5S20       ROW5      -1.81706
5S20       ROW6       .43556
5S20       ROW7       .07858
5S20       ROW8       .01247
5S21       ROW5      -1.18856
5S21       ROW6       .29780
5S21       ROW7       .05524
5S21       ROW8       .01033
5S22       ROW5      -1.69268
5S22       ROW6       .44275
5S22       ROW7       .08435
5S22       ROW8       .01800
5S23       ROW5      -1.62601
5S23       ROW6       .44711
5S23       ROW7       .08781
5S23       ROW8       .02129
5S24       ROW5      -1.56414
5S24       ROW6       .45152
5S24       ROW7       .09128
5S24       ROW8       .02459
5S25       ROW5      -1.52359
5S25       ROW6       .46125
5S25       ROW7       .09588
5S25       ROW8       .02823
5S26       ROW5      -1.92509
5S26       ROW6       .61509
5S26       ROW7       .13186
5S26       ROW8       .04239
5S27       ROW5      -1.83776
5S27       ROW6       .62327
5S27       ROW7       .13812
5S27       ROW8       .04828
5S28       ROW5      -1.77235
5S28       ROW6       .63708
5S28       ROW7       .14570
5S28       ROW8       .05468
5S29       ROW5      -1.68293
5S29       ROW6       .64015
5S29       ROW7       .15083
5S29       ROW8       .06015
5S30       ROW5      -1.61451
5S30       ROW6       .64877
```

```
5S30       ROW7       .15722
5S30       ROW8       .06609
5S31       ROW5      -1.56410
5S31       ROW6       .66326
5S31       ROW7       .16509
5S31       ROW8       .07269
5S32       ROW5      -1.85545
5S32       ROW6       .83465
5S32       ROW7       .21377
5S32       ROW8       .09852
5S33       ROW5      -2.12703
5S33       ROW6       1.02643
```

```
5$33      RØW7      .27163
5$33      RØW8      .13140
5$34      RØW5     -2.16051
5$34      RØW6      1.12661
5$34      RØW7      .30864
5$34      RØW8      .15647
5$35      RØW5     -2.17017
5$35      RØW6      1.22662
5$35      RØW7      .34774
5$35      RØW8      .18400
5$36      RØW5     -2.05030
5$36      RØW6      1.25659
5$36      RØW7      .36806
5$36      RØW8      .20223
5$37      RØW5     -1.94125
5$37      RØW6      1.28766
5$37      RØW7      .38873
5$37      RØW8      .22063
5$38      RØW5     -1.84153
5$38      RØW6      1.31989
5$38      RØW7      .40978
5$38      RØW8      .23922
5$39      RØW5     -1.74993
5$39      RØW6      1.35337
5$39      RØW7      .43122
5$39      RØW8      .25802
5$40      RØW5     -1.66542
5$40      RØW6      1.38819
5$40      RØW7      .45310
5$40      RØW8      .27704
5$41      RØW5     -2.35315
5$41      RØW6      2.15074
5$41      RØW7      .72168
5$41      RØW8      .45176
5$42      RØW5     -2.19504
5$42      RØW6      2.23763
5$42      RØW7      .77364
5$42      RØW8      .49603
5$43      RØW5     -2.05215
5$43      RØW6      2.33044
5$43      RØW7      .82749
```

```
5$43      RØW8      .54131
5$44      RØW5     -1.84949
5$44      RØW6      2.33320
5$44      RØW7      .84790
5$44      RØW8      .56394
$6BØUND   'MARKER'                              'SF PØRG'
6$ 1      RØW1       .73245
6$ 1      RØW5      4.00720
6$ 2      RØW1       .8
6$ 2      RØW5      3.99174
6$ 3      RØW1       .89
6$ 3      RØW5      4.04816
6$ 4      RØW1       .9
6$ 4      RØW5      3.74265
6$ 5      RØW1      1.0
6$ 5      RØW5      3.81205
6$ 6      RØW1      1.1
6$ 6      RØW5      3.83964
6$ 7      RØW1      1.1
6$ 7      RØW5      3.52757
6$ 8      RØW1      1.3
6$ 8      RØW5      3.83016
6$ 9      RØW1      1.4
6$ 9      RØW5      3.77886
6$10      RØW1      1.5
6$10      RØW5      3.71418
6$11      RØW1      1.6
6$11      RØW5      3.64020
6$12      RØW1      1.7
6$12      RØW5      3.56007
6$13      RØW1      1.8
6$13      RØW5      3.47621
6$14      RØW1      1.9
6$14      RØW5      3.39046
6$15      RØW1      2.0
6$15      RØW5      3.30421
6$16      RØW1      2.1
6$16      RØW5      3.21847
6$17      RØW1      2.2
6$17      RØW5      3.13398
6$18      RØW1      2.3
6$18      RØW5      3.05126
6$19      RØW1      2.4
6$19      RØW5      2.97069
6$20      RØW1      2.5
6$20      RØW5      2.89251
6$21      RØW1      2.5
6$21      RØW5      2.71184
6$22      RØW1      2.7
6$22      RØW5      2.71076
6$23      RØW1      2.7
6$23      RØW5      2.62592
6$24      RØW1      2.8
```

```
6$24      RØW5      2.52883
6$25      RØW1      2.8
6$25      RØW5      2.38965
6$26      RØW1      2.8
```

```
6$26      RØW5      2.26499
6$27      RØW1      3.0
6$27      RØW5      2.30245
6$28      RØW1      3.0
6$28      RØW5      2.18651
6$29      RØW1      3.2
6$29      RØW5      2.21698
6$30      RØW1      3.2
6$30      RØW5      2.10928
6$31      RØW1      3.3
6$31      RØW5      2.07295
6$32      RØW1      3.2
6$32      RØW5      1.91984
6$33      RØW1      3.0532
6$33      RØW5      1.75590
S7BØUND   'MARKER'                     'SEPØRG'
7$ 1      RØW2      1.9973
7$ 1      RØW6      1.97803
7$ 2      RØW2      2.7
7$ 2      RØW6      2.62592
7$ 3      RØW2      2.8
7$ 3      RØW6      2.52883
7$ 4      RØW2      2.8
7$ 4      RØW6      2.38965
7$ 5      RØW2      2.8
7$ 5      RØW6      2.26499
7$ 6      RØW2      3.0
7$ 6      RØW6      2.30245
7$ 7      RØW2      3.0
7$ 7      RØW6      2.18651
7$ 8      RØW2      3.2
7$ 8      RØW6      2.21698
7$ 9      RØW2      3.2
7$ 9      RØW6      2.10928
7$10      RØW2      3.3
7$10      RØW6      2.07295
7$11      RØW2      3.2
7$11      RØW6      1.91984
7$12      RØW2      3.0532
7$12      RØW6      1.75590
7$13      RØW2      2.8308
7$13      RØW6      1.56692
7$14      RØW2      3.116
7$14      RØW6      1.66182
7$15      RØW2      3.0
7$15      RØW6      1.54204
7$16      RØW2      3.0
7$16      RØW6      1.48915
7$17      RØW2      3.0
```

```
7$17      RØW6      1.43978
7$18      RØW2      3.0
7$18      RØW6      1.39358
7$19      RØW2      3.0
7$19      RØW6      1.35025
7$20      RØW2      3.0
7$20      RØW6      1.30953
7$21      RØW2      3.0
7$21      RØW6      1.27119
7$22      RØW2      3.0
7$22      RØW6      1.23505
7$23      RØW2      3.0
7$23      RØW6      1.20089
7$24      RØW2      3.0
7$24      RØW6      1.16857
7$25      RØW2      3.0
7$25      RØW6      1.13796
7$26      RØW2      3.0
7$26      RØW6      1.10890
7$27      RØW2      3.0
7$27      RØW6      1.08128
7$28      RØW2      3.0
7$28      RØW6      1.05502
7$29      RØW2      3.0
7$29      RØW6      1.03000
7$30      RØW2      3.0
7$30      RØW6      1.00613
7$31      RØW2      1.682
7$31      RØW6       .55393
S8BØUND   'MARKER'                     'SEPØRG'
8$ 1      RØW3      1.6936
8$ 1      RØW7      1.00549
8$ 2      RØW3      3.0532
8$ 2      RØW7      1.75590
8$ 3      RØW3      2.8308
8$ 3      RØW7      1.56692
8$ 4      RØW3      3.116
8$ 4      RØW7      1.66182
8$ 5      RØW3      3.0
8$ 5      RØW7      1.54204
8$ 6      RØW3      3.0
8$ 6      RØW7      1.48915
8$ 7      RØW3      3.0
8$ 7      RØW7      1.43978
8$ 8      RØW3      3.0
8$ 8      RØW7      1.39358
8$ 9      RØW3      3.0
8$ 9      RØW7      1.35025
8$10      RØW3      3.0
8$10      RØW7      1.30953
8$11      RØW3      3.0
```

```
    8S11      ROW7      1.27119
    8S12      ROW3      3.0
```

```
    8S12      ROW7      1.23505
    8S13      ROW3      3.0
    8S13      ROW7      1.20089
    8S14      ROW3      3.0
    8S14      ROW7      1.16857
    8S15      ROW3      3.0
    8S15      ROW7      1.13796
    8S16      ROW3      3.0
    8S16      ROW7      1.10890
    8S17      ROW3      3.0
    8S17      ROW7      1.08128
    8S18      ROW3      3.0
    8S18      ROW7      1.05502
    8S19      ROW3      3.0
    8S19      ROW7      1.03
    8S20      ROW3      3.0
    8S20      ROW7      1.00613
    8S21      ROW3      3.079
    8S21      ROW7      1.00885
    S9BOUND   'MARKER'                      'SEPORG'
    9S 1      ROW4      3.116
    9S 1      ROW8      1.66182
    9S 2      ROW4      3.0
    9S 2      ROW8      1.54204
    9S 3      ROW4      3.0
    9S 3      ROW8      1.48915
    9S 4      ROW4      3.0
    9S 4      ROW8      1.43978
    9S 5      ROW4      3.0
    9S 5      ROW8      1.39358
    9S 6      ROW4      3.0
    9S 6      ROW8      1.35025
    9S 7      ROW4      3.0
    9S 7      ROW8      1.30953
    9S 8      ROW4      3.0
    9S 8      ROW8      1.27119
    9S 9      ROW4      3.0
    9S 9      ROW8      1.23505
    9S10      ROW4      3.0
    9S10      ROW8      1.20089
    9S11      ROW4      3.0
    9S11      ROW8      1.16857
    9S12      ROW4      3.0
    9S12      ROW8      1.13796
    9S13      ROW4      3.0
    9S13      ROW8      1.10890
    9S14      ROW4      3.0
    9S14      ROW8      1.08128
    9S15      ROW4      3.0
    9S15      ROW8      1.05502
    9S16      ROW4      3.0
    9S16      ROW8      1.03
    9S17      ROW4      3.0
```

```
    9S17      ROW8      1.00613
    9S18      ROW4      3.079
    9S18      ROW8      1.00885
    9S19      ROW4       .497
    9S19      ROW8       .16090
    S10BOUND  'MARKER'                      'SEPORG'
    1S 1      ROW2      -.51
    1S 1      ROW9      .27814
    1S 2      ROW2      -3.116
    1S 2      ROW9      1.66182
    1S 3      ROW2      -3.0
    1S 3      ROW9      1.54204
    1S 4      ROW2      -3.0
    1S 4      ROW9      1.48915
    1S 5      ROW2      -3.0
    1S 5      ROW9      1.43978
    1S 6      ROW2      -3.0
    1S 6      ROW9      1.39358
    1S 7      ROW2      -3.0
    1S 7      ROW9      1.35025
    1S 8      ROW2      -3.0
    1S 8      ROW9      1.30953
    1S 9      ROW2      -3.0
    1S 9      ROW9      1.27119
    1S10      ROW2      -3.0
    1S10      ROW9      1.23505
    1S11      ROW2      -3.0
    1S11      ROW9      1.20089
    1S12      ROW2      -3.0
    1S12      ROW9      1.16857
    1S13      ROW2      -3.0
    1S13      ROW9      1.13796
    1S14      ROW2      -3.0
    1S14      ROW9      1.10890
    1S15      ROW2      -2.119
    1S15      ROW9      .76643
    SEPEND    'MARKER'                      'SEPEND'
RHS
    1RHS      ROW1      30.0166
    1RHS      ROW2      44.95945
    1RHS      ROW3      27.4145
    1RHS      ROW4      99.8369
```

```
     1RHS         ROW5           .0052
     1RHS         ROW6         31.60104
     1RHS         ROW7          8.46602
     1RHS         ROW8          4.13771
     1RHS         ROW9           .001
BOUNDS
 UP  BND          U1              1.
 UP  BND          U2              1.
 UP  BND          U3              1.
 UP  BND          U4              1.
 UP  BND          U5              1.
```

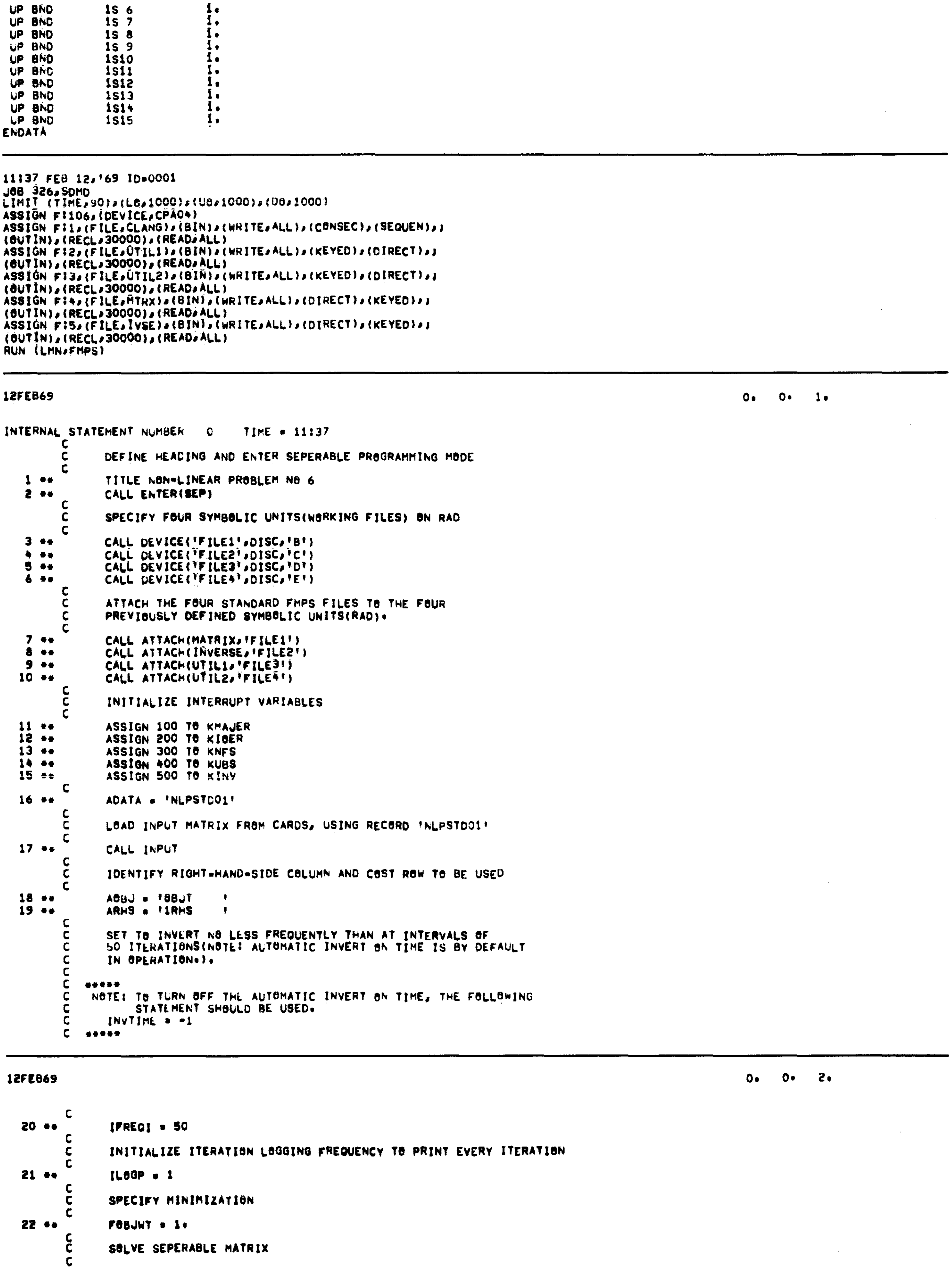
```
 UP  BND          3S 1            1.
 UP  BND          3S 2            1.
 UP  BND          3S 3            1.
 UP  BND          3S 4            1.
 UP  BND          3S 5            1.
 UP  BND          3S 6            1.
 UP  BND          3S 7            1.
 UP  BND          3S 8            1.
 UP  BND          3S 9            1.
 UP  BND          3S10            1.
 UP  BND          3S11            1.
 UP  BND          3S12            1.
 UP  BND          3S13            1.
 UP  BND          3S14            1.
 UP  BND          3S15            1.
 UP  BND          3S16            1.
 UP  BND          4S 1            1.
 UP  BND          4S 2            1.
 UP  BND          4S 3            1.
 UP  BND          4S 4            1.
 UP  BND          4S 5            1.
 UP  BND          4S 6            1.
 UP  BND          4S 7            1.
 UP  BND          4S 8            1.
 UP  BND          4S 9            1.
 UP  BND          4S10            1.
 UP  BND          4S11            1.
 UP  BND          4S12            1.
 UP  BND          4S13            1.
 UP  BND          4S14            1.
 UP  BND          4S15            1.
 UP  BND          4S16            1.
 UP  BND          5S 1            1.
 UP  BND          5S 2            1.
 UP  BND          5S 3            1.
 UP  BND          5S 4            1.
 UP  BND          5S 5            1.
 UP  BND          5S 6            1.
 UP  BND          5S 7            1.
 UP  BND          5S 8            1.
 UP  BND          5S 9            1.
 UP  BND          5S10            1.
 UP  BND          5S11            1.
 UP  BND          5S12            1.
 UP  BND          5S13            1.
 UP  BND          5S14            1.
 UP  BND          5S15            1.
 UP  BND          5S16            1.
 UP  BND          5S17            1.
 UP  BND          5S18            1.
 UP  BND          5S19            1.
 UP  BND          5S20            1.
 UP  BND          5S21            1.
```

```
 UP  BND          5S22            1.
 UP  BND          5S23            1.
 UP  BND          5S24            1.
 UP  BND          5S25            1.
 UP  BND          5S26            1.
 UP  BND          5S27            1.
 UP  BND          5S28            1.
 UP  BND          5S29            1.
 UP  BND          5S30            1.
 UP  BND          5S31            1.
 UP  BND          5S32            1.
 UP  BND          5S33            1.
 UP  BND          5S34            1.
 UP  BND          5S35            1.
 UP  BND          5S36            1.
 UP  BND          5S37            1.
 UP  BND          5S38            1.
 UP  BND          5S39            1.
 UP  BND          5S40            1.
 UP  BND          5S41            1.
 UP  BND          5S42            1.
 UP  BND          5S43            1.
 UP  BND          5S44            1.
 UP  BND          6S 1            1.
 UP  BND          6S 2            1.
 UP  BND          6S 3            1.
 UP  BND          6S 4            1.
 UP  BND          6S 5            1.
 UP  BND          6S 6            1.
 UP  BND          6S 7            1.
 UP  BND          6S 8            1.
 UP  BND          6S 9            1.
 UP  BND          6S10            1.
 UP  BND          6S11            1.
```

```
UP BND       6S12          1.
UP BND       6S13          1.
UP BND       6S14          1.
UP BND       6S15          1.
UP BND       6S16          1.
UP BND       6S17          1.
UP BND       6S18          1.
UP BND       6S19          1.
UP BND       6S20          1.
UP BND       6S21          1.
UP BND       6S22          1.
UP BND       6S23          1.
UP BND       6S24          1.
UP BND       6S25          1.
UP BND       6S26          1.
UP BND       6S27          1.
UP BND       6S28          1.
UP BND       6S29          1.
UP BND       6S30          1.
```

```
UP BND       6S31          1.
UP BND       6S32          1.
UP BND       6S33          1.
UP BND       7S 1          1.
UP BND       7S 2          1.
UP BND       7S 3          1.
UP BND       7S 4          1.
UP BND       7S 5          1.
UP BND       7S 6          1.
UP BND       7S 7          1.
UP BND       7S 8          1.
UP BND       7S 9          1.
UP BND       7S10          1.
UP BND       7S11          1.
UP BND       7S12          1.
UP BND       7S13          1.
UP BND       7S14          1.
UP BND       7S15          1.
UP BND       7S16          1.
UP BND       7S17          1.
UP BND       7S18          1.
UP BND       7S19          1.
UP BND       7S20          1.
UP BND       7S21          1.
UP BND       7S22          1.
UP BND       7S23          1.
UP BND       7S24          1.
UP BND       7S25          1.
UP BND       7S26          1.
UP BND       7S27          1.
UP BND       7S28          1.
UP BND       7S29          1.
UP BND       7S30          1.
UP BND       7S31          1.
UP BND       8S 1          1.
UP BND       8S 2          1.
UP BND       8S 3          1.
UP BND       8S 4          1.
UP BND       8S 5          1.
UP BND       8S 6          1.
UP BND       8S 7          1.
UP BND       8S 8          1.
UP BND       8S 9          1.
UP BND       8S10          1.
UP BND       8S11          1.
UP BND       8S12          1.
UP BND       8S13          1.
UP BND       8S14          1.
UP BND       8S15          1.
UP BND       8S16          1.
UP BND       8S17          1.
UP BND       8S18          1.
UP BND       8S19          1.
```

```
UP BND       8S20          1.
UP BND       8S21          1.
UP BND       9S 1          1.
UP BND       9S 2          1.
UP BND       9S 3          1.
UP BND       9S 4          1.
UP BND       9S 5          1.
UP BND       9S 6          1.
UP BND       9S 7          1.
UP BND       9S 8          1.
UP BND       9S 9          1.
UP BND       9S10          1.
UP BND       9S11          1.
UP BND       9S12          1.
UP BND       9S13          1.
UP BND       9S14          1.
UP BND       9S15          1.
UP BND       9S16          1.
UP BND       9S17          1.
UP BND       9S18          1.
UP BND       9S19          1.
UP BND       1S 1          1.
UP BND       1S 2          1.
UP BND       1S 3          1.
UP BND       1S 4          1.
UP BND       1S 5          1.
```

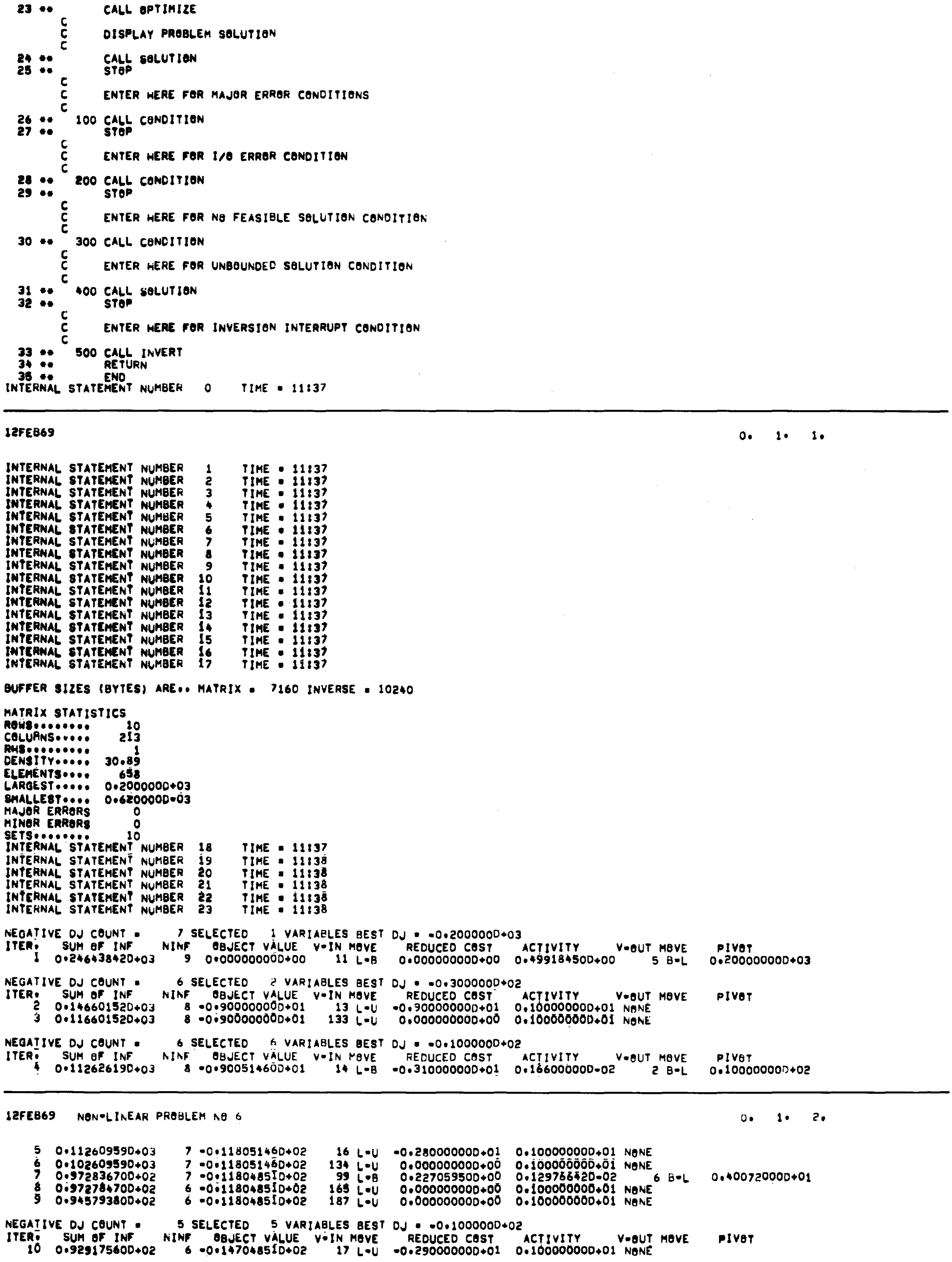


```
 UP BND        1S 6           1.
 UP BND        1S 7           1.
 UP BND        1S 8           1.
 UP BND        1S 9           1.
 UP BND        1S10           1.
 UP BND        1S11           1.
 UP BND        1S12           1.
 UP BND        1S13           1.
 UP BND        1S14           1.
 UP BND        1S15           1.
ENDATA
```

```
11:37 FEB 12,'69 ID=0001
JØB 326,SDMD
LIMIT (TIME,90),(LØ,1000),(UØ,1000),(DØ,1000)
ASSIGN F:106,(DEVICE,CPA04)
ASSIGN F:1,(FILE,CLANG),(BIN),(WRITE,ALL),(CØNSEC),(SEQUEN),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:2,(FILE,UTIL1),(BIN),(WRITE,ALL),(KEYED),(DIRECT),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:3,(FILE,UTIL2),(BIN),(WRITE,ALL),(KEYED),(DIRECT),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:4,(FILE,MTRX),(BIN),(WRITE,ALL),(DIRECT),(KEYED),;
(ØUTIN),(RECL,30000),(READ,ALL)
ASSIGN F:5,(FILE,IVSE),(BIN),(WRITE,ALL),(DIRECT),(KEYED),;
(ØUTIN),(RECL,30000),(READ,ALL)
RUN (LMN,FMPS)
```

```
12FEB69                                                                                   0.   0.   1.


INTERNAL STATEMENT NUMBER   0     TIME = 11:37
        C
        C      DEFINE HEADING AND ENTER SEPERABLE PRØGRAMMING MØDE
        C
   1 **        TITLE NØN-LINEAR PRØBLEM NØ 6
   2 **        CALL ENTER(SEP)
        C
        C      SPECIFY FØUR SYMBØLIC UNITS(WØRKING FILES) ØN RAD
        C
   3 **        CALL DEVICE('FILE1',DISC,'B')
   4 **        CALL DEVICE('FILE2',DISC,'C')
   5 **        CALL DEVICE('FILE3',DISC,'D')
   6 **        CALL DEVICE('FILE4',DISC,'E')
        C
        C      ATTACH THE FØUR STANDARD FMPS FILES TØ THE FØUR
        C      PREVIØUSLY DEFINED SYMBØLIC UNITS(RAD).
        C
   7 **        CALL ATTACH(MATRIX,'FILE1')
   8 **        CALL ATTACH(INVERSE,'FILE2')
   9 **        CALL ATTACH(UTIL1,'FILE3')
  10 **        CALL ATTACH(UTIL2,'FILE4')
        C
        C      INITIALIZE INTERRUPT VARIABLES
        C
  11 **        ASSIGN 100 TØ KMAJER
  12 **        ASSIGN 200 TØ KIØER
  13 **        ASSIGN 300 TØ KNFS
  14 **        ASSIGN 400 TØ KUBS
  15 **        ASSIGN 500 TØ KINV
        C
  16 **        ADATA = 'NLPSTD01'
        C
        C      LØAD INPUT MATRIX FRØM CARDS, USING RECØRD 'NLPSTD01'
        C
  17 **        CALL INPUT
        C
        C      IDENTIFY RIGHT-HAND-SIDE CØLUMN AND CØST RØW TØ BE USED
        C
  18 **        AØBJ = 'ØBJT    '
  19 **        ARHS = '1RHS    '
        C
        C      SET TØ INVERT NØ LESS FREQUENTLY THAN AT INTERVALS ØF
        C      50 ITERATIØNS(NØTE: AUTØMATIC INVERT ØN TIME IS BY DEFAULT
        C      IN ØPERATIØN.).
        C
        C   *****
        C    NØTE: TØ TURN ØFF THE AUTØMATIC INVERT ØN TIME, THE FØLLØWING
        C          STATEMENT SHØULD BE USED.
        C      INVTIME = -1
        C   *****
```

```
12FEB69                                                                                   0.   0.   2.


        C
  20 **        IFREQI = 50
        C
        C      INITIALIZE ITERATIØN LØGGING FREQUENCY TØ PRINT EVERY ITERATIØN
        C
  21 **        ILØGP = 1
        C
        C      SPECIFY MINIMIZATIØN
        C
  22 **        FØBJWT = 1.
        C
        C      SØLVE SEPERABLE MATRIX
        C
```



```
 23 **        CALL OPTIMIZE
       C
       C      DISPLAY PROBLEM SOLUTION
       C
 24 **        CALL SOLUTION
 25 **        STOP
       C
       C      ENTER HERE FOR MAJOR ERROR CONDITIONS
       C
 26 **    100 CALL CONDITION
 27 **        STOP
       C
       C      ENTER HERE FOR I/O ERROR CONDITION
       C
 28 **    200 CALL CONDITION
 29 **        STOP
       C
       C      ENTER HERE FOR NO FEASIBLE SOLUTION CONDITION
       C
 30 **    300 CALL CONDITION
       C
       C      ENTER HERE FOR UNBOUNDED SOLUTION CONDITION
       C
 31 **    400 CALL SOLUTION
 32 **        STOP
       C
       C      ENTER HERE FOR INVERSION INTERRUPT CONDITION
       C
 33 **    500 CALL INVERT
 34 **        RETURN
 35 **        END
INTERNAL STATEMENT NUMBER   0    TIME = 11:37
```

```
12FEB69                                                                                                        0.   1.   1.


INTERNAL STATEMENT NUMBER   1    TIME = 11:37
INTERNAL STATEMENT NUMBER   2    TIME = 11:37
INTERNAL STATEMENT NUMBER   3    TIME = 11:37
INTERNAL STATEMENT NUMBER   4    TIME = 11:37
INTERNAL STATEMENT NUMBER   5    TIME = 11:37
INTERNAL STATEMENT NUMBER   6    TIME = 11:37
INTERNAL STATEMENT NUMBER   7    TIME = 11:37
INTERNAL STATEMENT NUMBER   8    TIME = 11:37
INTERNAL STATEMENT NUMBER   9    TIME = 11:37
INTERNAL STATEMENT NUMBER  10    TIME = 11:37
INTERNAL STATEMENT NUMBER  11    TIME = 11:37
INTERNAL STATEMENT NUMBER  12    TIME = 11:37
INTERNAL STATEMENT NUMBER  13    TIME = 11:37
INTERNAL STATEMENT NUMBER  14    TIME = 11:37
INTERNAL STATEMENT NUMBER  15    TIME = 11:37
INTERNAL STATEMENT NUMBER  16    TIME = 11:37
INTERNAL STATEMENT NUMBER  17    TIME = 11:37

BUFFER SIZES (BYTES) ARE.. MATRIX =  7160 INVERSE = 10240

MATRIX STATISTICS
ROWS.........     10
COLUMNS......    213
RHS..........      1
DENSITY......  30.89
ELEMENTS.....    658
LARGEST......  0.200000D+03
SMALLEST.....  0.620000D-03
MAJOR ERRORS       0
MINOR ERRORS       0
SETS.........     10
INTERNAL STATEMENT NUMBER  18    TIME = 11:37
INTERNAL STATEMENT NUMBER  19    TIME = 11:38
INTERNAL STATEMENT NUMBER  20    TIME = 11:38
INTERNAL STATEMENT NUMBER  21    TIME = 11:38
INTERNAL STATEMENT NUMBER  22    TIME = 11:38
INTERNAL STATEMENT NUMBER  23    TIME = 11:38

NEGATIVE DJ COUNT =      7 SELECTED   1 VARIABLES BEST DJ = -0.200000D+03
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
    1  0.24643842D+03    9  0.00000000D+00    11 L-B   0.00000000D+00  0.49918450D+00       5 B-L   0.20000000D+03

NEGATIVE DJ COUNT =      6 SELECTED   2 VARIABLES BEST DJ = -0.300000D+02
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
    2  0.14660152D+03    8 -0.90000000D+01    13 L-U  -0.90000000D+01  0.10000000D+01 NONE
    3  0.11660152D+03    8 -0.90000000D+01   133 L-U   0.00000000D+00  0.10000000D+01 NONE

NEGATIVE DJ COUNT =      6 SELECTED   6 VARIABLES BEST DJ = -0.100000D+02
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
    4  0.11262619D+03    8 -0.90051460D+01    14 L-B  -0.31000000D+01  0.16600000D-02       2 B-L   0.10000000D+02
```

```
12FEB69   NON-LINEAR PROBLEM NO 6                                                                      0.   1.   2.


    5  0.11260959D+03    7 -0.11805146D+02    16 L-U  -0.28000000D+01  0.10000000D+01 NONE
    6  0.10260959D+03    7 -0.11805146D+02   134 L-U   0.00000000D+00  0.10000000D+01 NONE
    7  0.97283670D+02    7 -0.11804851D+02    99 L-B   0.22705950D+00  0.12976642D-02       6 B-L   0.40072000D+01
    8  0.97278470D+02    6 -0.11804851D+02   165 L-U   0.00000000D+00  0.10000000D+01 NONE
    9  0.94579380D+02    6 -0.11804851D+02   187 L-U   0.00000000D+00  0.10000000D+01 NONE

NEGATIVE DJ COUNT =      5 SELECTED   5 VARIABLES BEST DJ = -0.100000D+02
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   10  0.92917560D+02    6 -0.14704851D+02    17 L-U  -0.29000000D+01  0.10000000D+01 NONE
   11  0.82917560D+02     6 -0.14704851D+02   135 L-U   0.00000000D+00  0.10000000D+01 NONE
   12  0.77588730D+02     6 -0.14704851D+02   166 L-U   0.00000000D+00  0.10000000D+01 NONE
   13  0.72779630D+02     6 -0.14704851D+02   188 L-U   0.00000000D+00  0.10000000D+01 NONE
   14  0.71237590D+02     6 -0.14700000D+02    54 L-B   0.11618327D+00  0.41756037D-01     14 B-L    0.37478475D-01

NEGATIVE DJ COUNT =     6 SELECTED   6 VARIABLES BEST DJ = -0.800000D+02
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   15  0.71229899D+02     6 -0.19938645D+02    18 L-B  -0.24000000D+02  0.21827687D+00      3 B-L    0.80000000D+02
   16  0.53767749D+02     5 -0.19830904D+02    13 U-B   0.90000000D+01  0.11971174D-01     54 B-U    0.80045946D+02
   17  0.53591250D+02     5 -0.19830904D+02   167 L-U   0.00000000D+00  0.10000000D+01 NONE
   18  0.49193530D+02     5 -0.18990904D+02   136 L-U   0.84000000D+00  0.10000000D+01 NONE
   19  0.46803880D+02     5 -0.18990904D+02   189 L-B   0.00000000D+00  0.63733674D+00      9 B-L    0.14891500D+01

NEGATIVE DJ COUNT =     4 SELECTED   4 VARIABLES BEST DJ = -0.477782D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   20  0.45854790D+02     4 -0.18990904D+02   168 L-U   0.00000000D+00  0.10000000D+01 NONE
   21  0.41076970D+02     4 -0.18150904D+02   137 L-U   0.84000000D+00  0.10000000D+01 NONE
   22  0.38811980D+02     4 -0.18043890D+02    55 L-B   0.14798663D+00  0.72313581D+00     99 B-U   -0.67347774D+00
   23  0.38609466D+02     4 -0.18044440D+02   207 L-B  -0.15300000D+00  0.35953117D-02     10 B-L    0.27814000D+00

NEGATIVE DJ COUNT =     5 SELECTED   5 VARIABLES BEST DJ = -0.454204D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVF     PIVOT
   24  0.38608466D+02     3 -0.18044440D+02   169 L-U   0.00000000D+00  0.10000000D+01 NONE
   25  0.34066426D+02     3 -0.17144440D+02   138 L-U   0.90000000D+00  0.10000000D+01 NONE
   26  0.31763976D+02     3 -0.17099516D+02   100 L-B   0.24000000D+00  0.18718404D+00     55 B-U   -0.14791015D+01

NEGATIVE DJ COUNT =     3 SELECTED   3 VARIABLES BEST DJ = -0.448915D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   27  0.31686440D+02     3 -0.17099516D+02   170 L-B   0.00000000D+00  0.60056408D+00      8 B-L    0.14891500D+01
   28  0.28990418D+02     2 -0.16199516D+02   139 L-U   0.90000000D+00  0.10000000D+01 NONE
   29  0.26803908D+02     2 -0.16046835D+02    56 L-U   0.15268109D+00  0.10000000D+01 NONE

NEGATIVE DJ COUNT =     5 SELECTED   5 VARIABLES BEST DJ = -0.221698D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   30  0.26596092D+02     2 -0.15086835D+02   140 L-U   0.96000000D+00  0.10000000D+01 NONE
   31  0.24379112D+02     2 -0.15252059D+02    37 L-B  -0.30245146D+00  0.54628243D+00    207 B-U   -0.18239735D+01
   32  0.24097933D+02     2 -0.15226327D+02    57 L-B   0.14416209D+00  0.17849308D+00    100 B-U   -0.60067539D+00
SEP VAR.  169 REJECTED
SEP VAR.  171 REJECTED

NEGATIVE DJ COUNT =     5 SELECTED   5 VARIABLES BEST DJ = -0.210928D+01
```

```
12FEB69   NON-LINEAR PROBLEM NO 6                                                                   0.   1.   3.

ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   33  0.24061077D+02     2 -0.14266327D+02   141 L-U   0.96000000D+00  0.10000000D+01 NONE
   34  0.21951797D+02     2 -0.14415506D+02   208 L-B  -0.11113125D+01  0.13423722D+00    189 B-U   -0.11080446D+01
   35  0.21744679D+02     2 -0.14261510D+02   101 L-B   0.28631922D+00  0.53784929D+00     57 B-U   -0.17003717D+01

NEGATIVE DJ COUNT =     3 SELECTED   3 VARIABLES BEST DJ = -0.207295D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   36  0.21571605D+02     2 -0.14101639D+02   142 L-B   0.99000000D+00  0.16148534D+00     18 B-L    0.41250000D-01
   37  0.21236854D+02     2 -0.14100146D+02   190 L-B   0.94961973D-01  0.15724625D-01     37 B-U   -0.28380115D+01
   38  0.21182182D+02     2 -0.13978245D+02    58 L-B   0.15229149D+00  0.80044522D+00    101 B-U   -0.57038012D+00

NEGATIVE DJ COUNT =     7 SELECTED   7 VARIABLES BEST DJ = -0.628167D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   39  0.21015125D+02     2 -0.13190422D+02    17 U-B   0.29000000D+01  0.27166314D+00    142 B-U    0.30303030D+01
   40  0.19308628D+02     2 -0.13294060D+02    38 L-B  -0.32571218D+00  0.31819018D+00    170 B-U   -0.34131159D+00
   41  0.19160983D+02     2 -0.13253702D+02   102 L-B   0.24045859D+00  0.16784097D+00     58 B-U   -0.16011816D+01

NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.191984D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   42  0.19097140D+02     2 -0.12325702D+02   143 L-U   0.92800000D+00  0.10000000D+01 NONE
   43  0.17177300D+02     2 -0.12187235D+02    59 L-U   0.13846627D+00  0.10000000D+01 NONE

NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.175590D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   44  0.16943570D+02     2 -0.11301807D+02   144 L-U   0.88542800D+00  0.10000000D+01 NONE
   45  0.15187670D+02     2 -0.11243659D+02    60 L-B   0.13016288D+00  0.44673239D+00    102 B-U   -0.55639988D+00

NEGATIVE DJ COUNT =     4 SELECTED   4 VARIABLES BEST DJ = -0.156692D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   46  0.15083166D+02     2 -0.10808245D+02   145 L-B   0.82093200D+00  0.53039018D+00     17 B-L    0.28308000D+00
   47  0.14252087D+02     2 -0.10717575D+02   103 L-B   0.30000000D+00  0.30223371D+00     60 B-U   -0.18305953D+01

NEGATIVE DJ COUNT =     6 SELECTED   6 VARIABLES BEST DJ = -0.553526D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   48  0.14101472D+02     2 -0.10356069D+02    16 U-B   0.28000000D+01  0.12910909D+00    145 B-U    0.35325703D+01
   49  0.13386820D+02     2 -0.10219771D+02    61 L-U   0.13629826D+00  0.10000000D+01 NONE

NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.166182D+01
ITER.   SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY     V-OUT MOVE     PIVOT
   50  0.13152680D+02     2 -0.93472912D+01   146 L-U   0.87248000D+00  0.10000000D+01 NONE
INTERNAL STATEMENT NUMBER  33    TIME = 11:38

   7 NON-BASIC SLACKS. COMPLETELY TRIANGULARIZED     1 ROWS AND     9 COLS.
   0 IN NON-COMPLETELY TRIANGULARIZED PART. OF THESE     0 WHERE NOT TRIANGULARIZED AND     0 WERE REJECTED FOR TOO SMALL A PIVOT.
MATRIX TO BE INVERTED HAD    10 COLS AND      21 ELEMENTS. INVERSE HAS    7 COLS AND      18 ELEMENTS.

 1200 MS FOR INVERT
INTERNAL STATEMENT NUMBER  34    TIME = 11:38
INTERNAL STATEMENT NUMBER  23    TIME = 11:38
```

```
12FEB69   NON-LINEAR PROBLEM NO 6                                                                   0.   1.   4.


NEGATIVE DJ COUNT =      3 SELECTED   3 VARIABLES BEST DJ = -0.156022D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   51  0.11490860D+02     2 -0.94797587D+01   171 L-B  -0.78850702D+00  0.16799789D+00    38 B-U  -0.28676307D+01
   52  0.11228746D+02     2 -0.89285761D+01   147 L-B   0.84000000D+00  0.65616975D+00     7 B-L   0.15420400D+01

NEGATIVE DJ COUNT =      4 SELECTED   4 VARIABLES BEST DJ = -0.103422D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   53  0.10216906D+02     1 -0.89308365D+01    39 L-B  -0.26208076D-02  0.86247608D+00   103 B-U  -0.13020553D+00
   54  0.93249159D+01     1 -0.89315055D+01    61 U-B  -0.19185388D-01  0.34870475D-01    39 B-U   0.39438501D+01
SEP VAR.  170 REJECTED
SEP VAR.  172 REJECTED

NEGATIVE DJ COUNT =      2 SELECTED   2 VARIABLES BEST DJ = -0.103868D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   55  0.91803303D+01     1 -0.89317706D+01    40 L-B  -0.60144084D-02  0.44072817D-01   147 B-U  -0.36130351D+00

NEGATIVE DJ COUNT =      4 SELECTED   4 VARIABLES BEST DJ = -0.277622D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   56  0.91345526D+01     1 -0.89284077D+01   148 L-B   0.12735461D-01  0.26405516D+00   190 B-U  -0.90516706D+00

NEGATIVE DJ COUNT =      2 SELECTED   2 VARIABLES BEST DJ = -0.296866D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   57  0.84014782D+01     1 -0.89277712D+01   191 L-B   0.13618262D-01  0.46740380D-01   208 B-U  -0.84336886D+00
   58  0.82627220D+01     1 -0.89305175D+01   103 U-B  -0.71899480D-02  0.38196246D+00    61 B-L  -0.19473776D+01

NEGATIVE DJ COUNT =      2 SELECTED   2 VARIABLES BEST DJ = -0.321307D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   59  0.82124751D+01     1 -0.89309813D+01   209 L-B  -0.12514221D-01  0.37067780D-01    40 B-U  -0.31446459D+01
   60  0.80933736D+01     1 -0.89430260D+01    60 U-L  -0.12044714D-01  0.10000000D+01 NONE

NEGATIVE DJ COUNT =      2 SELECTED   2 VARIABLES BEST DJ = -0.101109D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   61  0.80289262D+01     1 -0.89445705D+01    41 L-B  -0.50698032D-02  0.30464709D+00   171 B-U  -0.33703066D+00

NEGATIVE DJ COUNT =      0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00
INTERNAL STATEMENT NUMBER  33     TIME = 11:38

   8 NON-BASIC SLACKS. COMPLETELY TRIANGULARIZED     1 ROWS AND     9 COLS.
   0 IN NON-COMPLETELY TRIANGULARIZED PART. OF THESE     0 WHERE NOT TRIANGULARIZED AND     0 WERE REJECTED FOR TOO SMALL A PIVOT.
MATRIX TO BE INVERTED HAD    10 COLS AND      22 ELEMENTS. INVERSE HAS     8 COLS AND      20 ELEMENTS.

  600 MS FOR INVERT
INTERNAL STATEMENT NUMBER  34     TIME = 11:38
INTERNAL STATEMENT NUMBER  23     TIME = 11:38

NEGATIVE DJ COUNT =      1 SELECTED   1 VARIABLES BEST DJ = -0.300000D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   62  0.77209000D+01     1 -0.89480958D+01   172 L-B  -0.14559869D-01  0.24212460D+00    41 B-U  -0.28718805D+01
```

```
12FEB69   NON-LINEAR PROBLEM NO 6                                                                   0.   1.   5.


NEGATIVE DJ COUNT =      4 SELECTED   4 VARIABLES BEST DJ = -0.128957D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   63  0.69945262D+01     1 -0.89483064D+01    42 L-B  -0.79020578D-02  0.26642628D-01   148 B-U  -0.40226975D+00
SEP VAR.  171 REJECTED
SEP VAR.  173 REJECTED

NEGATIVE DJ COUNT =      2 SELECTED   2 VARIABLES BEST DJ = -0.309946D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   64  0.69601686D+01     1 -0.89447198D+01   149 L-B   0.88562092D-02  0.40497854D+00    42 B-U  -0.24034789D+01

NEGATIVE DJ COUNT =      2 SELECTED   2 VARIABLES BEST DJ = -0.127200D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   65  0.57049556D+01     1 -0.89440710D+01    43 L-B   0.18445854D-02  0.35171270D+00   191 B-U  -0.42400149D+00
   66  0.52575753D+01     1 -0.89564202D+01    59 U-B  -0.15985190D-01  0.77253447D+00   103 B-L  -0.56156661D+00

NEGATIVE DJ COUNT =      2 SELECTED   2 VARIABLES BEST DJ = -0.296576D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   67  0.51852656D+01     1 -0.89572056D+01   192 L-B  -0.58674346D-02  0.13386169D+00   209 B-U  -0.88136453D+00
   68  0.47882639D+01     1 -0.89574776D+01   102 U-B  -0.33028602D-02  0.82348069D-01    59 B-L  -0.17369300D+01

NEGATIVE DJ COUNT =      2 SELECTED   2 VARIABLES BEST DJ = -0.320574D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   69  0.47791959D+01     1 -0.89580945D+01   210 L-B  -0.33925645D-01  0.18184871D-01   172 B-U  -0.10685788D+01

NEGATIVE DJ COUNT =      0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00
INTERNAL STATEMENT NUMBER  33     TIME = 11:38

   8 NON-BASIC SLACKS. COMPLETELY TRIANGULARIZED     1 ROWS AND     9 COLS.
   0 IN NON-COMPLETELY TRIANGULARIZED PART. OF THESE     0 WHERE NOT TRIANGULARIZED AND     0 WERE REJECTED FOR TOO SMALL A PIVOT.
MATRIX TO BE INVERTED HAD    10 COLS AND      22 ELEMENTS. INVERSE HAS     8 COLS AND      20 ELEMENTS.

  600 MS FOR INVERT
INTERNAL STATEMENT NUMBER  34     TIME = 11:38
INTERNAL STATEMENT NUMBER  23     TIME = 11:38

NEGATIVE DJ COUNT =      1 SELECTED   1 VARIABLES BEST DJ = -0.300000D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   70  0.47209000D+01     1 -0.89618127D+01   173 L-B  -0.30761241D-01  0.12087391D+00    43 B-U  -0.22851510D+01

NEGATIVE DJ COUNT =      4 SELECTED   4 VARIABLES BEST DJ = -0.129523D+01
ITER.    SUM OF INF     NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY       V-OUT MOVE     PIVOT
   71  0.43582783D+01     1 -0.89618813D+01    44 L-R  -0.15141831D-01  0.45286126D-02   149 B-U  -0.40489519D+00
SEP VAR.  172 REJECTED
SEP VAR.  174 REJECTED
```

```
NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.309627D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   72  0.43524127D+01    1 -0.89626458D+01    150 L-B  -0.92427965D-02  0.82715022D-01    102 B-U  -0.37235114D+00
________________________________________________________________________________________________________________

12FEB69   NON-LINEAR PROBLEM NO 6                                                                    0.    1.    6.


NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.846965D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   73  0.40963046D+01    1 -0.89626812D+01    103 L-B  -0.28969843D-03  0.12199473D+00     44 B-U  -0.65391279D+01
   74  0.30630515D+01    1 -0.89669867D+01     58 U-B  -0.21377213D-01  0.20140841D+00    103 B-L  -0.60570821D+00

NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.129379D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   75  0.30503248D+01    1 -0.89720433D+01     45 L-B  -0.71962478D-02  0.70267649D+00    192 B-U  -0.42310593D+00
   76  0.21412061D+01    1 -0.89760298D+01    102 U-B  -0.10434214D-01  0.38205550D+00     58 B-L  -0.16320572D+01

NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.290953D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   77  0.20799446D+01    1 -0.89773365D+01    193 L-B  -0.12844359D-01  0.10173116D+00    150 B-U  -0.93968771D+00

NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.300000D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   78  0.17839550D+01    1 -0.89770698D+01    151 L-B   0.12874013D-01  0.20714682D-01     45 B-U  -0.23473219D+01

NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.151165D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   79  0.17218109D+01    1 -0.89770673D+01     46 L-B   0.40787216D-02  0.60261328D-03    173 B-U  -0.50388447D+00

NEGATIVE DJ COUNT =     0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00
INTERNAL STATEMENT NUMBER  33    TIME = 11:38

   8 NON-BASIC SLACKS. COMPLETELY TRIANGULARIZED    1 ROWS AND    9 COLS.
   0 IN NON-COMPLETELY TRIANGULARIZED PART. OF THESE    0 WHERE NOT TRIANGULARIZED AND    0 WERE REJECTED FOR TOO SMALL A PIVOT.
MATRIX TO BE INVERTED HAD   10 COLS AND     22 ELEMENTS. INVERSE HAS   8 COLS AND     20 ELEMENTS.

  600 MS FOR INVERT
INTERNAL STATEMENT NUMBER  34    TIME = 11:38
INTERNAL STATEMENT NUMBER  23    TIME = 11:38

NEGATIVE DJ COUNT =     1 SELECTED   1 VARIABLES BEST DJ = -0.300000D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   80  0.17209000D+01    1 -0.89762265D+01    174 L-B   0.78504466D-02  0.10710713D+00    210 B-U  -0.87938085D+00

NEGATIVE DJ COUNT =     4 SELECTED   4 VARIABLES BEST DJ = -0.329839D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   81  0.13995786D+01    1 -0.89834301D+01    211 L-B  -0.19217362D-01  0.37484894D+00     46 B-U  -0.21161721D+01
SEP VAR.  173 REJECTED
SEP VAR.  175 REJECTED
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   82  0.16318082D+00    1 -0.89947768D+01     57 U-L  -0.11346689D-01  0.10000000D+01 NONE

NEGATIVE DJ COUNT =     2 SELECTED   2 VARIABLES BEST DJ = -0.153463D+01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   83  0.10254074D+00    1 -0.89955497D+01     47 L-B  -0.11567315D-01  0.66813026D-01      4 B-L   0.15346269D+01
SOLUTION FEASIBLE AT ITERATION     83
________________________________________________________________________________________________________________

12FEB69   NON-LINEAR PROBLEM NO 6                                                                    0.    1.    7.


NEGATIVE DJ COUNT =     8 SELECTED   8 VARIABLES BEST DJ = -0.100000D+00
SEP VAR.   14 REJECTED
SEP VAR.   17 REJECTED
SEP VAR.  210 REJECTED
SEP VAR.  212 REJECTED
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   84  0.00000000D+00    0 -0.90024753D+01     56 U-B  -0.22163360D-01  0.31248004D+00    102 B-L  -0.68518830D+00
   85  0.00000000D+00    0 -0.90054897D+01     20 L-B  -0.10240052D-01  0.29437382D+00    151 B-U  -0.66539230D+00
SEP VAR.   46 REJECTED
SEP VAR.   48 REJECTED

NEGATIVE DJ COUNT =     4 SELECTED   4 VARIABLES BEST DJ = -0.113348D-01
SEP VAR.   55 REJECTED
SEP VAR.   57 REJECTED
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   86  0.00000000D+00    0 -0.90078467D+01    101 U-B  -0.57516848D-02  0.40979606D+00     56 B-L  -0.14470169D+01
SEP VAR.   21 REJECTED

NEGATIVE DJ COUNT =     0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00
INTERNAL STATEMENT NUMBER  33    TIME = 11:38

   9 NON-BASIC SLACKS. COMPLETELY TRIANGULARIZED    0 ROWS AND    7 COLS.
   3 IN NON-COMPLETELY TRIANGULARIZED PART. OF THESE    1 WHERE NOT TRIANGULARIZED AND    0 WERE REJECTED FOR TOO SMALL A PIVOT.
MATRIX TO BE INVERTED HAC   10 COLS AND     28 ELEMENTS. INVERSE HAS   11 COLS AND     28 ELEMENTS.

 1200 MS FOR INVERT
INTERNAL STATEMENT NUMBER  34    TIME = 11:38
INTERNAL STATEMENT NUMBER  23    TIME = 11:38

NEGATIVE DJ COUNT =     8 SELECTED   8 VARIABLES BEST DJ = -0.100000D+00
SEP VAR.   14 REJECTED
SEP VAR.   17 REJECTED
SEP VAR.  210 REJECTED
SEP VAR.  212 REJECTED
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
   87  0.00000000D+00    0 -0.90147912D+01     55 U-B  -0.15533716D-01  0.44706210D+00     20 B-L  -0.28292903D+00
SEP VAR.   46 REJECTED
SEP VAR.   48 REJECTED
```

```
NEGATIVE DJ COUNT =     5 SELECTED   5 VARIABLES BEST DJ = -0.750330D-01
ITER.   SUM OF INF    NINF   OBJECT VALUE  V-IN MOVE    REDUCED COST    ACTIVITY      V-OUT MOVE    PIVOT
  88  0.000000000D+00    0 -0.90200293D+01   152 L-B  -0.75032995D-01  0.69809508D-01   101 B-L   0.37793404D+01
SEP VAR.   56 REJECTED
SEP VAR.   54 REJECTED

NEGATIVE DJ COUNT =     0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00
INTERNAL STATEMENT NUMBER  33     TIME = 11:38
```

```
12FEB69    NON-LINEAR PROBLEM NO 6                                                                   0.   1.   8.


   9 NON-BASIC SLACKS. COMPLETELY TRIANGULARIZED     1 ROWS AND     7 COLS.
   2 IN NON-COMPLETELY TRIANGULARIZED PART. OF THESE     1 WHERE NOT TRIANGULARIZED AND     0 WERE REJECTED FOR TOO SMALL A PIVOT.
MATRIX TO BE INVERTED HAD    10 COLS AND      25 ELEMENTS. INVERSE HAS   11 COLS AND      26 ELEMENTS.

 1200 MS FOR INVERT
INTERNAL STATEMENT NUMBER  34     TIME = 11:38
INTERNAL STATEMENT NUMBER  23     TIME = 11:38

NEGATIVE DJ COUNT =     8 SELECTED   8 VARIABLES BEST DJ = -0.100000D+00
SEP VAR.   14 REJECTED
SEP VAR.   17 REJECTED
SEP VAR.  210 REJECTED
SEP VAR.  212 REJECTED
SEP VAR.   56 REJECTED
SEP VAR.   54 REJECTED
SEP VAR.   46 REJECTED
SEP VAR.   48 REJECTED

NEGATIVE DJ COUNT =     0 SELECTED   0 VARIABLES BEST DJ =  0.000000D+00
LOCAL OPTIMUM ENCOUNTERED

OPTIMAL SOLUTION. OBJECTIVE VALUE =-0.90200293D+01
INTERNAL STATEMENT NUMBER  24     TIME = 11:39
```

```
12FEB69    NON-LINEAR PROBLEM NO 6                                                                   0.   2.   1.


IDENTIFIER SECTION

          PROBLEM... NAME..
                     MODE.. SEP
                     CLASS. SEP
                     STATUS OPTIMAL.
          FUNCTIONAL NAME.. OBJT
                     OBJECT MINIMIZE
                     VALUE.      -9.020030
          RESTRAINT. NAME.. 1RHS
          ITERATION. COUNT.        88
```

```
12FEB69    NON-LINEAR PROBLEM NO 6                                                                   0.   2.   2.


SECTION 1 - ROWS                            PRIMAL-DUAL OUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  SLACK ACTIVITY  ..LOWER LIMIT.  ..UPPER LIMIT.  .DUAL ACTIVITY  ..INPUT COST..  .REDUCED COST.
     1  OBJT     FR      -9.020030       9.020029            NONE            NONE        1.000000        1.000000        0.000000
     2  ROW1     EQ      30.016586       0.000000       30.016586       30.016586        0.300000        0.000000        0.300000
     3  ROW2     EQ      44.959442       0.000000       44.959442       44.959442        0.280000        0.000000        0.280000
     4  ROW3     EQ      27.414490       0.000000       27.414490       27.414490        0.003934        0.000000        0.003934
     5  ROW4     EQ      99.836899       0.000000       99.836899       99.836899        0.000000        0.000000        0.000000
     6  ROW5     EQ       0.005200       0.000000        0.005200        0.005200       -0.061052        0.000000       -0.061052
     7  ROW6     EQ      31.601028       0.000000       31.601028       31.601028       -0.641452        0.000000       -0.641452
     8  ROW7     EQ       8.466020       0.000000        8.466020        8.466020       -0.009013        0.000000       -0.009013
     9  ROW8     EQ       4.137710       0.000000        4.137710        4.137710        0.000000        0.000000        0.000000
    10  ROW9     EQ       0.001000       0.000000        0.001000        0.001000        0.583422        0.000000        0.583422
```

```
12FEB69    NON-LINEAR PROBLEM NO 6                                                                   0.   2.   3.


SECTION 2 - COLUMNS                         PRIMAL-DUAL OUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  ..INPUT COST..  ..LOWER LIMIT.  ..UPPER LIMIT.  .REDUCED COST.
    11  X5       BS       0.514710        0.000000        0.000000            NONE        0.000000
    12  UBOUND1  EQ       0.000000        0.000000        0.000000        0.000000        0.000000
    13  U1       BS       0.949472       -9.000000        0.000000        1.000000        0.000000
    14  U2       LL       0.000000       -3.100000        0.000000        1.000000       -0.100000
    15  UBOUND2  EQ       0.000000        0.000000        0.000000        0.000000        0.000000
    16  U3       BS       0.169566       -2.800000        0.000000        1.000000        0.000000
    17  U4       LL       0.000000       -2.900001        0.000000        1.000000       -0.100000
    18  U5       LL       0.000000      -24.000000        0.000000        1.000000       -1.600000
    19  S3BOUND  EQ       0.000000        0.000000        0.000000        0.000000        0.000000
    20  3S 1     LL       0.000000        0.000000        0.000000        1.000000        0.003676
    21  3S 2     LL       0.000000        0.000000        0.000000        1.000000        0.002288
    22  3S 3     LL       0.000000        0.000000        0.000000        1.000000        0.000837
    23  3S 4     LL       0.000000        0.000000        0.000000        1.000000       -0.000600
    24  3S 5     LL       0.000000        0.000000        0.000000        1.000000       -0.002019
    25  3S 6     LL       0.000000        0.000000        0.000000        1.000000       -0.004496
    26  3S 7     LL       0.000000        0.000000        0.000000        1.000000       -0.006622
    27  3S 8     LL       0.000000        0.000000        0.000000        1.000000       -0.008797
    28  3S 9     LL       0.000000        0.000000        0.000000        1.000000       -0.010907
```

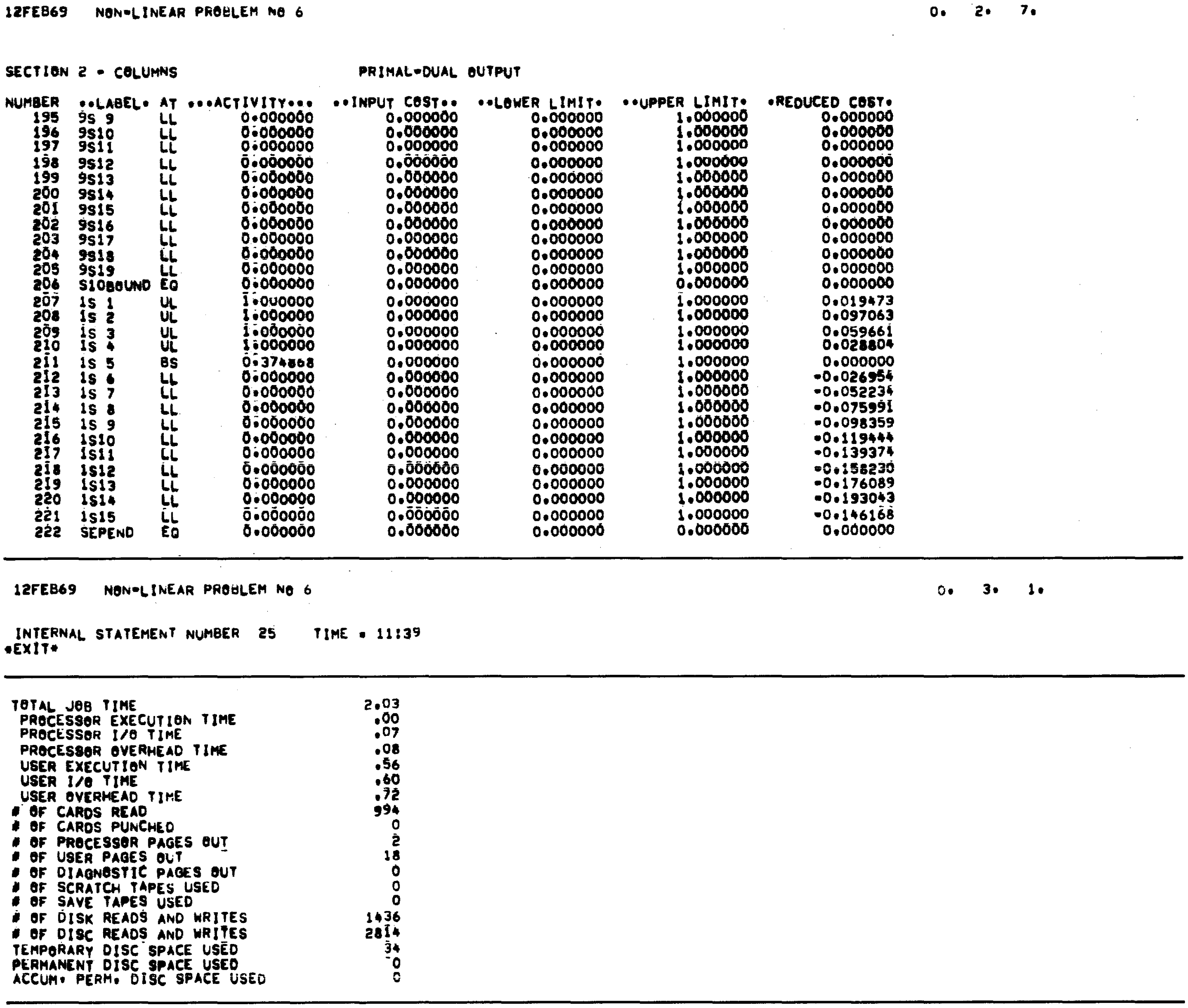
```
      29  3S10     LL        0.000000       0.000000       0.000000       1.000000      -0.015836
      30  3S11     LL        0.000000       0.000000       0.000000       1.000000      -0.018794
      31  3S12     LL        0.000000       0.000000       0.000000       1.000000      -0.021711
      32  3S13     LL        0.000000       0.000000       0.000000       1.000000      -0.024588
      33  3S14     LL        0.000000       0.000000       0.000000       1.000000      -0.027429
      34  3S15     LL        0.000000       0.000000       0.000000       1.000000      -0.025002
      35  3S16     LL        0.000000       0.000000       0.000000       1.000000      -0.026929
      36  S4BOUND  EQ        0.000000       0.000000       0.000000       0.000000       0.000000
      37  4S 1     UL        1.000000       0.000000       0.000000       1.000000       0.014691
      38  4S 2     UL        1.000000       0.000000       0.000000       1.000000       0.013432
      39  4S 3     UL        1.000000       0.000000       0.000000       1.000000       0.012110
      40  4S 4     UL        1.000000       0.000000       0.000000       1.000000       0.010755
      41  4S 5     UL        1.000000       0.000000       0.000000       1.000000       0.009511
      42  4S 6     UL        1.000000       0.000000       0.000000       1.000000       0.010098
      43  4S 7     UL        1.000000       0.000000       0.000000       1.000000       0.008171
      44  4S 8     UL        1.000000       0.000000       0.000000       1.000000       0.006200
      45  4S 9     UL        1.000000       0.000000       0.000000       1.000000       0.004291
      46  4S10     UL        1.000000       0.000000       0.000000       1.000000       0.002667
      47  4S11     BS        0.000041       0.000000       0.000000       1.000000       0.000000
      48  4S12     LL        0.000000       0.000000       0.000000       1.000000      -0.002627
      49  4S13     LL        0.000000       0.000000       0.000000       1.000000      -0.005215
      50  4S14     LL        0.000000       0.000000       0.000000       1.000000      -0.007766
      51  4S15     LL        0.000000       0.000000       0.000000       1.000000      -0.008395
      52  4S16     LL        0.000000       0.000000       0.000000       1.000000      -0.010121
      53  S5BOUND  EQ        0.000000       0.000000       0.000000       0.000000       0.000000
      54  5S 1     UL        1.000000       0.000000       0.000000       1.000000       0.007376
      55  5S 2     BS        0.160604       0.000000       0.000000       1.000000       0.000000
      56  5S 3     LL        0.000000       0.000000       0.000000       1.000000      -0.010786
```

```
12FEB69   NON-LINEAR PROBLEM NO 6                                                                        0.   2.   4.


SECTION 2 - COLUMNS                         PRIMAL-DUAL OUTPUT

NUMBER   ..LABEL. AT ...ACTIVITY...   ..INPUT COST..   ..LOWER LIMIT.   ..UPPER LIMIT.   .REDUCED COST.
      57  5S 4     LL        0.000000       0.000000       0.000000       1.000000      -0.020508
      58  5S 5     LL        0.000000       0.000000       0.000000       1.000000      -0.029915
      59  5S 6     LL        0.000000       0.000000       0.000000       1.000000      -0.037532
      60  5S 7     LL        0.000000       0.000000       0.000000       1.000000      -0.044928
      61  5S 8     LL        0.000000       0.000000       0.000000       1.000000      -0.051767
      62  5S 9     LL        0.000000       0.000000       0.000000       1.000000      -0.058118
      63  5S10     LL        0.000000       0.000000       0.000000       1.000000      -0.064042
      64  5S11     LL        0.000000       0.000000       0.000000       1.000000      -0.070844
      65  5S12     LL        0.000000       0.000000       0.000000       1.000000      -0.074910
      66  5S13     LL        0.000000       0.000000       0.000000       1.000000      -0.079837
      67  5S14     LL        0.000000       0.000000       0.000000       1.000000      -0.084508
      68  5S15     LL        0.000000       0.000000       0.000000       1.000000      -0.088946
      69  5S16     LL        0.000000       0.000000       0.000000       1.000000      -0.093176
      70  5S17     LL        0.000000       0.000000       0.000000       1.000000      -0.098929
      71  5S18     LL        0.000000       0.000000       0.000000       1.000000      -0.153149
      72  5S19     LL        0.000000       0.000000       0.000000       1.000000      -0.161372
      73  5S20     LL        0.000000       0.000000       0.000000       1.000000      -0.169165
      74  5S21     LL        0.000000       0.000000       0.000000       1.000000      -0.118959
      75  5S22     LL        0.000000       0.000000       0.000000       1.000000      -0.181422
      76  5S23     LL        0.000000       0.000000       0.000000       1.000000      -0.188320
      77  5S24     LL        0.000000       0.000000       0.000000       1.000000      -0.194958
      78  5S25     LL        0.000000       0.000000       0.000000       1.000000      -0.203716
      79  5S26     LL        0.000000       0.000000       0.000000       1.000000      -0.278209
      80  5S27     LL        0.000000       0.000000       0.000000       1.000000      -0.288844
      81  5S28     LL        0.000000       0.000000       0.000000       1.000000      -0.301764
      82  5S29     LL        0.000000       0.000000       0.000000       1.000000      -0.309239
      83  5S30     LL        0.000000       0.000000       0.000000       1.000000      -0.319003
      84  5S31     LL        0.000000       0.000000       0.000000       1.000000      -0.331446
      85  5S32     LL        0.000000       0.000000       0.000000       1.000000      -0.424036
      86  5S33     LL        0.000000       0.000000       0.000000       1.000000      -0.530995
      87  5S34     LL        0.000000       0.000000       0.000000       1.000000      -0.593545
      88  5S35     LL        0.000000       0.000000       0.000000       1.000000      -0.657459
      89  5S36     LL        0.000000       0.000000       0.000000       1.000000      -0.684185
      90  5S37     LL        0.000000       0.000000       0.000000       1.000000      -0.710959
      91  5S38     LL        0.000000       0.000000       0.000000       1.000000      -0.737911
      92  5S39     LL        0.000000       0.000000       0.000000       1.000000      -0.765172
      93  5S40     LL        0.000000       0.000000       0.000000       1.000000      -0.792864
      94  5S41     LL        0.000000       0.000000       0.000000       1.000000      -1.242437
      95  5S42     LL        0.000000       0.000000       0.000000       1.000000      -1.308294
      96  5S43     LL        0.000000       0.000000       0.000000       1.000000      -1.377036
      97  5S44     LL        0.000000       0.000000       0.000000       1.000000      -1.391363
      98  S6BOUND  EQ        0.000000       0.000000       0.000000       0.000000       0.000000
      99  6S 1     UL        1.000000       0.000000       0.000000       1.000000      -0.024911
     100  6S 2     UL        1.000000       0.000000       0.000000       1.000000      -0.003702
     101  6S 3     LL        0.000000       0.000000       0.000000       1.000000       0.019853
     102  6S 4     LL        0.000000       0.000000       0.000000       1.000000       0.041505
```

```
12FEB69   NON-LINEAR PROBLEM NO 6                                                                        0.   2.   5.


SECTION 2 - COLUMNS                         PRIMAL-DUAL OUTPUT

NUMBER   ..LABEL. AT ...ACTIVITY...   ..INPUT COST..   ..LOWER LIMIT.   ..UPPER LIMIT.   .REDUCED COST.
     103  6S 5     LL        0.000000       0.000000       0.000000       1.000000       0.067268
     104  6S 6     LL        0.000000       0.000000       0.000000       1.000000       0.095584
     105  6S 7     LL        0.000000       0.000000       0.000000       1.000000       0.114636
     106  6S 8     LL        0.000000       0.000000       0.000000       1.000000       0.156163
     107  6S 9     LL        0.000000       0.000000       0.000000       1.000000       0.189295
     108  6S10     LL        0.000000       0.000000       0.000000       1.000000       0.223243
     109  6S11     LL        0.000000       0.000000       0.000000       1.000000       0.257760
     110  6S12     LL        0.000000       0.000000       0.000000       1.000000       0.292652
     111  6S13     LL        0.000000       0.000000       0.000000       1.000000       0.327772
```

```
   112  6S14     LL       0.000000        0.000000        0.000000        1.000000        0.363007
   113  6S15     LL       0.000000        0.000000        0.000000        1.000000        0.398273
   114  6S16     LL       0.000000        0.000000        0.000000        1.000000        0.433507
   115  6S17     LL       0.000000        0.000000        0.000000        1.000000        0.468666
   116  6S18     LL       0.000000        0.000000        0.000000        1.000000        0.503716
   117  6S19     LL       0.000000        0.000000        0.000000        1.000000        0.538635
   118  6S20     LL       0.000000        0.000000        0.000000        1.000000        0.573408
   119  6S21     LL       0.000000        0.000000        0.000000        1.000000        0.584438
   120  6S22     LL       0.000000        0.000000        0.000000        1.000000        0.644504
   121  6S23     LL       0.000000        0.000000        0.000000        1.000000        0.649683
   122  6S24     LL       0.000000        0.000000        0.000000        1.000000        0.685611
   123  6S25     LL       0.000000        0.000000        0.000000        1.000000        0.694108
   124  6S26     LL       0.000000        0.000000        0.000000        1.000000        0.701719
   125  6S27     LL       0.000000        0.000000        0.000000        1.000000        0.759432
   126  6S28     LL       0.000000        0.000000        0.000000        1.000000        0.766510
   127  6S29     LL       0.000000        0.000000        0.000000        1.000000        0.824650
   128  6S30     LL       0.000000        0.000000        0.000000        1.000000        0.831225
   129  6S31     LL       0.000000        0.000000        0.000000        1.000000        0.863443
   130  6S32     LL       0.000000        0.000000        0.000000        1.000000        0.842791
   131  6S33     LL       0.000000        0.000000        0.000000        1.000000        0.808760
   132  S7BØUND  EQ       0.000000        0.000000        0.000000        0.000000        0.000000
   133  7S 1     UL       1.000000        0.000000        0.000000        1.000000       -0.709566
   134  7S 2     UL       1.000000        0.000000        0.000000        1.000000       -0.928400
   135  7S 3     UL       1.000000        0.000000        0.000000        1.000000       -0.838122
   136  7S 4     UL       1.000000        0.000000        0.000000        1.000000       -0.748845
   137  7S 5     UL       1.000000        0.000000        0.000000        1.000000       -0.668881
   138  7S 6     UL       1.000000        0.000000        0.000000        1.000000       -0.636910
   139  7S 7     UL       1.000000        0.000000        0.000000        1.000000       -0.562540
   140  7S 8     UL       1.000000        0.000000        0.000000        1.000000       -0.526085
   141  7S 9     UL       1.000000        0.000000        0.000000        1.000000       -0.457001
   142  7S10     UL       1.000000        0.000000        0.000000        1.000000       -0.405697
   143  7S11     UL       1.000000        0.000000        0.000000        1.000000       -0.335484
   144  7S12     UL       1.000000        0.000000        0.000000        1.000000       -0.271429
   145  7S13     UL       1.000000        0.000000        0.000000        1.000000       -0.212479
   146  7S14     UL       1.000000        0.000000        0.000000        1.000000       -0.193497
   147  7S15     UL       1.000000        0.000000        0.000000        1.000000       -0.149144
   148  7S16     UL       1.000000        0.000000        0.000000        1.000000       -0.115218
```

```
12FEB69   NØN-LINEAR PRØBLEM NØ 6                                                     0.   2.   6.


SECTIØN 2 - CØLUMNS                      PRIMAL-DUAL ØUTPUT

NUMBER  ..LABEL. AT ...ACTIVITY...  ..INPUT CØST..  ..LØWER LIMIT.  ..UPPER LIMIT.  .REDUCED CØST.
   149  7S17     UL       1.000000        0.000000        0.000000        1.000000       -0.083549
   150  7S18     UL       1.000000        0.000000        0.000000        1.000000       -0.053914
   151  7S19     UL       1.000000        0.000000        0.000000        1.000000       -0.026120
   152  7S20     BS       0.069809        0.000000        0.000000        1.000000        0.000000
   153  7S21     LL       0.000000        0.000000        0.000000        1.000000        0.024593
   154  7S22     LL       0.000000        0.000000        0.000000        1.000000        0.047775
   155  7S23     LL       0.000000        0.000000        0.000000        1.000000        0.069687
   156  7S24     LL       0.000000        0.000000        0.000000        1.000000        0.090419
   157  7S25     LL       0.000000        0.000000        0.000000        1.000000        0.110054
   158  7S26     LL       0.000000        0.000000        0.000000        1.000000        0.128694
   159  7S27     LL       0.000000        0.000000        0.000000        1.000000        0.146411
   160  7S28     LL       0.000000        0.000000        0.000000        1.000000        0.163256
   161  7S29     LL       0.000000        0.000000        0.000000        1.000000        0.179305
   162  7S30     LL       0.000000        0.000000        0.000000        1.000000        0.194616
   163  7S31     LL       0.000000        0.000000        0.000000        1.000000        0.115641
   164  S8BØUND  EQ       0.000000        0.000000        0.000000        0.000000        0.000000
   165  8S 1     UL       1.000000        0.000000        0.000000        1.000000       -0.002399
   166  8S 2     UL       1.000000        0.000000        0.000000        1.000000       -0.003814
   167  8S 3     UL       1.000000        0.000000        0.000000        1.000000       -0.002986
   168  8S 4     UL       1.000000        0.000000        0.000000        1.000000       -0.002719
   169  8S 5     UL       1.000000        0.000000        0.000000        1.000000       -0.002096
   170  8S 6     UL       1.000000        0.000000        0.000000        1.000000       -0.001619
   171  8S 7     UL       1.000000        0.000000        0.000000        1.000000       -0.001174
   172  8S 8     UL       1.000000        0.000000        0.000000        1.000000       -0.000758
   173  8S 9     UL       1.000000        0.000000        0.000000        1.000000       -0.000367
   174  8S10     BS       0.573633        0.000000        0.000000        1.000000        0.000000
   175  8S11     LL       0.000000        0.000000        0.000000        1.000000        0.000346
   176  8S12     LL       0.000000        0.000000        0.000000        1.000000        0.000671
   177  8S13     LL       0.000000        0.000000        0.000000        1.000000        0.000979
   178  8S14     LL       0.000000        0.000000        0.000000        1.000000        0.001270
   179  8S15     LL       0.000000        0.000000        0.000000        1.000000        0.001546
   180  8S16     LL       0.000000        0.000000        0.000000        1.000000        0.001808
   181  8S17     LL       0.000000        0.000000        0.000000        1.000000        0.002057
   182  8S18     LL       0.000000        0.000000        0.000000        1.000000        0.002294
   183  8S19     LL       0.000000        0.000000        0.000000        1.000000        0.002519
   184  8S20     LL       0.000000        0.000000        0.000000        1.000000        0.002735
   185  8S21     LL       0.000000        0.000000        0.000000        1.000000        0.003021
   186  S9BØUND  EQ       0.000000        0.000000        0.000000        0.000000        0.000000
   187  9S 1     UL       1.000000        0.000000        0.000000        1.000000        0.000000
   188  9S 2     UL       1.000000        0.000000        0.000000        1.000000        0.000000
   189  9S 3     UL       1.000000        0.000000        0.000000        1.000000        0.000000
   190  9S 4     UL       1.000000        0.000000        0.000000        1.000000        0.000000
   191  9S 5     UL       1.000000        0.000000        0.000000        1.000000        0.000000
   192  9S 6     UL       1.000000        0.000000        0.000000        1.000000        0.000000
   193  9S 7     BS       0.602818        0.000000        0.000000        1.000000        0.000000
   194  9S 8     LL       0.000000        0.000000        0.000000        1.000000        0.000000
```



```
12FEB69   NON-LINEAR PROBLEM NO 6                                                          0.   2.   7.


SECTION 2 - COLUMNS                        PRIMAL-DUAL OUTPUT

NUMBER   ..LABEL. AT ...ACTIVITY...   ..INPUT COST..   ..LOWER LIMIT.   ..UPPER LIMIT.   .REDUCED COST.
   195   9S 9     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   196   9S10     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   197   9S11     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   198   9S12     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   199   9S13     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   200   9S14     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   201   9S15     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   202   9S16     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   203   9S17     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   204   9S18     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   205   9S19     LL        0.000000        0.000000         0.000000         1.000000         0.000000
   206   S10BOUND EQ        0.000000        0.000000         0.000000         0.000000         0.000000
   207   1S 1     UL        1.000000        0.000000         0.000000         1.000000         0.019473
   208   1S 2     UL        1.000000        0.000000         0.000000         1.000000         0.097063
   209   1S 3     UL        1.000000        0.000000         0.000000         1.000000         0.059661
   210   1S 4     UL        1.000000        0.000000         0.000000         1.000000         0.028804
   211   1S 5     BS        0.374868        0.000000         0.000000         1.000000         0.000000
   212   1S 6     LL        0.000000        0.000000         0.000000         1.000000        -0.026954
   213   1S 7     LL        0.000000        0.000000         0.000000         1.000000        -0.052234
   214   1S 8     LL        0.000000        0.000000         0.000000         1.000000        -0.075991
   215   1S 9     LL        0.000000        0.000000         0.000000         1.000000        -0.098359
   216   1S10     LL        0.000000        0.000000         0.000000         1.000000        -0.119444
   217   1S11     LL        0.000000        0.000000         0.000000         1.000000        -0.139374
   218   1S12     LL        0.000000        0.000000         0.000000         1.000000        -0.158230
   219   1S13     LL        0.000000        0.000000         0.000000         1.000000        -0.176089
   220   1S14     LL        0.000000        0.000000         0.000000         1.000000        -0.193043
   221   1S15     LL        0.000000        0.000000         0.000000         1.000000        -0.146168
   222   SEPEND   EQ        0.000000        0.000000         0.000000         0.000000         0.000000
```

```
 12FEB69   NON-LINEAR PROBLEM NO 6                                                          0.   3.   1.

 INTERNAL STATEMENT NUMBER  25     TIME - 11:39
*EXIT*
```

```
TOTAL JOB TIME                           2.03
 PROCESSOR EXECUTION TIME                 .00
 PROCESSOR I/O TIME                       .07
 PROCESSOR OVERHEAD TIME                  .08
 USER EXECUTION TIME                      .56
 USER I/O TIME                            .60
 USER OVERHEAD TIME                       .72
# OF CARDS READ                           994
# OF CARDS PUNCHED                          0
# OF PROCESSOR PAGES OUT                    2
# OF USER PAGES OUT                        18
# OF DIAGNOSTIC PAGES OUT                   0
# OF SCRATCH TAPES USED                     0
# OF SAVE TAPES USED                        0
# OF DISK READS AND WRITES               1436
# OF DISC READS AND WRITES               2814
TEMPORARY DISC SPACE USED                  34
PERMANENT DISC SPACE USED                   0
ACCUM. PERM. DISC SPACE USED                0
```