HONEYWELL EDP

SOFTWARE MANUAL

SERIES 200

LINEAR PROGRAMMING PACKAGE D

GENERAL SYSTEM: SERIES 200/OPERATING SYSTEM - MOD 1

SUBJECT: An In-Core Linear Programming System Using the Standard Simplex Algorithm

SPECIAL INSTRUCTIONS: The reader is assumed to be familiar with the Series 200/Operating System - Mod 1 Fortran Compiler D Reference Manual, Order No. 027.

DATE: March 15, 1966

FILE NO.: 123.8305.001D.0-276*

8781
5366
Printed in U. S. A.

*When ordering this publication please specify Title and Underscored portion of File Number.

PREFACE

This manual describes how to use the Series 200 Linear Programming Package D (abbreviated LPD). Linear programming is a group of mathematical procedures used to calculate what amount of resources, such as raw material or manpower, should be allocated to either maximize some objective such as profit or minimize some objective such as cost.

Linear programming can help to make precise decisions about problems which otherwise are too large and complex to be handled except by trial and error. As a result, this new and powerful mathematical tool will help to solve difficult problems without incurring the uncertainties and losses inherent in a trial and error method.

The LPD source program is written in Honeywell Fortran D language and requires a minimum memory size of 16K characters for compilation. The object program requires at least a 16K memory, and may use additional memory if available.

This manual is divided into six sections. Section I describes how to formulate a linear programming problem. Included is a discussion of the mathematical formulation required, definitions of terms used, kinds of solution obtained, and method of solution used. Section II describes how to prepare the required input data. Included is a discussion of the kind of data used and how it is coded. Section III describes the individual programs used in the LPD system. Included is a discussion of the names and purposes of the programs (called agenda) and a description of how they are coded. Section IV describes the operating procedures for the system. Included are basic equipment requirements, the general procedures used, the names and explanations printed out during and at the end of an LPD run, the SENSE switch options available, and source language considerations for introducing array dimensioning and tolerance adjustments. Section V describes how to interpret the output from an LPD run. Included are a list of printout headings which result and an explanation of each. Finally, Section VI relates all the previous sections to the actual solution of a sample problem. Included is a statement of the sample problem, how it is formulated, the individual programs or agenda required, the input procedure required, the messages which can result, the output obtained, and changes made in the problem to see how these changes affect the original solution.

It is assumed that the reader is familiar with the Fortran Compiler D Reference Manual (order number 027).

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

SECTION I

PROBLEM FORMULATION

This section describes the mathematical formulation of the LP problem. Included is a description of the general mathematical form of the problem, an explanation of terms commonly used, and a general discussion of the kinds of results that can be obtained.

FORMULATION OF THE PROBLEM

The general linear programming problem can be formulated as follows. Find non-negative values of the variables,

$$x_1, x_2, \ldots, x_n$$

which maximize (or minimize) the expression

$$c_1x_1+c_2x_2+\ldots+c_nx_n = Z \qquad (1\text{-}1)$$

and which satisfy a given set of simultaneous equations

$$\begin{aligned}
a_{11}x_1+a_{12}x_2+\ldots+a_{1n}x_n &= b_1\\
a_{21}x_1+a_{22}x_2+\ldots+a_{2n}x_n &= b_2\\
&\vdots\\
a_{m1}x_1+a_{m2}x_2+\ldots+a_{mn}x_n &= b_m
\end{aligned} \qquad (1\text{-}2)$$

Equation (1-1) is called the objective function. Equations (1-2) are called the constraints of the problem. The requirement that the variables be non-negative in value is called the non-negativity condition for the problem. The a's, b's and c's are constants while the x's are variables. The individual a's are called input/output (I/O) coefficients. The b's are called the constraint values. The c's are called either cost coefficients or objective coefficients. In this manual, they are called objective coefficients. In equation (1-1), Z is called the objective value. The column made up of the b's is called the right hand side, the constraint vector, the requirements vector, or the stipulations vector. In this manual, it is called the right hand side.

The above formulation is the standard form of the linear programming problem where all the constraints (1-2) are initially written as equations or have been reduced to equations. However, linear programming methods are so general that the constraints can initially be inequalities as well. Such constraints can always be made into equalities by adding or subtracting a compensating quantity to the left side of each inequality. Whether the compensating quantity is added or subtracted depends on whether the inequality is of the less-than or greater-than type.

If a constraint is of the less-than type,

$$a_{i1}x_1+a_{i2}x_2+\ldots+a_{in}x_n \quad b_i \qquad (1\text{-}3)$$

it can always be made into an equation by adding a non-negative value s_i to the left hand side. The result is

$$a_ix_1+a_{i2}x_2+\ldots+a_{in}x_n+s_i = b_i \qquad (1\text{-}4)$$

The variable s_i is another unknown in the equation and is called a slack variable. If a constraint is of the greater-than type,

$$a_{i1}x_1+ a_{i2}x_2+\ldots+a_{in}x_n \quad b_i \qquad (1\text{-}5)$$

a non-negative slack variable s_i is subtracted to produce the equation,

$$a_{i1}x_1+a_{i2}x_2+\ldots+a_{in}x_n - s_i = b_i \qquad (1\text{-}6)$$

Linear Programming Package D introduces slack variables automatically; no calculations are required of the user.

KINDS OF SOLUTIONS

Since the number of variables in a linear programming problem is generally greater than the number of constraints, the result is that the constraints are undetermined. This means that there can be an infinite number of solutions to the problem. Any solution of the constraints (1-2) which also satisfies the non-negativity condition is called a feasible solution to the problem.

However, because the number of variables is usually greater than the number of constraints, the constraints can only be solved for all values of the variables if the number of variables equals the number of constraints, or if the extra variables are set equal to zero. Since pertinent constraints in a problem cannot be eliminated to make the number of variables equal the number of constraints, the excess variables must be set equal to zero. To obtain all possible sets of values for the variables under this condition, every combination of the excess variables is set equal to zero and the remaining variables are solved for simultaneously. Any solution of the constraints obtained by setting the excess variables equal to zero is called a basic solution to the problem. The collection of non-zero variables is called the basis of the problem. A basic solution which also satisfies the non-negativity condition is called a basic feasible solution to the problem. Any feasible solution which minimizes Z in the objective function (1-1) is called an optimal feasible solution to the problem. Any basic feasible solution which minimizes Z is called an optimal basic feasible solution to the problem. If the constraints do not confine the objective function to finite values, the problem is said to be unbounded. Linear Programming Package D considers only basic solutions to a problem.

METHOD OF SOLUTION

LPD first generates an initial basic feasible solution. To do so, the system converts any

inequality constraints to equalities by introducing positive or negative slacks. If some of the constraints do not have positive slacks, a method called Phase I is used to find the initial basic feasible solution by introducing artificial vectors. When the system solves for the optimal solution finally desired, the method used is called Phase II. Phase I may not always be required, but Phase II is always used. If there is no feasible solution or the solution is unbounded, a message is given.

For either Phase I or Phase II, the objective function and constraints are first put in matrix form. A matrix is any array of numbers, called matrix elements, arranged in rows and columns. The matrix elements for the objective function and constraints of a linear programming problem are the objective coefficients and I/O coefficients, respectively. The right hand side is a separate column placed to the right of the matrix. This combination of the matrix and right hand side shown below is called a tableau.

$$\begin{array}{cccc} c_{11} & c_{12} \cdots\cdots c_{1n} & & \\ a_{11} & a_{12} & a_{1n} & b_1 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ a_{m1} & a_{m2} & a_{mn} & b_n \end{array} \qquad (1\text{-}7)$$

The system uses a maximizing algorithm, called the standard simplex method. If it is necessary to minimize a given objective function, the objective coefficients must first be multiplied by - 1. This is done automatically by setting a scale factor to - 1. The system will then maximize - Z which is equivalent to minimizing the original Z.

SECTION II

INPUT DATA

This section contains instructions for preparing data for input to LPD. Data formats are described and the sequence in which the data must appear is specified.

Input data consists of row names, column names, matrix elements (including objective function coefficients) and the right hand side. Data is submitted as card images, punched in a format similar to the SHARE and other linear programming standards. The sequence of input data is shown in Figure 2-1. The data definitions and the preparation of each card in the input deck are explained below.

INPUT CARD CODING

There are two kinds of input cards, indicator cards and data cards. Indicator cards are control cards which announce the data cards to follow. Data cards contain the names and numerical values of the matrix data for the problem. Data cards include row-name cards, matrix-element cards, and right-hand-side element cards.

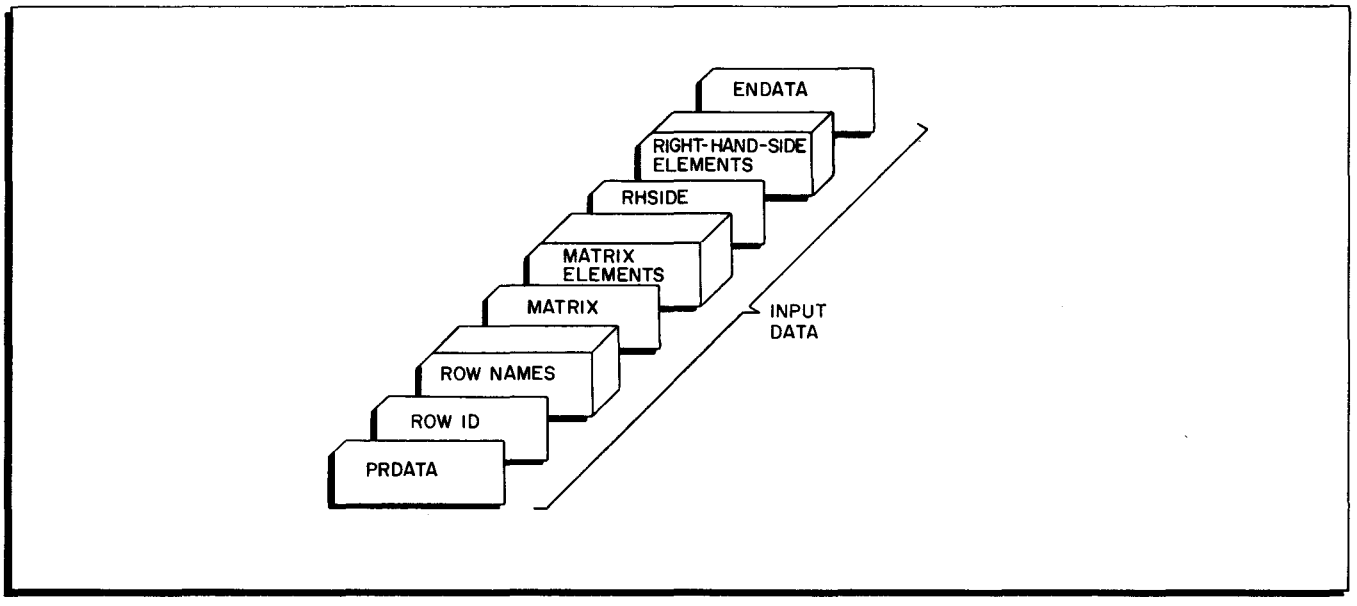


Figure 2-1. Input Data Sequence

The coding of indicator and data cards is described below. A special coding form is available for use in preparing the input data. Shown first is the actual layout for the card on the coding form. Below this layout is an interpretation of the coding, including the number of columns to be punched and the description of each parameter involved.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7	12	13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 PRDATA			a						
2									

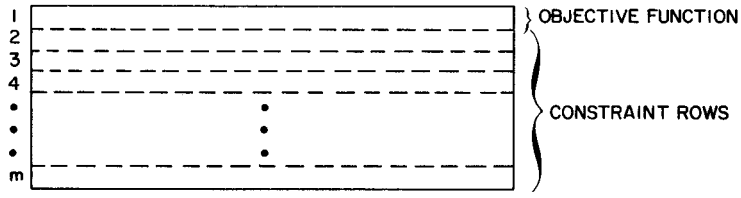
Columns	Parameter	Description
1-6	PRDATA	PROBLEM-DATA INDICATOR. This parameter specifies that the cards between this card and the ENDATA card are data cards for one problem.
13-18	a	PROBLEM IDENTIFICATION. This alphanumeric parameter must be unique with no leading blanks.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7	12	13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 ROWΔID									
2		a	b						

Columns	Parameter	Description
1-6	ROWΔID	ROW IDENTIFICATION INDICATOR. This card specifies that the data cards that follow consecutively are row-name cards. One row description is punched per card as follows:
12	a	ROW TYPE. If the row is an inequality row of the sense (≤), parameter "a" must be a "+". If the row is an inequality row of the sense (≥), parameter "a" must be "-". If the row is an equality row (=), as is the objective function row, parameter "a" must be blank (Δ) or "0".
13-18	b	ROW NAME. Alphanumeric. Cannot be blank or duplicate a column name of a matrix-element-card or right-hand-side card. Should be unique within the problem. The name of the objective function row must precede the names of the constraint rows, and all row-name cards precede the MATRIX card described below.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7	12	13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 MATRIX									
2	a		b	c	d		e		

Columns	Parameter	Description
1-6	MATRIX	MATRIX INDICATOR CARD. This card specifies that the data cards that follow consecutively are matrix-element cards. Matrix element cards define the position and value of each element. Vectors (columns) cannot be split, i.e., all elements for a given

Columns	Parameter	Description
		matrix column must appear together. Vectors can appear in random order, and elements can appear in random order within vectors. Matrix elements are punched one per card as follows:
7-12	a	COLUMN NAME. This parameter assigns an alphanumeric name to the matrix column under consideration. The name cannot be blank and cannot duplicate a row name or a name assigned to another column. As mentioned above, matrix element values are specified by column. That is, all matrix-element cards with the same column name must appear together.
13-18	b	ROW NAME. This alphanumeric name corresponds to the appropriate row of the element as defined under ROWΔID. The matrix rows are indexed by the LPD system beginning with one. For example, the rows would be indexed as follows: 
19	c	SIGN OF ELEMENT. If the element is zero or positive, leave this column blank or enter "+"; if the element is negative enter "-".
20-23	d	INTEGRAL PART OF ELEMENT. These decimal digits must be right-justified against the decimal point. Leading zeros can be omitted.
24	.	DECIMAL POINT OF ELEMENT. The decimal point should be punched in this column. If the decimal point is omitted, it will be supplied by the system.
25-30	e	FRACTIONAL PART OF ELEMENT. These decimal digits must be left-justified against the decimal point. Trailing zeros can be omitted. Only non-zero entries need be specified. NOTE: To accommodate a value whose integral or fractional part exceeds the allotted space on the coding form, the decimal point may be punched anywhere in columns 20-30, provided that the integral part, decimal point, and fractional part can all be contained in these columns. However, such data cards are not acceptable to Honeywell H-400/1400 or H-800/1800 linear programming systems.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12		13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 RHSIDE									
2	a		b	c	d	.	e		

Columns	Parameter	Description
1-6	RHSIDE	RIGHT-HAND-SIDE INDICATOR CARD. This card specifies that the data cards that follow consecutively are right-hand-side-element cards. Only one right-hand side vector is allowed, hence the same name appears in columns 7-12 of every right-hand-side-element card. Elements can appear in random order within the vector. Right-hand-side elements are punched one per card as follows:
7-12	a	NAME OF RIGHT-HAND-SIDE VECTOR. This alphanumeric name cannot be blank and cannot duplicate a column or row name.
13-18	b	ROW NAME. This alphanumeric name corresponds to the appropriate row of the element, as defined under ROWΔID.
19	c	SIGN OF ELEMENT. If the element is zero or positive, leave this column blank or enter "+"; if negative, enter "-".
20-23	d	INTEGRAL PART OF ELEMENT. These decimal digits must be right-justified against the decimal point.
24	.	DECIMAL POINT OF ELEMENT. The decimal point should be punched in this column. If the decimal point is omitted, it will be supplied by the system.
25-30	e	FRACTIONAL PART OF ELEMENT. These decimal digits must be left-justified against the decimal point. Trailing zeros can be omitted and only non-zero entries need be specified.
		NOTE: To accommodate a value whose integral or fractional part exceeds the allotted space on the coding form, the decimal point may be punched anywhere in columns 20-30, provided that the integral part, decimal point, and fractional part can all be contained in these columns. However, such data cards are not acceptable to Honeywell H-400/1400 or H-800/'800 linear programming systems.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12		13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 ENDATA									
2									

Columns	Parameter	Description
1-6	ENDATA	END-OF-DATA INDICATOR CARD. This card must appear in the input deck in the position indicated in Figure 2-1.

SECTION III
AGENDA

The LPD system consists of a series of discrete programs. Each program is called an agendum. The processing functions of each LPD run are determined by the programs requested on control, or agendum, cards.

NATURE AND PURPOSE OF AGENDA

The programmer's agendum call cards are interpreted by a control program which loads the agenda for the LPD run into memory, providing complete control over the actions of the system, the sequence in which the actions are performed, and the final output produced. As a result, complete system flexibility is always available. Depending on his processing needs, the programmer uses control cards to specify any of the following agenda:

1. INPUT, a,b
2. NORMAL
3. OUTPUT
4. OBJECT
5. RIGHT
6. DUAL

AGENDUM CARD CODING

The coding form used for input cards is also used for agendum cards. Shown first is the actual layout for each card on the coding form. Below this layout is an interpretation of the coding, including the number of columns to be punched and the description of each parameter involved.

Agenda names are punched one to a card beginning in column one. These agenda cards are then inserted in the card reader in the order in which the agenda are to be called. Only those agenda which are actually used in the run need to be called.

INDICATOR FIELD	COLUMN NAME / ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN / INTEGRAL PART / DEC. PT. / FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12	13 18	19 20 23 24 25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 INPUT,	a,b				
2					

Columns	Parameter	Description
1-6	INPUT,	The INPUT agendum card should be followed by the input data for the current problem, as described in Section II. INPUT reads the data and establishes the standard simplex tableau in memory.
7-8	a	a = +1 if objective function is to be maximized. a = -1 if objective function is to be minimized.
9-10	,b	b = R for a Restart after a GETOFF option or for post-optimal analysis by OBJECT, RIGHT, and DUAL agenda. (See Section IV). Otherwise omit "b".

INDICATOR FIELD	COLUMN NAME / ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN / INTEGRAL PART / DEC. PT. / FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12	13 18	19 20 23 24 25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 NORMAL					
2					

Columns	Parameter	Description
1-6	NORMAL	The NORMAL agendum carries out the simplex method. It first finds a feasible solution if one is not given and then finds an optimum solution. If there is no feasible solution or the solution is unbounded, a message is given as shown in Table 4-1.

INDICATOR FIELD	COLUMN NAME / ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN / INTEGRAL PART / DEC. PT. / FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12	13 18	19 20 23 24 25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 OUTPUT					
2					

Columns	Parameter	Description
1-6	OUTPUT	The OUTPUT agendum prints the final solution and, optionally, intermediate data.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12		13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 OBJECT									
2	a			b				c	

Columns	Parameter	Description
1-6	OBJECT	The OBJECT agendum reads changes to the objective function and updates the tableau. Each change is punched in a separate card and in the following format:
7-12	a	Column name.
19-30	b	Previous objective coefficient value.
31-42	c	New objective coefficient value.
		The above change cards should follow the OBJECT agendum card. After each change is made, a message is given indicating whether the solution is optimal. There is no limit on the number of changes. The last change should be followed by a blank card.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12		13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 RIGHT									
2	a		b	c				d e	

Columns	Parameter	Description
1-5	RIGHT	The RIGHT agendum reads changes to the right hand side and updates the right hand side and solution. If a change is made to the right hand side element of row i, then the initial tableau must contain a unit vector with a + 1.0 in row i. This may be a dummy vector with a large negative objective coefficient to keep it out of the basis, or it may be a positive slack. Each change is punched in a separate card in the following format:
7-12	a	Name of the dummy vector or blank if a slack vector is used.
13-18	b	Name of the row in which the change is being made.
19-30	c	Previous right hand side value
31-42	d	New right hand side value
43-54	e	Objective coefficient value of dummy vector, or blank if a slack is used.
		The above described change cards must follow the RIGHT agendum card. After each change is made, a message is given indicating whether the solution is optimal. There is no limit to the number of changes that can be made. The last change should be followed by a blank card.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12		13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 DUAL									
2									

Columns	Parameter	Description
1-4	DUAL	The DUAL agendum performs the dual simplex algorithm. This agendum is used after the right hand side has been changed using RIGHT and the solution is no longer feasible because one or more right hand side elements are negative. The DUAL agendum then determines the new optimal solution.

INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
1 6	7 12		13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1 ENDLPD									
2									

Columns	Parameter	Description
1-6	ENDLPD	The ENDLPD card is placed after the last agendum card for the last problem.

SECTION IV
OPERATING PROCEDURES

This section describes the required operating procedures for LPD. Included are the equipment requirements, general operating procedure, available messages with explanations and/or suggested actions, and SENSE switch options.

EQUIPMENT REQUIREMENTS

The minimum equipment required depends upon the size of the object program, and the size of the problem to be solved.

Compilation of the LPD Source Program

The LPD source program can be compiled on any Series 200 system equipped with:

1. a 16K memory
2. 4 magnetic tape units
3. 1 printer
4. 1 card reader
5. the Advanced Programming Feature (011)
6. the Edit Feature (013)

A card punch may be included as optional equipment.

Execution of the LPD Object Program

Minimum equipment required for the LPD object program is:

1. a 16K memory
2. 1 magnetic tape unit
3. 1 printer
4. 1 card reader
5. the Advanced Programming Feature (011)
6. the Edit Feature (013)

The object program may also require a card punch for certain options. As specified below, the object program may use additional memory above the 16K minimum requirement to solve linear programming problems of increasing size.

Problem Size

The memory required to solve different sized problems is specified below. These estimates assume that the number of columns (including slacks) is twice the number of rows in the problem.

Maximum Number of Rows	Memory Requirement
11	16K
30	32K
34	40K
39	48K
49	65K

SOURCE LANGUAGE CONSIDERATIONS

Two parameters must be considered before compiling the LPD source program, array dimensions and tolerances.

Instructions for Dimensioning Arrays

Let m = number of constraints (excluding the objective function).

n = number of variables and slacks (excluding the right hand side).

The arrays used are dimensioned in the LPD source program by inserting the following COMMON cards in each chain:

```
COMMON A (I, J), W(I), L(I)
COMMON NROW1 (I), NROW2(I), NCOL1(J), NCOL2(J)
```

where I = m+2

J = n+1

Once these arrays are dimensioned, it is not always necessary to change them for each problem. For any problem with the number of constraints $\leq$ m and the number of variables and slacks $\leq$ n, the same dimensions can be used.

Tolerances

In each step of the simplex algorithm, the program computes a D/J value for each vector. D/J is proportional to the change in the objective function for introducing the particular vector into the current basis. When testing D/J values to determine if an optimal solution has been reached, a variable called DJT is used. If $|D/J| \leq$ DJT, small values of D/J are considered to be zero. A good beginning value to use for DJT is .001. If this value does not produce an optimal solution, try different values of DJT. The parameter DJT is set in the EXEC chain and may require adjustment for different problems. The adjustment is made simply by inserting a new value and recompiling.

RUNNING PROCEDURES

There are two Fortran running modes available. They are called Load-And-Go and Go-Later. The operating procedures for each are described in the Fortran Compiler D Reference Manual.

MESSAGES

Each time an agendum is called, the name of that agendum is printed. In addition, intermediate messages are given to indicate the progress of the run. These messages, together with explanations and suggested actions where applicable, are listed in Table 4-1.

Table 4-1. Messages and Explanations

Agendum	Message	Explanation/Action
INPUT	PR DATA AAAAAA	This message presents the contents of the PR DATA card for the current problem.
	DATA LOADED	This message indicates that input is finished. The next agendum card is then read.
NORMAL	INFEAS.	This message indicates the problem has no feasible solution. The next agendum card is then read.
	FEASIBLE	This message indicates that a feasible solution has been found and that calculations will continue.
	UNBDD.	This message indicates that the problem has an unbounded solution. The next agendum card is then read.
	OPTIMUM	This message indicates that an optimum solution has been found. The next agendum card is then read.
OUTPUT	GETOFF	This message indicates that a GETOFF has been initiated.
	END.	This message indicates the end of GETOFF. All processing then stops.
OBJECT	STILL OPTIMAL	This message indicates that the solution is still optimal after the last coefficient change. The next card is then read.
	NOT OPTIMAL	This message indicates that the solution is no longer optimal. When this occurs, the program pauses (see SENSE Switch Options section).
	VARIABLE NOT FOUND	This message shows that the objective coefficient to be changed is not defined. The program then reads the next change card.
RIGHT	OPTIMAL	This message indicates that the solution is optimal after the last requirements change. The program then reads the next change card.
	NOT FEASIBLE	This message indicates the solution is no longer feasible. When this occurs, the program then pauses (see SENSE Switch Options section).
	ROW NOT FOUND	This message indicates that the row in which a change is to be made is not defined. The program then reads the next change card.

Table 4-1 (cont). Messages and Explanations

Agendum	Message	Explanation/Action
	VARIABLE NOT FOUND	This message indicates that the variable to be used in changing the right-hand side is not defined. The program reads the next change card.
DUAL	INFEAS.	This message indicates the solution is infeasible. Then the program reads the next agendum card.
	OPTIMUM	This message indicates an optimal solution has been found. The program reads the next agendum card.
	ENDLPD	This message indicates the end of the entire run.

SENSE SWITCH OPTIONS

Several SENSE switch options enhance the ease of operating the system. They are described below under the agenda with which they are associated.

OUTPUT Agendum

Associated with the OUTPUT agendum are a print option and a punch option. SENSE switch 1 controls the print option while SENSE switch 2 controls the punch option. Table 4-2 describes how each switch is used.

Table 4-2. SENSE Switch Settings for OUTPUT Agendum

		ACTION	
Option	Switch	ON Position	OFF Position
PRINT	1	Permits printing variables not in the basis together with their D/J values.	Suppresses printing of such variables and their D/J values
PUNCH	3	Permits punching the entire tableau in a format suitable for reading by the INPUT agendum. The resulting punched deck includes the PRDATA and ENDATA cards. The punch option may be used to save the optimal tableau for later post-optimal calculations.	Suppresses the punching permitted in the ON position.

NORMAL Agendum

The option associated with the NORMAL agendum is the GETOFF option. SENSE switch 2 controls this option. Table 4-3 describes how the switch is used.

Table 4-3. SENSE Switch Setting for NORMAL Agendum

Option	SENSE Switch	ACTION	
		ON Position	OFF Position
GETOFF	2	Permits interrupting the current iteration and automatically exiting to the OUTPUT agendum. The entire tableau is then punched in the same format that is used for the INPUT agendum. The punched deck may be reloaded at a later time and the run will continue where it left off. This option permits long runs to be divided into several short ones.	No interruption.

To restart a problem after the GETOFF option has been used, the INPUT agendum card must have the format: INPUT, a, b, where a = the same value as at the start of the problem and b = R. If the original INPUT agendum card does not have R, it must be repunched in the format just described and put at the front of the resulting punched deck when used at a later time. In addition, a NORMAL agendum card must be added to the back of the resulting punched deck when it is rerun, in order to restart the NORMAL agendum. After this, other agenda may be called as they are needed. Figure 4-1 illustrates the sequence described above.

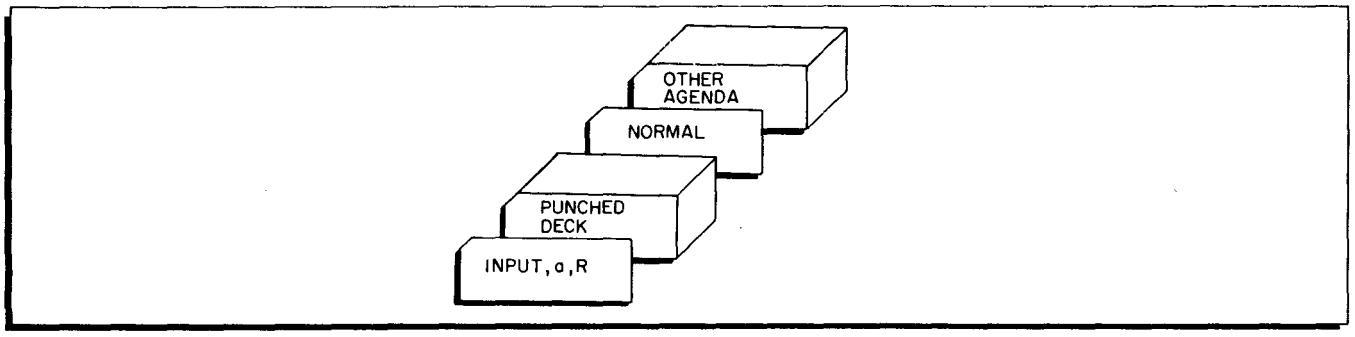


Figure 4-1. Sequence of Rerun Deck

OBJECT Agendum

When a NOT OPTIMAL message is printed, the program halts with 70700 in the A Address Register. At this time, the option to either read the next change card or skip the remaining change cards and read the next agendum card is available. SENSE switch 4 is used to exercise this option. Table 4-4 describes how the switch is used.

Table 4-4. SENSE Switch Setting for RIGHT and OBJECT Agenda

Option	SENSE Switch	ACTION	
		ON Position	OFF Position
NEXT AGENDUM	4	Press RUN and the program skips to the next agendum card.	Press RUN and the program reads the next change card for the current agendum.

RIGHT Agendum

When a NOT FEASIBLE message is printed, the program pauses with 70700 in the A Address Register. At this time, the option to either read the next change card or skip the remaining change cards and read the next agendum card is again available. SENSE Switch 4 is again used to exercise this option, as described above in Table 4-4.

SECTION V
LPD OUTPUT

This section presents examples and explanations of the printout headings produced by LPD. The printout headings detail the possible intermediate and final data generated by the system.

An explanation of each printout heading, listed under the appropriate agendum, is given in Table 5-1. For a further explanation of the terms given in Table 5-1, consult Matrices, Elimination and the Simplex Method by William Orchard-Hayes.

Table 5-1. Printout Headings and Explanations

Agendum	Printout Heading	Explanation
NORMAL	INTERS.	Number of iterations completed.
	OBJECTIVE	Current value of objective function.
	VAR. IN	Name of last vector to enter basis.
OUTPUT	BASIS VAR.	Names of basis variables.
	VALUE	Values of basis variables.
	VARIABLE	Names of non-basic variables with $D/J \neq 0$.
	D/J	D/J value of non-basic variables.
OBJECT	VARIABLE	Name of the variable whose objective coefficient is being changed.
	OLD COEFF.	Old value of the objective coefficient.
	NEW COEFF.	New value of the objective coefficient.
RIGHT	ROW	Name of row in which the change is being made to the right-hand side.
	OLD REQ.	Old value of right-hand side element.
	NEW REQ.	New value of right-hand side element.
DUAL	ITERS.	Number of iterations.
	OBJECTIVE	Current value of the objective function.
	VAR. IN	Name of last variable to enter the basis.

SECTION VI
SAMPLE PROBLEM

This section describes how a sample problem is set up and solved using LPD. The problem is called the Breakfast Food Problem. Included in the discussion is a statement of the problem, how it is formulated, and how it is coded. The solution printouts are also included.

BREAKFAST FOOD PROBLEM (BFP)

The Wholesome Cereal Company wishes to mix four breakfast foods in such a way as to achieve certain dietary properties while minimizing the cost of the resulting mixture. The dietary properties involved are the sodium, protein and caloric constituents in each breakfast food. The cost of each breakfast food and the dietary properties of each are shown in Table 6-1.

Table 6-1. Cereal Mixture Data

	CRISPIES	CRUNCHIES	CRACKLES	CHORTLES	CEREAL MIXTURE
COST	4.0	7.0	8.0	6.0	
CALORIES	150.0	140.0	170.0	160.0	=150.0
SODIUM	0.1	0.1	0.3	0.3	$\leq$ 0.2
PROTEIN	2.0	4.0	5.0	3.0	$\geq$ 3.0

As shown in the table above, the cereal mixture is to have 150 calories, the sodium content is not to exceed 0.2 grams, and the protein content must be at least 3.0 grams. This and other data in the table are punched on data cards as described on the following pages.

Formulating the Breakfast Food Problem

From the data in Table 6-1, the first step is to put the problem in mathematical form. Using the terminology defined in Section I, the objective coefficients (the c's in equation 1-1) are the costs of each breakfast food in Table 6-1: 4.0 for CRISPIES, 7.0 for CRUNCHIES, 8.0 for CRACKLES and 6.0 for CHORTLES. The level of the activities (the x's) are the unknown amounts of each breakfast food required to minimize the cost of the resulting mixture. The objective value (the Z in equation 1-1 of Section I) is the unknown minimum cost of the mixture. In mathematical form, the objective function is:

$$4.0x_1+7.0x_2+8.0x_3+6.0x_4 = Z \qquad (6-1)$$

where 4.0 = unit cost of CRISPIES

x_1 = amount of CRISPIES needed

7.0 = unit cost of CRUNCHIES

x_2 = amount of CRUNCHIES needed

8.0 = unit cost of CRACKLES

x_3 = amount of CRACKLES needed

6.0 = unit cost of CHORTLES

x_4 = amount of CHORTLES needed

Again using the terminology defined in Section I, the I/O coefficients (the a's of equations 1-2 in Section I) are the calories, sodium, and protein contents of each breakfast food in Table 6-1. The constraint values (the b's of equations 1-2) are the values shown in the cereal mixture column of the table. In mathematical form, the constraints are:

$$\begin{aligned} 150.0x_1 + 140.0x_2 + 170.0x_3 + 160.0x_4 &= 150.0 \\ 0.1x_1 + 0.1x_2 + 0.3x_3 + 0.3x_4 &\leq 0.2 \\ 2.0x_1 + 4.0x_2 + 5.0x_3 + 3.0x_4 &\geq 3.0 \end{aligned} \qquad (6\text{-}2)$$

where 150.0= calories per unit of CRISPIES

140.0 = calories per unit of CRUNCHIES

170.0 = calories per unit of CRACKLES

160.0 = calories per unit of CHORTLES

0.1 = sodium per unit of CRISPIES

0.1 = sodium per unit of CRUNCHIES

0.3 = sodium per unit of CRACKLES

0.3 = sodium per unit of CHORTLES

2.0 = protein per unit of CRISPIES

4.0 = protein per unit of CRUNCHIES

5.0 = protein per unit of CRACKLES

3.0 = protein per unit of CHORTLES

Finally, since the amount of each breakfast food used must be equal to or greater than zero, the non-negativity condition for this example is:

$$x_j \geq 0 \text{ where } j = 1, 2, 3, 4 \qquad (6\text{-}3)$$

Equations (6-1) through (6-3) constitute the mathematical formulation of this sample problem.

Coding the Breakfast Food Problem Data

To code the breakfast food problem data, a matrix of detached coefficients, taken from the objective function (6-1) and the constraints (6-2), should first be written. Rewriting (6-1) and (6-2):

$$
\begin{aligned}
4.0x_1 + \ 7.0x_2 + \ 8.0x_3 + \ 6.0x_4 &= Z \\
150.0x_1 + 140.0x_2 + 170.0x_3 + 160.0x_4 &= 150.0 \\
0.1x_1 + \ 0.1x_2 + \ 0.3x_3 + \ 0.3x_4 &\leq \ 0.2 \\
2.0x_1 + \ 4.0x_2 + \ 5.0x_3 + \ 3.0x_4 &\geq \ 3.0
\end{aligned}
\qquad (6\text{-}4)
$$

The matrix of detached coefficients, together with the right-hand side of the constraints, constitutes the input tableau as shown in (6-5). Included are the chosen column names and row names to be used in the coding.

CRISP	CRUNCH	CRACK	CHORT	ONEΔΔΔ (column names)	
4.0	7.0	8.0	6.0		COST
150.0	140.0	170.0	160.0	150.0	Calory
0.1	0.1	0.3	0.3	0.2	Sodium
2.0	4.0	5.0	3.0	3.0	Proten
Matrix				Right-Hand Side	Row Names

(6-5)

Using (6-5), the input data is prepared as described in Section II. The actual coding is shown in Figure 6-1. An additional column called DUMMY has also been added to the data. This column is a dummy vector with a large positive cost. Because of its large value, this vector will not be in the optimal solution and does not therefore interfere with the data of the original problem. The sole purpose of the vector is to provide a unit positive vector for making changes to the right-hand side after an optimum solution is found (see Post-Optimal Analysis).

Input Procedure

The complete input deck for this problem is shown in Figure 6-2. The three agenda INPUT, a, NORMAL, and OUTPUT are required to solve the problem. Consult Section III to see how the cards for these agenda are coded. Note on the INPUT agendum card that since the objective function is to be minimized, "a" is set equal to "-1." The ENDLPD card is placed after the OUTPUT agendum card.

There are three basic steps in the input procedure:

1. Load the LPD program.
2. Insert the input deck shown in Figure 6-2 into the card reader.
   a. Agendum cards must appear in the order shown.
   b. Each agendum card must be followed by the data cards required for that agendum, again as shown.

PROBLEM THE BREAKFAST FOOD PROBLEM DATE __/__/__ PAGE ____ OF ____

	INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
	1 6	7 12		13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1	PRDATA			BFP						
2	ROWΔID									
3			Ø	COST						
4			Ø	CALORY						
5			+	SODIUM						
6			-	PROTEN						
7	MATRIX									
8		CRISP,I		COST		4	.	Ø		
9		CRISP,I		CALORY		15Ø	.	Ø		
10		CRISP,I		SODIUM		Ø	.	1		
11		CRISP,I		PROTEN		2	.	Ø		
12		CRUNCH		COST		7	.	Ø		
13		CRUNCH		CALORY		14Ø	.	Ø		
14		CRUNCH		SODIUM		Ø	.	1		
15		CRUNCH		PROTEN		4	.	Ø		
16		CRACKL		COST		8	.	Ø		
17		CRACKL		CALORY		17Ø	.	Ø		
18		CRACKL		SODIUM		Ø	.	3		
19		CRACKL		PROTEN		5	.	Ø		
20		CHORTL		COST		6	.	Ø		
21		CHORTL		CALORY		16Ø	.	Ø		
22		CHORTL		SODIUM		Ø	.	3		
23		CHORTL		PROTEN		3	.	Ø		
1		DUMMY		COST		1ØØ	.	Ø		
2		DUMMY		CALORY		1	.	Ø		
3	RHSIDE									
4		ONE		CALORY		15Ø	.	Ø		
5		ONE		SODIUM		Ø	.	2		
6		ONE		PROTEN		3	.	Ø		
7	ENDATA									

Figure 6-1. Coding of Input Data for Breakfast Food Problem

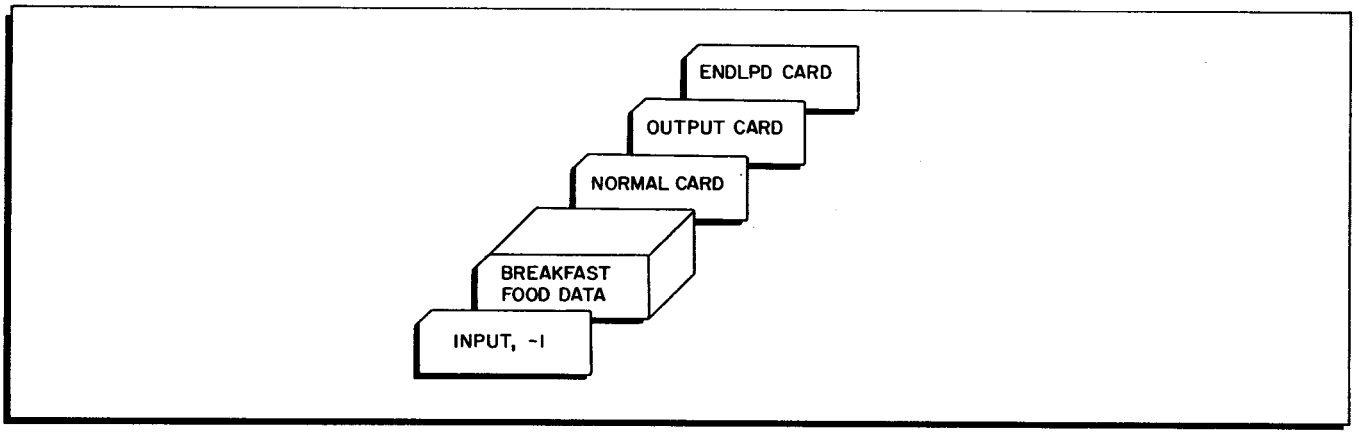


Figure 6-2. Input for Breakfast Food Problem

c. When more than one problem or other versions of the same problem are being run (as in the two variations of this problem described later), all the problems can be stacked in the card reader followed by an ENDLPD card.

3. Start the program.

Messages

Each time an agendum is called, the name of that agendum is printed. In addition, intermediate messages are given to indicate the progress of the run. These messages, together with explanations and suggested actions where applicable, are shown in Table 4-1.

Output

The complete printed output for this problem is shown in Figure 6-3.

```
INPUT, -1
PRDATA              BFP
DATA LOADED

NORMAL
ITERS.              OBJECTIVE       VAR.IN
   1                  -4.800000     CRACKL
   2                  -5.268292     CRISPI

FEASIBLE
OPTIMUM

OUTPUT
COST                  -5.268292

BASIS VAR.             VALUE
CRISPI                  .585365
SLP003                  .031707
CRACKL                  .365853

VARIABLE               D/J
SLN004                1.268292
CRUNCH                 .560975
CHORTL                 .634146
DUMMY                99.990250
```

Figure 6-3. Output for Breakfast Food Problem

SENSE switch 1 is set to ON so that all non-basic variables together with their D/J values are listed in the output. In addition, SENSE switch 3 is set ON so that the optimum tableau is punched and saved for use in the post-optimal analysis of the problem described below.

The vectors SLP003 and SLN004 which appear in Figure 6-3 are respectively the positive and negative slacks introduced by LPD, according to the ROWID cards in the input data. LPD assigns a name beginning with SLP to each positive slack introduced and a name beginning with SLN to each negative slack introduced. Artificial vectors used in Phase I are denoted by the configuration 000000.

POST-OPTIMAL ANALYSIS

Once the initial optimum solution shown in Figure 6-3 has been obtained, it is sometimes useful to change one or more of the objective coefficients or right-hand side elements in order to see how their change affects the initial optimum solution. Two such post-optimal changes in the Breakfast Food Problem are described below.

PROBLEM BREAKFAST FOOD PROBLEM DATE / / PAGE OF

	INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT: SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
	1 6	7 12		13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1	OBJECT									
2		CRISPI				4	.	Ø	4.2	
3		CRISPI				4	.	2	4.4	
4		CRISPI				4	.	4	4.6	
5		CRISPI				4	.	6	4.8	
6		CRISPI				4	.	8	5.Ø	
7		CRISPI				5	.	Ø	5.2	
8										
9										
10										
11										

Figure 6-4. OBJECT Change Cards

Objective Coefficient Change

Suppose that a change in the availability of wheat is expected to drive the cost of CRISPIES upward. Vector CRISPI is in the optimal basis. If the price of CRISPIES rises high enough, vector CRISPI will have to leave the basis to maintain an optimal solution. We want to know the critical price at which the change in the basis is required. For prices slightly larger than the critical price, we also want to know what the new optimal solution is, i.e., we want to know what cereals to mix, in what proportions, and at what minimal cost.

The set of change cards in Figure 6-4 progressively increments the objective coefficient 4.0 by 0.2 until either the current value of the coefficient is 5.2 or the solution becomes non-optimal. If the solution becomes non-optimal, the program halts before all the changes have been implemented (see Table 4-4). The operator is instructed then to set SENSE switch 4 ON, so that the remaining change cards are skipped, the agendum NORMAL (the next agendum card) reoptimizes the problem, and the agendum OUTPUT prints the solution.

The change cards in Figure 6-4 are inserted in the OBJECT run deck in Figure 6-5.

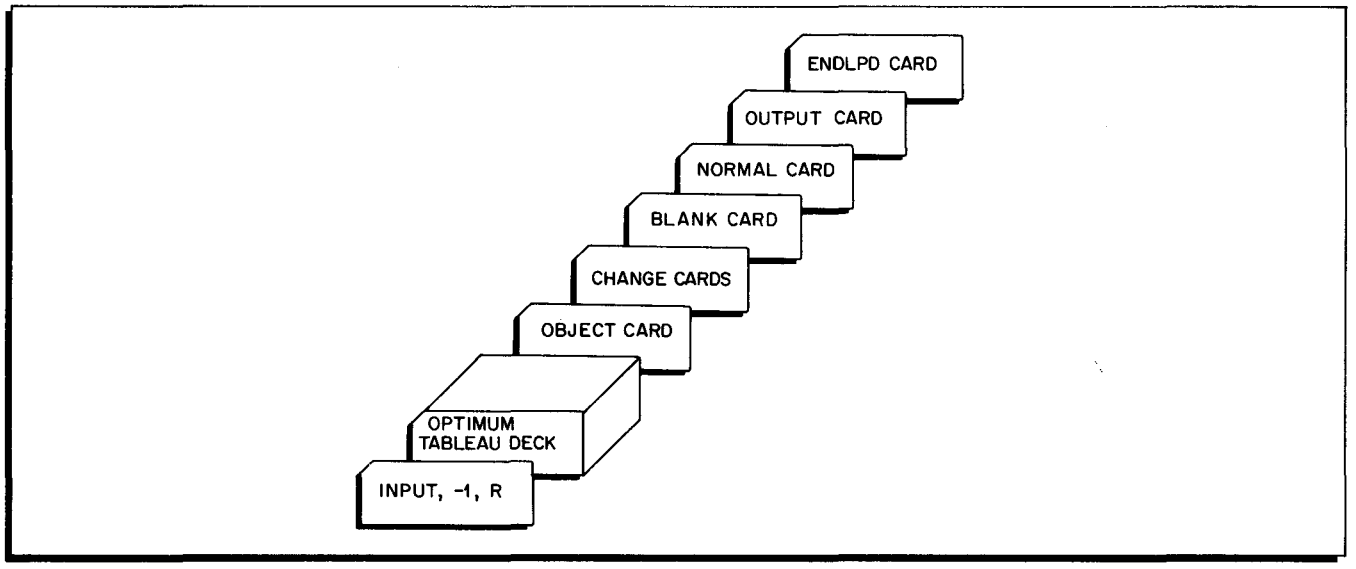


Figure 6-5. Input Deck for Objective Coefficient Change

The OBJECT printout in Figure 6-6 indicates that five of the six requested changes were executed. The fourth change indicates that if the value of the objective coefficient is changed from 4.6 to 4.8, the solution is still optimal and the objective value is about 5.7. The fifth change indicates that if the value of the coefficient is changed from 4.8 to 5.0, the solution is not optimal. Thus, the critical value of the coefficient lies in the range 4.8 to 5.0.

At the NOT OPTIMAL halt, the operator sets switch 4 ON and presses RUN, calling NORMAL to attempt to reoptimize the problem. This new problem differs from the original problem only in that the objective coefficient of CRISPI is 5.0 rather than 4.0.

In one iteration, NORMAL reoptimizes the problem by replacing basis vector SLP003 with vector CHORTL (compare Figure 6-3). The new optimal solution indicates that three cereals must be mixed to obtain an optimal solution. The minimal cost of the mixture is 5.8 rather than the original 5.2.

Right-Hand Side Change - 1

Suppose that the Wholesome Cereal Company wants to know how low they can reduce the calory content of the cereal mix and still have the same optimal basis that was obtained in the original problem. Suppose also that they want to know what change, if any, can be made to the optimal basis to permit even further lowering of the calory content. The agenda RIGHT and DUAL together with the optimum tableau punched in the original problem are used to obtain this information.

```
INPUT,-1,R
PRDATA      BFP
DATA LOADED

OBJECT

VARIABLE     OLD COEFF.     NEW COEFF.
CRISPI          4.000000          4.200000
STILL OPTIMAL
COST           -5.385366

VARIABLE     OLD COEFF.     NEW COEFF.
CRISPI          4.200000          4.400000
STILL OPTIMAL
COST           -5.502439

VARIABLE     OLD COEFF.     NEW COEFF.
CRISPI          4.400000          4.600000
STILL OPTIMAL
COST           -5.619513

VARIABLE     OLD COEFF.     NEW COEFF.
CRISPI          4.600000          4.800000
STILL OPTIMAL
COST           -5.736586

VARIABLE     OLD COEFF.     NEW COEFF.
CRISPI          4.800000          5.000000
NOT OPTIMAL
COST           -5.853659

NORMAL
ITERS.       OBJECTIVE     VAR.IN
FEASIBLE
    1         -5.836364     CHORTL
OPTIMUM

OUTPUT
COST           -5.836364

BASIS VAR.      VALUE
CRISPI           .418183
CHORTL           .236362
CRACKL           .290910

VARIABLE         D/J
SLP003           .545458
SLN004           .890910
CRUNCH           .436364
DUMMY          99.978188

ENDLPD
```

Figure 6-6. Output for Objective Coefficient Change and Reoptimization

The set of change cards in Figure 6-7 progressively decrements the right-hand-side element 150.0 by 10.0 until either the current value of the coefficient is 90.0 or the solution becomes infeasible. If the solution becomes infeasible, the program halts before all the changes

have been implemented (see Table 4-4). The operator is instructed then to set SENSE switch 4 ON, so that the remaining change cards are skipped and the agendum DUAL is called. The change cards in Figure 6-7 are inserted in the input deck as illustrated in Figure 6-8.

The RIGHT printout in Figure 6-9 indicates that five of the six requested changes were executed. The fourth change indicates that, if the value of the right-hand-side element is changed from 120 to 110, the solution is still feasible (and optimal) and the objective value is 4.87. The fifth change indicates that, if the element is changed from 110 to 100, the solution is not feasible. Thus, the critical value lies in the range 110 to 100.

PROBLEM BREAKFAST FOOD PROBLEM DATE __/__/__ PAGE ____ OF ____

	INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
	1 6	7	12	13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1	RIGHT									
2		DUMMY		CALORY		15Ø	.	Ø	14Ø.Ø 1ØØ.Ø	
3		DUMMY		CALORY		14Ø	.	Ø	13Ø.Ø 1ØØ.Ø	
4		DUMMY		CALORY		13Ø	.	Ø	12Ø.Ø 1ØØ.Ø	
5		DUMMY		CALORY		12Ø	.	Ø	11Ø.Ø 1ØØ.Ø	
6		DUMMY		CALORY		11Ø	.	Ø	1ØØ.Ø 1ØØ.Ø	
7		DUMMY		CALORY		1ØØ	.	Ø	9Ø.Ø 1ØØ.Ø	
8										
9										
10										
11										

Figure 6-7. RIGHT Change Cards

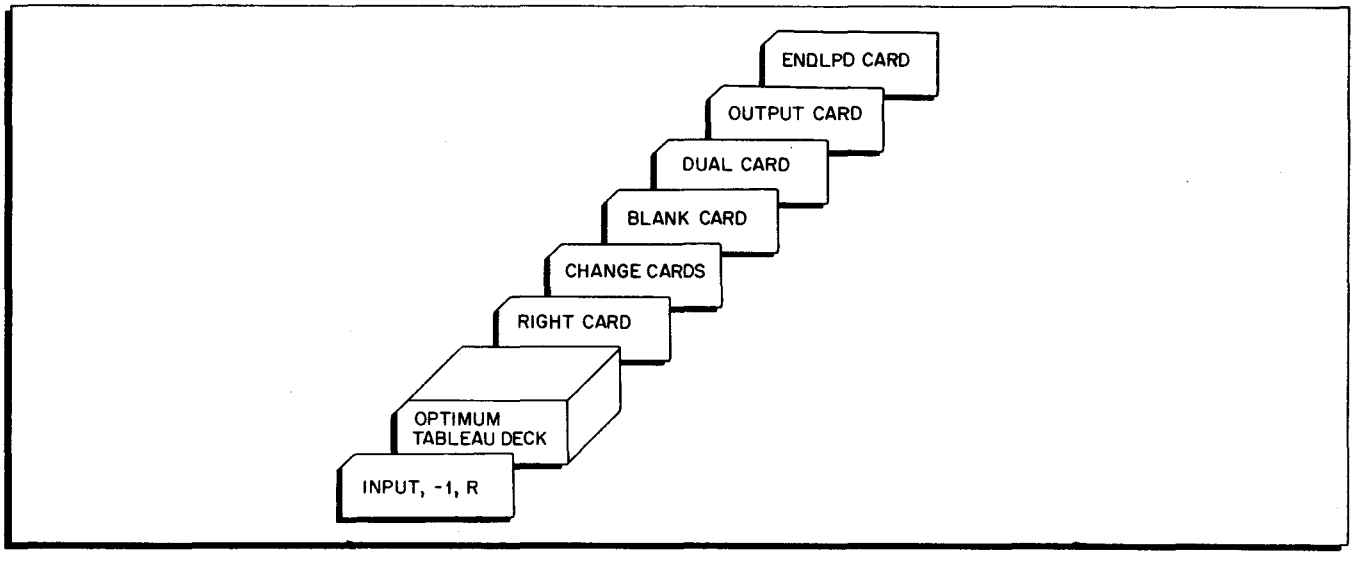


Figure 6-8. Input Deck for Right-Hand-Side Element Change

```
INPUT,-1,R
PRDATA        BFP
DATA LOADED

RIGHT

          ROW      OLD REQ.          NEW REQ.
DUMMY CALORY      150.000000        140.000000    100.000000
OPTIMAL
COST                -5.170793

          ROW      OLD REQ.          NEW REQ.
DUMMY CALORY      140.000000        130.000000    100.000000
OPTIMAL
COST                -5.073293

          ROW      OLD REQ.          NEW REQ.
DUMMY CALORY      130.000000        120.000000    100.000000
OPTIMAL
COST                -4.975793

          ROW      OLD REQ.          NEW REQ.
DUMMY CALORY      120.000000        110.000000    100.000000
OPTIMAL
COST                -4.878293

          ROW      OLD REQ.          NEW REQ.
DUMMY CALORY      110.000000        100.000000    100.000000
NOT FEASIBLE
COST                -4.780793

DUAL
ITERS.       OBJECTIVE                VAR.IN
INFEASIBLE

OUTPUT
COST                -4.780793

BASIS VAR.          VALUE
CRISPI             -.024384
SLP003              .019507
CRACKL              .609754

VARIABLE            D/J
SLN004             1.268293
CRUNCH              .560976
CHORTL              .634146
DUMMY             99.990250

ENDLPD
```

Figure 6-9. Output for Right-Hand-Side Element Change

At the NOT FEASIBLE pause, the operator sets SENSE switch 4 ON and presses RUN, calling DUAL to attempt to optimize the new problem. The new problem differs from the original problem only in that the right-hand-side element in row CALORY is 100 rather than 150. The printout INFEASIBLE indicates that the new problem has no feasible solution. The solution printed by OUTPUT indicates that vector CRISPI is at a negative level (-0.02) in the solution vector.

This result indicates that the restriction of 100 units on the amount of calories in the mix is too stringent. The limit on calory content can be reduced as low as 110.0 and the optimal solution obtained in the original problem will still be optimal, but a limit of 100.0 on the calory content is too low. With this low limit on calory content, even a change of basis would not help because the problem is infeasible.

Right-Hand-Side Change - 2

Suppose now that the Wholesome Cereal Company wants to know how they can raise the calory content of the cereal mix and still have the same optimal basis that was obtained in the original problem. The agenda RIGHT and DUAL together with the optimum tableau punched in the original problem are used to determine the upper limit on calories.

The change cards are illustrated in Figure 6-10. These cards are inserted in the input deck again, as illustrated in Figure 6-8.

The RIGHT printout in Figure 6-11 indicates that the limit on calories can be raised to 225.0 units without changing the optimal basis. The next change indicates that if the requirement is changed from 225.0 to 230.0, the basis is no longer feasible.

At the NOT FEASIBLE pause, the operator sets SENSE switch 4 ON and presses RUN, calling DUAL to attempt to optimize the new problem. This new problem differs from the original problem only in that the right-hand-side element in row CALORY is 230.0 rather than 150.0. In this case the DUAL agendum is able to modify the basis and determine a new basis which satisfies the new requirement on CALORY and is optimal. The agendum OUTPUT is then called and prints the optimal solution to the new problem.

PROBLEM BREAKFAST FOOD PROBLEM DATE / / PAGE OF

	INDICATOR FIELD	COLUMN NAME	ROW TYPE	ROW NAME	VALUE OF ELEMENT SIGN	INTEGRAL PART	DEC. PT.	FRACTIONAL PART	REMARKS	CARD IDENTIFICATION
	1 6	7	12	13 18	19	20 23	24	25 30	31 36 37 42 43 48 49 54 55 60 61 66 67 72	73 80
1	RIGHT									
2		DUMMY		CALORY		150	.	0	220.0 100.0	
3		DUMMY		CALORY		220	.	0	225.0 100.0	
4		DUMMY		CALORY		225	.	0	230.0 100.0	
5		DUMMY		CALORY		230	.	0	235.0 100.0	
6										
7										
8										
9										
10										

Figure 6-10. RIGHT Change Cards

```
INPUT, -1, R
PRDATA        BFP
DATA LOADED

RIGHT

          ROW       OLD REQ.    NEW REQ.
DUMMY CALORY       150.000000  220.000000    100.000000
OPTIMAL
COST                -5.950793

          ROW       OLD REQ.    NEW REQ.
DUMMY CALORY       220.000000  225.000000    100.000000
OPTIMAL
COST                -5.999543

          ROW       OLD REQ.    NEW REQ.
DUMMY CALORY       225.000000  230.000000    100.000000
NOT FEASIBLE
COST                -6.048293

DUAL
ITERS.              OBJECTIVE      VAR.IN
   1                 -6.132831     SLN004
OPTIMUM

OUTPUT
COST                -6.132831

BASIS VAR.           VALUE
CRISPI              1.533329
SLP003               .046675
SLN004               .066655

VARIABLE             D/J
CRUNCH              3.266666
CRACKL              3.466664
CHORTL              1.733332
DUMMY              99.973340

ENDLPD
```

Figure 6-11. Right-Hand-Side Change and Reoptimization

HONEYWELL EDP TECHNICAL PUBLICATIONS
USERS' REMARKS FORM

TITLE: SERIES 200
LINEAR PROGRAMMING PACKAGE D
SOFTWARE MANUAL

DATED: MARCH, 1966

FILE NO: 123.8305.001D.0-276

ERRORS NOTED:

Fold

SUGGESTIONS FOR IMPROVEMENT:

Fold

FROM: NAME _____ DATE _____

COMPANY _____

TITLE _____

ADDRESS _____

_____

Cut Along Line

FIRST CLASS
PERMIT NO. 39531
WELLESLEY HILLS
MASS.

BUSINESS REPLY MAIL

No postage stamp necessary if mailed in the United States

POSTAGE WILL BE PAID BY

HONEYWELL
ELECTRONIC DATA PROCESSING DIVISION
60 WALNUT STREET
WELLESLEY HILLS, MASS. 02181

ATT'N: TECHNICAL COMMUNICATIONS DEPARTMENT

Cut Along Line

Honeywell
ELECTRONIC DATA PROCESSING

CRP