An Introduction to Linear Programming

IBM

Data Processing Application

An Introduction to Linear Programming

IBM

Data Processing Application

PREFACE

The purpose of this manual is to describe linear programming, to show what can be accomplished with it, and to prepare the reader to make intelligent use of a linear programming system on a computer. The presentation covers the entire scope of a linear programming application: problem formulation, computer operations, interpretation of results, and additional information that can be obtained through the use of a complete linear programming system for a computer. There is no attempt at mathematical rigor. Some of the mathematical techniques are presented briefly to indicate what is involved in the computer solution of a problem, but no mathematical background beyond high school algebra is assumed.

The glossary, however, is intended as a comprehensive list of technical terms associated with the solving of linear programming problems, rather than as a list of only those terms used in this manual. Furthermore, the definitions are written in a technical manner for the benefit of those readers who are studying linear programming on a deeper level than that of this introductory manual.

Copies of this and other IBM publications can be obtained through IBM Branch Offices. Address comments concerning the contents of this publication IBM, Technical Publications Department, 112 East Post Road, White Plains, N. Y. 10601

CONTENTS

Chapter 1: Concepts and Examples 1

1.1: Example of Production Capacity Allocation 2
1.2: Example of Feed Blending 7
1.3: Example of Investment Policy . . . 9
1.4: Characteristics of a Linear Programming Application. 13
1.5: Typical Linear Programming Applications 13
Refinery Scheduling 13
Paper Trimming. 14
Production Allocation 14

Chapter 2: Deriving Additional Information About a Solution 15

2.1: The Fundamental "Answer". 15
2.2: Changing Limits: Right-Hand Side Ranges and Marginal Values 16
2.3: Changing Objective Function Coefficients: Profit Ranges 18
2.4: Tradeoffs: Rates of Substitution . . 19
2.5: Sensitivity to Technology 20
2.6: Changing Several Things at Once: Parametric Programming 21

Chapter 3: A Larger Example — Oil Refinery Scheduling 23

3.1: The Physical Situation 23
3.2: The Operating Data 25
Crude Supply 25
Pipe-Still Operations 25
Catalytic-Cracker Operation 25
Gasoline Blending 26
Distillate (heating oil) Blending. . . 26
Fuel Blending 27
3.3: The Objective Function and Constraints 27
Objective Function 27
Material-Balance Constraints . . . 27
Pipe-Still Constraints. 27
Cat-Cracker Constraint 27
Fuel Oil-Blending Constraint 27
Heating Oil-Blending Constraints. . 27
Gasoline-Blending Constraints . . . 28
3.4: Computer Input 29
3.5: The Solution 29
The Optimum Policy 31
Changes in Allocation 31
Marginal Values. 31
Reduced Costs. 31
Sensitivity of Profit to Constraint Coefficients 33
Ranges of Optimum-Policy Variables 33
Marginal Value Ranges 34
3.6: Parametric Programming 34

Chapter 4: The Simplex Method. 37

4.1: Geometrical Background 37
4.2: Algebraic Demonstration 38
4.3: Summary. 40

Glossary. 41

Bibliography. 63

Index 64

CHAPTER 1: CONCEPTS AND EXAMPLES

Linear programming is a mathematical technique for determining the optimum allocation of resources (such as capital, raw materials, manpower, plant or other facilities) to obtain a particular objective (such as minimum cost or maximum profit) when there are alternative uses for the resources. Linear programming can also be used to analyze the economics of alternate availability of resources, alternate objectives, and so on.

A few brief examples may serve to indicate more concretely what can be achieved with linear programming:

1. A manufacturer makes a number of different products. Each product uses certain production resources, each of which is available in a limited amount. The manufacturer knows how much profit he makes from each product. How much of each product should he produce in order to make the maximum total profit?

2. A producer of livestock feed is required to provide certain amounts of various nutritional elements in each sack of feed. He can obtain the various elements from different grains and supplements, and he knows the cost of each. What combination of grains and supplements should he use to meet the requirements at least cost?

3. The manager of an oil refinery is considering expanding the production of his plant by adding capacity at some point in the refining process. Of the many different processes involved in refining, which one should have its capacity increased so as to bring the greatest return on the capital expenditure?

4. Another manufacturer uses a large number of raw materials in the production of a line of products. The prices of his raw materials are subject to market fluctuations, and in some cases there are significant price breaks for large orders. He has a choice of which raw materials to use. One of the materials he is not using at the moment might profitably be used if the price were lower. How much would the price have to drop before he could make a greater profit by using it instead of something else?

5. This same manufacturer is faced with another problem: one of the raw materials he is now using will be unavailable for a while because of a fire at the plant of one of his suppliers. Of the various alternative raw materials that he could use to replace it, which one will cause the least decrease in profit, considering that the introduction of a different material may change the mixture of other materials he uses?

These examples emphasize the importance of linear programming. When a large number of interrelated choices exist, the best choice may be far from obvious. An intuitive solution may never uncover the best approach, and there is seldom any guarantee that what appears to be a fairly good policy is really the best.

Such problems often involve large amounts of money. A rational approach to the problems requires:

- A systematic way to represent the goal, or _objective_, of the system under study.
- A systematic way to describe the limitations, or _constraints_, under which the system must operate -- for instance, the limited amount of production resources in the first example given, and the minimum nutritional requirements in the second example.
- Some way to arrive at one policy out of the many possibilities, and to be sure that it is the best.
- Some way to explore the ramifications of changes in the stated problem (assuming, of course, that the best, or _optimum_, policy for the original objective and the original constraints has been determined).

In the third example, the refinery manager needed to know which capacity limitation could most profitably be relaxed. In the fourth example, the manufacturer needed to know the effect on his best policy of a change in the costs of materials. In the fifth example, the manufacturer needed to know how the nonavailability of one raw material would affect his profit and how to choose a new policy that would minimize this effect.

"Linear programming", as the term is used today, includes the formulation of the problem, the solution (finding the optimum policy), and the exploration of the effects of changes. Such large-scale work almost always involves a computer, for any problem small enough to be done "by hand" could probably be solved without resorting to linear programming. Because computers are so important for large-scale work, special application systems have been developed to assist in the practical use of linear programming methods. Thus the user is not required to write the complex program of instructions needed to tell the computer how to solve linear programming problems. The user is required, however, to formulate his problem in the proper form for a linear programming solution and to prepare the necessary input data for the computer. The solution (best policy) is then found by the computer.

Most systems also contain at least some features to allow investigation of the effects of problem changes of various sorts. The user must also know enough about linear programming to interpret the results printed by the computer, and to decide what changes should be explored.

Ordinarily, the user does not need to know a great deal about how the computer finds the optimum policy or how it arrives at the effect of changes. He does, however, need to know the elements of these methods in order to formulate his problem most effectively and to interpret the results intelligently.

The three examples that follow are designed to serve four essential purposes for the user who wants to employ linear programming intelligently but who needs only a minimum knowledge of the methods of solution.

1. The examples introduce the characteristics of a problem that can be handled with linear programming. The technique is not the universal remedy for all management problems; it is important to know not only what can be done with linear programming, but also what cannot.

2. They introduce the idea of problem formulation. Linear programming requires that the problem be stated in a specific manner (in terms of the objective and the constraints mentioned earlier). There is usually a certain amount of work involved in transforming a problem, as initially stated, into the form required for linear programming solution.

3. They indicate the method of computation of the optimum policy.

4. They reveal some of the information that can be derived from the solution to a linear programming problem, and they aid in interpreting the results.

Following these examples, Chapter 2 will discuss the types of changes in the problem that can be explored. In Chapter 3 a representative problem is carried from formulation, through preparation of the computer input, interpretation of initial results, and the change information presented by the computer system. For the reader who wishes to know more about solution methods, Chapter 4 is an introduction to the simplex method, which is the basis of most linear programming systems on computers.

1.1: EXAMPLE OF PRODUCTION CAPACITY ALLOCATION

We can get an idea of the characteristics of a problem that can be attacked effectively with linear programming by considering a simplified example. The illustration is realistic to the extent that businessmen do face such problems, but the problem presented is much smaller than the linear programming problems that are handled with computer systems.

A small machine shop manufactures two models, standard and deluxe, of an unspecified product. Each standard model requires four hours of grinding and two hours of polishing; each deluxe model requires two hours of grinding and five hours of polishing. The manufacturer has two grinders and three polishers; in his 40-hour week, therefore, he has 80 hours of grinding capacity and 120 hours of polishing capacity. He makes a profit of \$3 on each standard model and \$4 on each deluxe model. He can sell all he can make of both.

How should the manufacturer allocate his production capacity to standard and deluxe models; that is, how many of each model should he make in order to maximize his profit?

Let us begin by converting this problem statement into a mathematical form. Assign the symbol S to the number of standard models manufactured in a week, and the symbol D to the number of deluxe models. The profit from making S standard models and D deluxe models in a week, then, is

$$3S + 4D \text{ dollars}$$

For instance, if five standard models ($S = 5$) and seven deluxe models ($D = 7$) are built in a week, the profit is $(3 \times 5) + (4 \times 7) = \$43$; if the manufacturer could make 25 standard models and 20 deluxe models, the profit would be \$155.

How can we express the restrictions on machine capacity? The manufacture of each standard uses four hours of grinding. Making S standard models therefore uses 4S hours. Similarly, the manufacture of D deluxe models uses 2D hours of grinding time, since the manufacture of one deluxe uses two hours. The total number of hours of grinding capacity used in a week, therefore, is

$$4S + 2D$$

We said previously that 80 hours of grinding time was available, so we might be tempted to write

$$4S + 2D = 80$$

This would not be correct, however, because at this point we have no assurance that the greatest profit lies in using all the grinding time. All we know is that the total hours of grinding time must not exceed

80. "Must not exceed" can also be expressed as "must be less than or equal to", a more convenient expression. The mathematical symbol for "less than or equal to" is $\leq$. The correct formula for the restriction on grinding capacity would be

$$4S + 2D \leq 80 \text{ hours}$$

In the same manner, we arrive at the limitation on polisher capacity:

$$2S + 5D \leq 120 \text{ hours}$$

In view of the higher profit on the deluxe models, one might suggest that the optimum policy would be to make as many deluxe models as possible and forget the standard models. Let us calculate how many deluxe models alone could be made, and record the corresponding profit for future reference. The limitation on grinder time provides that two times the number of deluxe models must not exceed 80, and the limitation on polisher time provides that five times the number of deluxe models must not exceed 120. Grinder capacity permits 40 deluxe models to be made; polisher capacity permits no more than 24 to be made. Therefore, 24 is the maximum number of deluxe models that could be made, even though this policy would consume only 48 hours of grinder time out of the 80 available. The profit with this policy is $96, since there is a $4 profit on each of the 24 deluxe models.

Let us now explore what the constraints (restrictions, or limitations) on machine time mean in geometrical terms. We shall draw a graph on which the vertical axis represents S (the number of standard models made in a week) and on which the horizontal axis represents D (the number of deluxe models). Now remember the first constraint that on grinder capacity $4S + 2D \leq 80$ hours. Let us consider only the "equal" part of the symbol for the moment, and rearrange the statement of the constraint:

$$4S + 2D = 80$$

so $\quad 2D = 80 - 4S$

and $\quad D = 40 - 2S$

This equation lets us draw up a table of a few values of S and the corresponding values of D:

S	D
0	40
5	30
10	20
15	10
20	0

If we plot these points on a graph, the results would be those shown in Figure 1.

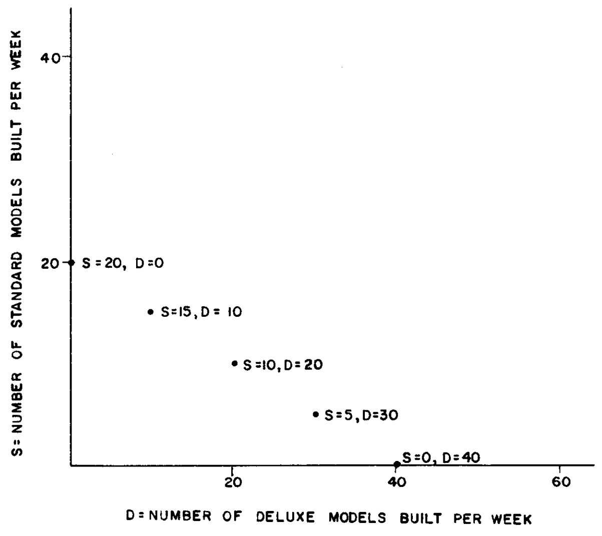


Figure 1. Five points that satisfy the equation $4S + 2D = 80$

These five points lie on a line. We say that the equation $4S + 2D = 80$ is linear. In geometrical terms, this means that all points that satisfy the equation lie on a straight line. In algebraic terms, it means that S and D both appear in the equation multiplied only by a constant coefficient; that is, they are not squared, multiplied together, and so on.

Reviewing what the graph of our equation means, we have a straight line which "represents" the equation in the sense that any point on the line corresponds to some specific combination of values of S and D, and these values satisfy the equation. With this meaning in mind, we can plot the equation of the constraint under consideration simply as a line, without identifying any of the specific points, as in Figure 2.

But our constraint is not an equation; it is an inequality: $4S + 2D \leq 80$ hours. Graphically, this means that the inequality is represented in the

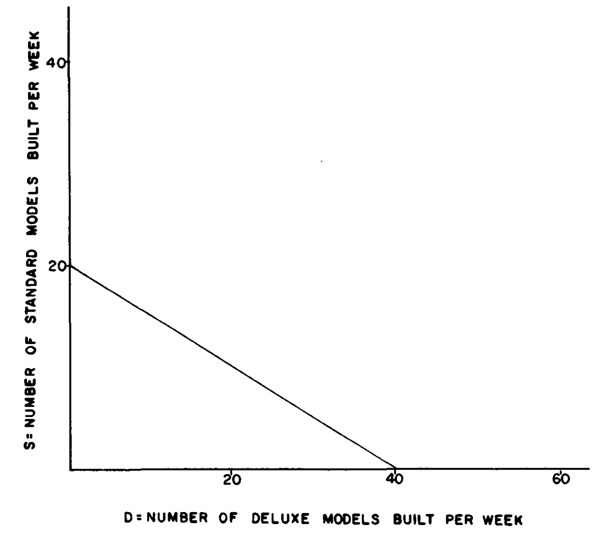


Figure 2. Graph of the equation 4S + 2D = 80

picture by any point on the line or below it. We can make this explicit by shading in the region that is covered by the inequality (see Figure 3).

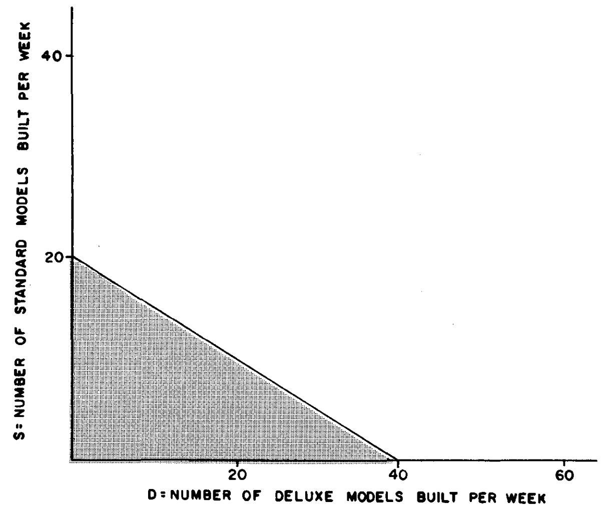


Figure 3. Graphical representation of the inequality 4S + 2D ≤ 80. Any point on the line or in the shaded region satisfies the inequality.

It is important to realize that any point on the line or below it still represents some specific combination of a value of S and a value of D; points below the line will be the "less than" part of the "less than or equal to" symbol.

In Figure 4 we also plot the constraint on the polishing capacity (2S + 5D ≤ 120), but we now have shaded only the part of the graph that is "below" both lines.

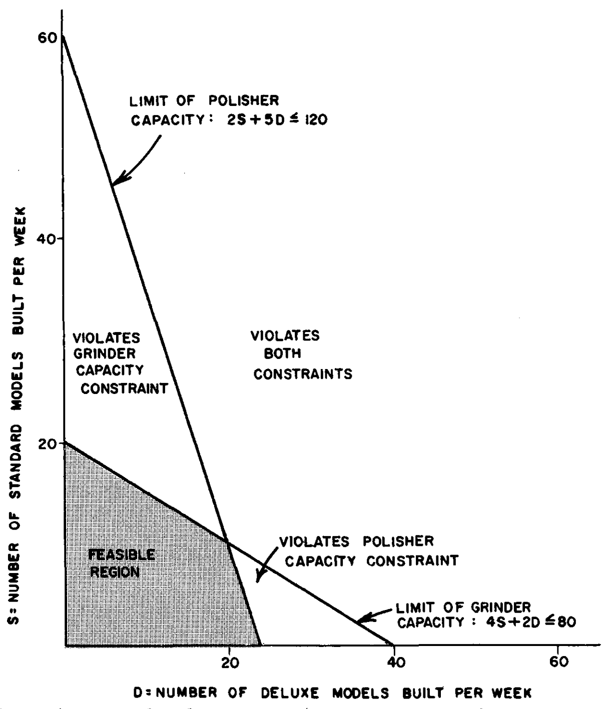


Figure 4. Graphical representation of both constraints. The small shaded region violates neither constraint and is called the feasible region.

In Figure 4 the shaded region reflects the effect of both constraints. The parts of the graph within the triangles violate one or the other of the constraints; the part outside both lines violates both; the shaded region violates neither, and is therefore called feasible. This means that any point in the feasible region represents a combination of so many standard models and so many deluxe models, the combination being possible (feasible) with due regard for the amount of machine time available. From now on, we shall display only the feasible region on such graphs, since this area is all that really interests us.

How would we take into account the profit? We already know that the profit is 3S + 4D dollars; in other words, any point anywhere on the graph represents some specific profit. Some of these points will, of course, lie outside the feasible region and therefore will not represent possibilities for the best policy. What we are looking for is the point in the feasible region that represents the policy with the greatest profit. (In fact, the essence of the linear programming computation is the search for this point.) In a computer the location of the optimum point is reached by algebraic methods, rather than by graphical ones. We shall sketch these methods in section 1.3 of this chapter, using the third example, "Investment Policy", and in Chapter 4 on the simplex method.

Here, we will see what can be done by graphical methods. We know that the profit for any particular point is 3S + 4D dollars. What would happen if we set this expression equal to some specific profit level, say $60? The answer is that we would have a linear equation, which we could plot as the broken line in figure 5. This line includes such points as 20 standard and zero deluxe models, 16 standard and three deluxe models, 12 standard and six deluxe models, eight standard and nine deluxe models, or zero standard and 15 deluxe models.

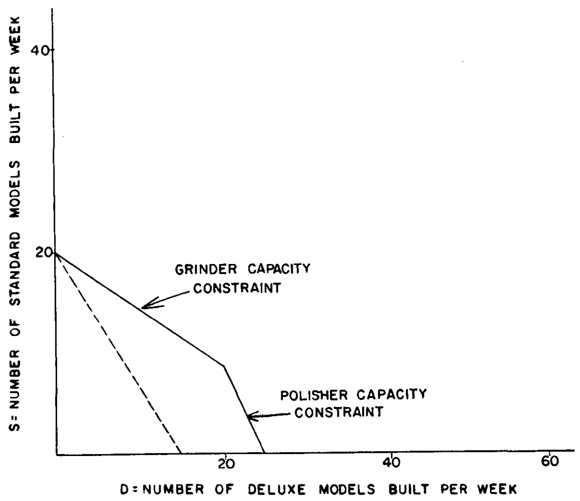


Figure 5. Graph of the constraints, the feasible region, and the profit line for $60: 3S + 4D = 60

All points on this profit line are feasible; there is no violation of either machine capacity constraint anywhere on this line. Any point on this line thus represents a feasible policy, and all such points would return a $60 profit.

This policy, however, is not the best one. If, for instance, we try plotting the profit line 3S + 4D = $96, we find proof that there is a better policy (see Figure 6).

Not all points on this profit line are feasible. We cannot make 36 standard models and zero deluxe models, even though this point is on the line, because there is not enough grinder capacity to permit it. Some points on the line are feasible, however; for instance, the combination of twelve standard models and 15 deluxe models violates neither constraint and is therefore a feasible combination.

At $60, all points on the profit line were feasible; at $96 some points are feasible. The question that interests us now is the extent to which we can go and

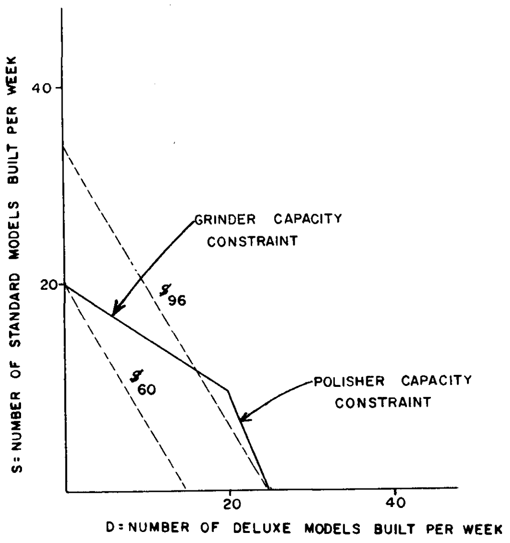


Figure 6. The constraints and the $60 and $96 profit lines

still have at least one feasible point on the profit line. This point represents the optimum policy. To find it is the fundamental goal of linear programming, and we shall see that there are ways of working with algebra to arrive at it.

Staying with the graphical approach for now, we notice in Figure 6 that the two profit lines are parallel, an observation that can be proved mathematically. Furthermore, all profit lines will be parallel for the problem as stated. This suggests a method of attack; look for the line parallel to the ones we have already drawn that goes as "far out" as possible but that still keeps one point in the feasible region. This will be the line that goes through the "corner" of the feasible region where the two constraint lines meet. It is shown in Figure 7. This is the profit line for $110, made from 10 standard models and 20 deluxe models.

It should be clear from Figure 7 that striving for any profit greater than $110 would involve a profit line with all points outside the feasible region. We have therefore established that $110 is the best profit that can be made under the original objective and constraints.

We see that getting the maximum profit possible does require manufacture of some of the less profitable standard models, because that policy makes the best use of machine capacity. Thus one of the most important things this example shows is that maximizing profit may require a tradeoff between exclusive manufacture of the most profitable item and the full use of facilities. In other words, the manufacturer's most profitable policy was determined not

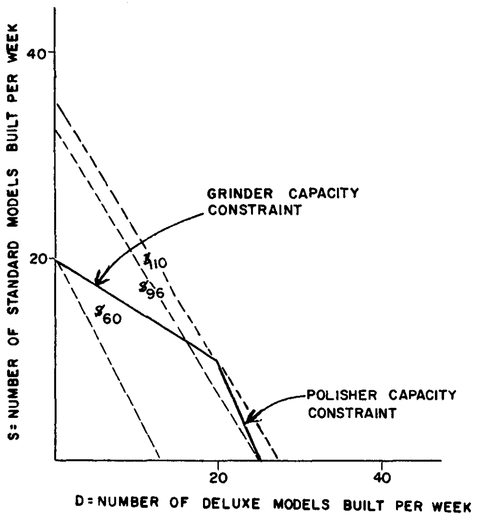


Figure 7. The constraints and the \$60, \$96 and \$110 profit lines

only by his unit profit on each product, but also by the constraints on his production capacity.

The problem of maximizing profit is representative of a large class of problems that can be solved by linear programming. Such problems are characterized by the allocation of limited resources to products of different profitability.

It need not always happen, however, that the most profitable policy will require a combination of products. Suppose that the profit on each standard model was still \$3, but that each deluxe model brought in \$9 profit. When we search for the line with the greatest profit that still has at least one feasible point, we get the situation shown in Figure 8.

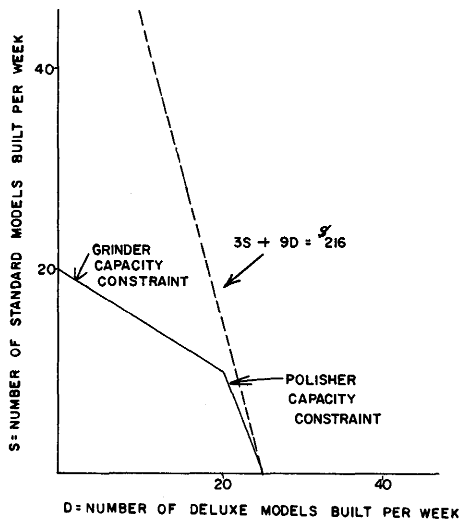


Figure 8. The constraints and the \$216 profit line when the profit on a standard model is \$3 and the profit on a deluxe model is \$9

Here the profit line for the greatest profit that contains one feasible point still intersects the feasible region at one of its vertices (corners), but now it is a vertex formed of one of the constraint lines and one of the axes. This means that it is indeed most profitable to make all deluxe models (24 in this instance) and zero standard models, even though this policy will leave 32 hours of grinder time unused.

It will be instructive to explore what the best policy would be if each standard model brought in \$6 profit and each deluxe model brought in \$3 profit. When we seek the greatest profit line that contains a feasible point, we get a profit line that coincides with one of the constraint lines, as in Figure 9.

The manufacturer has several choices of the best policy. He can make 20 standard and zero deluxe models, 15 standard and ten deluxe models, ten

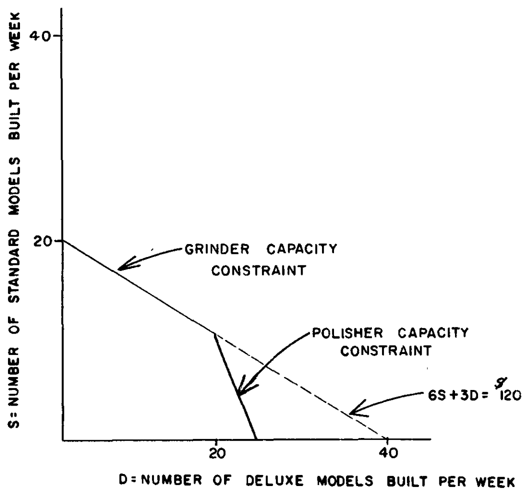


Figure 9 The constraints and the best profit line when the profit on each standard model is \$6 and the profit on each deluxe model is \$3

standard and 20 deluxe models, since each combination brings in the same profit of \$120. Any of these combinations can be taken as the "best" policy; they are all equally good. This type of solution occasionally appears in practical applications.

In each of the preceding examples, we have taken it for granted that S and D are either zero or positive; there is no way to make a negative quantity of anything. We reject such quantities without thinking; however, when we discuss the algebraic method of finding the maximum profit line subject to constraints, we will have to include an explicit requirement that no variable be negative, because ordinary algebraic methods do permit negative numbers. This nonnegativity requirement becomes an integral part of the algebraic approach, but a person using a computer program based on such an algebraic method does not have to go to any extra effort because of the requirement.

We are now in a position to state the original example in the standard form for a problem that can be handled with linear programming.

(1)	Maximize	$3S + 4D$	the objective function (profit)
(2)	Subject to	$4S + 2D \leq 80$ $2S + 5D \leq 120$	the problem constraints
(3)	And subject to	$S \geq 0,\ D \geq 0$	the nonnegativity constraints

We shall refer frequently to the parts of the mathematical statement of a problem for linear programming solution. Figure 10 shows the terminology that will be used in this manual. Note that the objective function has been set equal to Z as a convenience in later operations. Observe that a row refers to one constraint or to the objective function; one column refers to one activity or to the right-hand side. The values of the variables are called activity levels.

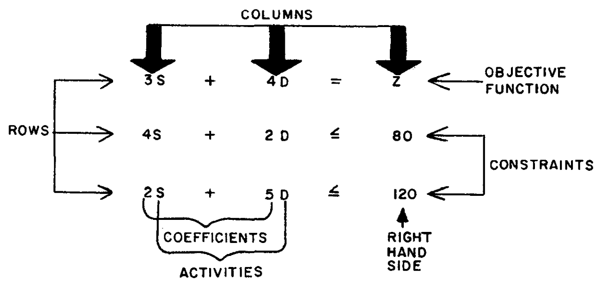


Figure 10. The terminology used in describing the normal formulation of a linear programming problem

We shall see that the standard form of a problem suitable for linear programming is characteristic, although there are important variations: we often want to minimize a cost rather than maximize a profit, and we often have constraints that state a "greater than or equal to" condition rather than a "less than or equal to" condition.

1.2: EXAMPLE OF FEED BLENDING

A poultry farmer needs to supplement the vitamins in the feed he buys. He is considering two products, each of which contains the four vitamins required, but in differing amounts. Naturally, he wants to meet (or exceed) the minimum vitamin requirements at least cost. Should he buy one product or the other, or should he mix the two? The facts are summarized in the table below.

	Product 1	Product 2
Cost per ounce	3 cents	4 cents
Vitamin 1 per ounce	5 units	25 units
Vitamin 2 per ounce	25 units	10 units
Vitamin 3 per ounce	10 units	10 units
Vitamin 4 per ounce	35 units	20 units

The farmer must provide, per hundred pounds of feed, at least 50 units of vitamin 1; 100 units of vitamin 2; 60 units of vitamin 3; and 180 units of vitamin 4.

Let us state the problem in the standard form described near the end of the previous example.

The objective in this case is to minimize the cost of obtaining the required vitamins. Let P1 represent the number of ounces of product 1 purchased, and let P2 represent the number of ounces of product 2. Then the objective is to minimize

$$3P1 + 4P2$$

The constraints this time are the minimum requirements. Each of the four vitamins can be obtained in varying amounts from either product. Whatever combination of products is bought, the sum of the units of a given vitamin in the two must equal or exceed the minimum requirement for that vitamin. We thus get the following four "greater than or equal to" constraints:

$$5P1 + 25P2 \geq 50$$

$$25P1 + 10P2 \geq 100$$

$$10P1 + 10P2 \geq 60$$

$$35P1 + 20P2 \geq 180$$

As always, we have the nonnegativity requirement on the variables, in this case the number of ounces of P1 and of P2 bought.

How does this problem compare with the earlier one? Before, we were trying to maximize profit; now we want to minimize cost. Before, we had two activities (manufacture of standard and deluxe models); now, we also have two activities (buying of two types of product). Before, there were two constraints; now there are four. The nonnegativity requirement on each activity level never changes.

The situation is graphed in Figure 11.

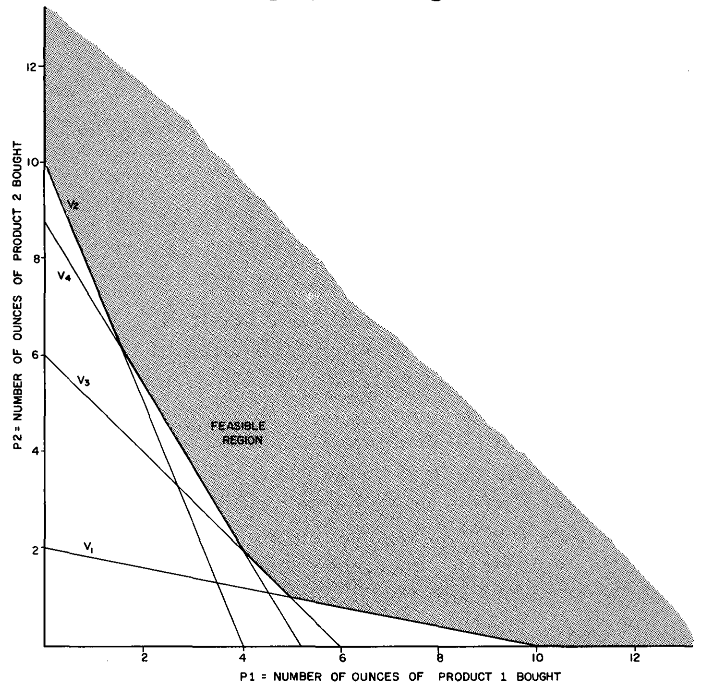


Figure 11. Graph of the constraints in the feed additive problem

The feasible region in this case is on the right-hand side of the lines, since the constraints are all "greater than or equal to"; there is no restriction on the amount by which the minimum can be exceeded. The feasible region consists of all combinations of products that do not violate any of the constraints.

In Figure 12 we have shown only that part of each constraint line that is on the border of the feasible region, and we have plotted four total-cost lines.

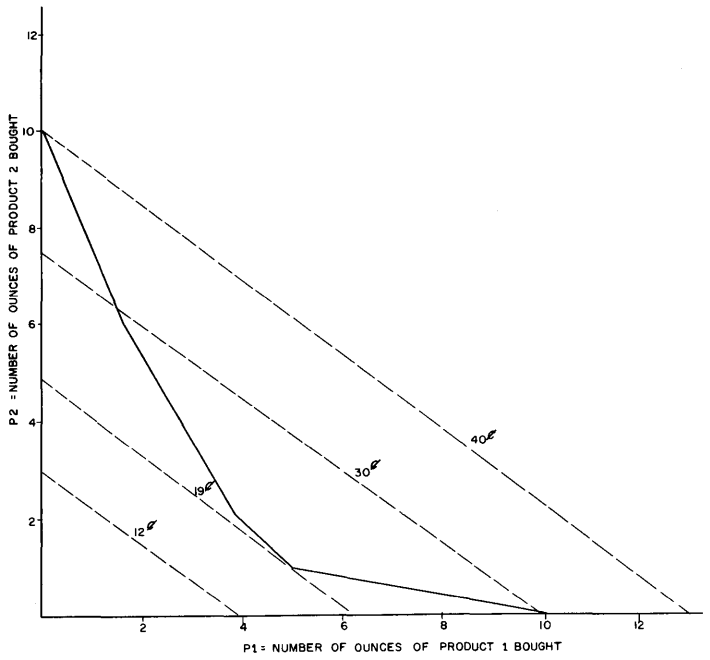


Figure 12. Graph of the feed additive constraints and the total cost lines for 12¢, 19¢, 30¢ and 40¢

We see that the 12¢ line is not feasible anywhere and that the 40¢ line is feasible everywhere. Since the 19¢ line is the lowest-cost line that touches the feasible region, there is no possibility of getting the required vitamin content for less than 19¢. This corresponds to five ounces of P1 and one ounce of P2, which minimizes the cost and still satisfies all four constraints.

One of the most valuable features of the methods used to solve general linear programming problems is that the methods guarantee that the optimum solution is the best. We have, of course, not proved this fact by the graphical demonstrations here, but the procedures used in linear programming guarantee optimality.

It might be interesting to see what the cost would be if the poultry farmer bought only one type of product. These numbers can be derived rather simply from the graph: they are the number of ounces at the points where the feasible region intersects the P1 and P2 axes. It happens, by coincidence, that it would take ten ounces either way. Ten ounces of P1 would satisfy exactly the requirement for the first vitamin and would exceed the requirements for the others in varying amounts; the cost would be 30¢. Ten ounces of P2 would satisfy exactly the requirement for the second vitamin and would cost 40¢.

Note that once again the optimum occurred at a vertex of the feasible region.

1.3: EXAMPLE OF INVESTMENT POLICY

Let us now turn to a rather different type of problem. Because there are six activities, in this example, graphing (which is limited to two dimensions and therefore to two activities) cannot be employed. By deliberate choice, the optimum policy is rather obvious; the point of this example is to demonstrate an algebraic technique that will guarantee finding the optimum, even when the first guess at the optimum is far from accurate.

A man has $1,000 to invest. He has chosen to invest his money in some combination of municipal bonds, preferred stock, and common stock; for each, there are two candidates, making a total of six activities. The following table shows the yields and the symbols for the amount to be invested in each of the six possible activities.

Type:	Bonds		Preferred		Common	
Yield:	3%	3-1/2%	4%	4-1/2%	5%	5-1/2%
Symbol for amount invested:	B1	B2	P1	P2	C1	C2

After consultation with his financial advisers, the investor decided upon the following restrictions on his investment policy: at least $400 must be invested in bonds; no more than $350 must be invested in preferred stock; and no more than $350 must be invested in common stock. To make the problem statement concrete, his policy would permit placing $350 in the 5% common stock, or $350 in the 5-1/2% common stock, or any combination of the two that added to no more than $350.

Now we begin to convert the problem to the standard form of a linear programming problem statement. Converting the percentages to decimal fractions, the objective is to maximize

$$0.03\ B1 + 0.035\ B2 + 0.04\ P1 + 0.045\ P2 + 0.05\ C1 + 0.055\ C2 = Z$$

As noted near the end of section 1.1, we have given the objective function the symbol Z so that we can use it easily in writing equations.

The constraints this time are a mixture of equalities, "less than", and "greater than" conditions. We begin with an ordinary equation, which states that the sum of the amounts invested must be exactly $1,000:

$$B1 + B2 + P1 + P2 + C1 + C2 = 1,000$$

Next we have the "greater than or equal to" restriction of at least $400 in either or both of the two types of bonds:

$$B1 + B2 \geq 400$$

Finally, we have two "less than or equal to" constraints on the maximum that may be invested in preferred or common stocks:

$$P1 + P2 \leq 350$$

$$C1 + C2 \leq 350$$

Because we are able to work more easily with equations than with inequalities, we therefore convert the three inequalities into equations by a simple technique.

Consider the second inequality, $P1 + P2 \leq 350$. Let us introduce a slack variable, which we call S2, as follows:

$$P1 + P2 + S2 = 350$$

What do "slack" and "variable" imply? Suppose that the optimum policy is to invest $250 in P2 and nothing in P1. Then the slack between the amount actually invested in the preferred stock and the maximum that is permitted would be $100. As we proceed through the calculations, the value of S2 may change before we arrive at the optimum; it is not fixed at any unchangeable value. In other words, it is truly a variable.

Note that with the introduction of a slack variable the result is equivalent to the original inequality. Like any other variable in a linear programming problem, slack variables are required to be zero or positive. Suppose that S2 were zero; then we would have P1 + P2 = 350, which is the "equal to" part of "less than or equal to". On the other hand, if S2 were $100, then P1 + P2 would be less than $350.

In other words, since S2 can vary, P1 + P2 can indeed be less than or equal to \$350. But the sum of P1 and P2 cannot be greater than \$350, for that would require S2 to be negative, a condition which is not permitted.

In the case of the third inequality, introduction of the slack variable S3 leads to the equation:

$$C1 + C2 + S3 = 350$$

We suspect, of course, that the man will invest everything he can in this type; in other words, S3 will probably turn out to be zero, a value which is permitted.

In the case of the first inequality, which is of the "greater than or equal to" type, we must subtract the slack in order to represent the fact that he is permitted to invest more than \$400 in the low-yield bonds--although he will not decide to do so. In other words, S1 will also eventually be zero.

$$B1 + B2 - S1 = 400$$

The minus sign here indicates subtraction, not that S1 is negative. As we have stated, the slack variables must be nonnegative, which means that they must be zero or positive. We are thus able to proceed with algebraic methods that depend on never having a negative variable--slack or otherwise.

Let us rewrite the four constraints as equations with slack variables where necessary. The equations have been numbered for later reference.

$$\begin{array}{llr}
(1) & B1 + B2 + P1 + P2 + C1 + C2 & = 1000 \\
(2) & B1 + B2 \qquad\qquad\qquad - S1 & = 400 \\
(3) & \qquad P1 + P2 \qquad\qquad + S2 & = 350 \\
(4) & \qquad\qquad C1 + C2 \qquad + S3 & = 350
\end{array}$$

Here we have four equations in nine unknowns. There is no one single solution to such a system; there are many sets of positive values of the nine unknowns that will satisfy the equations. Our approach will be to pick five of the variables to be zero, and find values for the other four that satisfy the four equations. We shall then proceed in a systematic way to see whether there are other ways to assign values that will improve the total yield.

The discussion of this process will be made somewhat easier if we first introduce some new terminology. (This terminology will also be useful to the reader in any further study of linear programming methods.)

A solution with just as many variables permitted to be nonzero as there are equations, with all other variables being forced to be zero, is called a <u>basic solution</u>. The variables that are permitted to be nonzero are called <u>basic variables</u>, and the collection of them is called a <u>basis</u>. Thus there must be exactly as many variables in a basis as there are problem constraint equations. If the values picked for the basic variables satisfy the constraints and are nonnegative, they represent a point in the feasible region, and we have a <u>basic feasible solution</u>.

Can we find a basic feasible solution for this example? There are many ways to do so. Here is one: let B2 = \$400, P1 = \$350, C2 = \$250, and S3 = \$100, and all others equal zero. Checking with equations (1) to (4), we see that these values satisfy the constraints. This is by no means the only basic feasible solution. Actually, this one was chosen simply to demonstrate that the algebraic technique we wish to explore does lead to an optimum solution.

Having found any basic feasible solution, we seek to improve the basis; that is, we seek another basic feasible solution which is closely related to the first one but which gives a greater value to the objective function. If the basis can be improved still further, we repeat the process. Eventually, we shall find a basic feasible solution that cannot be improved, and we shall have arrived at the optimum solution.

Let us now use this process of improvement in the algebraic approach. First, express the objective function entirely in terms of nonbasic variables; this will give a good clue as to where to seek improvement in the basis. This is easily enough done by finding expressions for the basic variables in terms of nonbasic ones, and by substituting in the objective function.

We begin by solving the four equations (1) to (4) for the basic variables in terms of nonbasic variables only. First we solve equation (2) for B2:

$$B2 = 400 + S1 - B1$$

Next, solve equation (3) for P1:

$$P1 = 350 - P2 - S2$$

Finding an expression for C2 will be more difficult, since all equations involving C2 also involve another basic variable. There are completely general methods for doing this; in this example we shall take advantage of the characteristics of the system of equations to do it simply. First, subtract equation (2) from equation (1):

$$P1 + P2 + C1 + C2 + S1 = 600$$

Now subtract equation (3) from the above:

$$C1 + C2 + S1 - S2 = 250$$

Now transfer everything except C2 to the right-hand side to get an expression for C2:

$$C2 = 250 - C1 - S1 + S2$$

A similar scheme leads to an expression for S3. Subtract the original equations (2), (3) and (4) from equation (1) in turn, and rearrange:

$$S3 = 100 + S1 - S2$$

Now we can substitute these results into the objective function:

$$0.03\ B1 + 0.035\ (400 + S1 - B1) + 0.04\ (350 - P2 - S2) + 0.045\ P2 + 0.05\ C1 + 0.055\ (250 - C1 - S1 + S2) = Z$$

By multiplying and rearranging, we get a transformed objective function:

$$\$41.75 - 0.005\ B1 + 0.005\ P2 - 0.005\ C1 - 0.02\ S1 + 0.015\ S2 = Z$$

Observe that the objective function is now expressed entirely in terms of nonbasic variables. Where did the $41.75 come from? That is the profit from the policy as it now stands, with the stated levels of the basic variables. We can demonstrate this another way, as in the following table.

Basic Variable	Activity Level	Yield Rate	Yield
B2	400	0.035	$14.00
P1	350	0.04	14.00
C2	250	0.055	13.75
S3	100	0.0	0.00
			$41.75

Thus the transformed objective function, now expressed in terms of nonbasic (zero level) variables, states what would happen to the value of Z if any of the nonbasic variables were given nonzero values. This will be important in the next step of the optimization process.

In the preceding process, the problem constraints were expressed in terms of equations by introducing slack variables into the three inequalities. We picked four of the nine variables and gave them values that satisfied the equations (and therefore satisfied the original inequalities as well). The four nonzero variables are then called basic variables and constitute a basic feasible solution. All other variables are zero. We next solved the four equations for the four basic variables and substituted the results into the objective function.

By using these procedures, we have:

(5) $\$41.75 - 0.005\ B1 + 0.005\ P2 - 0.005\ C1 - 0.02\ S1 + 0.015\ S2 = Z$

(6) $B2 = 400 + S1 - B1$

(7) $P1 = 350 - P2 - S2$

(8) $C2 = 250 - C1 - S1 + S2$

(9) $S3 = 100 + S1 - S2$

It is important to remember that all variables now on the right-hand sides of these equations (6-9) and in the objective function (5) are nonbasic variables and are therefore zero. Thus the profit from this choice of investments is $41.75.

Can this policy be improved? We know intuitively that it can, but what tells us so mathematically? The answer is the existence of positive terms in the objective function. All variables in the objective function are now zero, but what would happen if we were to give some positive value to one of the variables multiplied by a positive coefficient (P2 or S2)? The profit, of course, would be increased.

The next question is, Which of these two present zero (nonbasic) variables should be given a nonzero value. So far, no simple rule has been found for selecting the new variable that will give the greatest improvement in the objective function. It seems entirely reasonable, however, to pick the one that has the largest positive coefficient (multiplier) in the objective function, in this case S2. (And if we should pick the "wrong" variable, in the sense that it is not the one that gives greatest improvement in the objective function, it means at worst some extra computation. We will eventually arrive at the optimum, no matter which variable is chosen.)

If S2 is brought into the basis, which variable will leave? This question is answered by searching for the equation that permits S2 to be given the largest possible value without forcing any other variable into the negative. Equation (6) provides no information: S2 does not appear there. Equation (7) shows that S2 could be as large as 350, which would force P1 to zero and thus out of the basis; we keep this value in mind. Equation (8) does not limit the size of S2. Since it appears there with a plus sign, it could be given any positive value without forcing C2 to become negative. Equation (9) shows that S2 can be no larger than 100 without forcing S3 to become negative. This, then, is the equation that specifies the variable to leave the basis, since it is this equation that limits the value of the entering variable.

We now solve equation (9) for S2:

$$S2 = 100 + S1 - S3$$

The variable S3 thus leaves the basis. Substituting this result into equations (5) through (9) and rearranging, we get:

(10) $\$43.25 - 0.005\ B1 + 0.005\ P2 - 0.005\ C1 - 0.005\ S1 - 0.015\ S3 = Z$

(11) $B2 = 400 + S1 - B1$

(12) $P1 = 250 - P2 - S1 + S3$

(13) $C2 = 350 - C1 - S3$

(14) $S2 = 100 + S1 - S3$

We see that removing S3 from the basis and replacing it with S2 has improved the basis: the profit from the new policy is $1.50 greater than before.

The basis can be improved still further, since there is still one variable with a positive coefficient in the objective function. Let us therefore bring this variable, P2, into the basis, and once again inspect our equations to see which variable should leave. Scanning down equations (11) through (14), we see that P2 appears only in equation (12), so P1 is the variable to leave the basis. Solving this equation for P2 we get:

$$P2 = 250 - P1 - S1 + S3$$

As before, we substitute this result wherever P2 appears in equations (10) through (14), arriving at the following result:

(15) $\$44.50 - 0.005\ B1 - 0.005\ P1 - 0.005\ C1 - 0.01\ S1 - 0.01\ S3 = Z$

(16) $B2 = 400 + S1 - B1$

(17) $P2 = 250 - P1 - S1 + S3$

(18) $C2 = 350 - C1 - S3$

(19) $S2 = 100 + S1 - S3$

Now that all the coefficients in the objective function are negative, no further improvement could be made by making any of these nonbasic variables nonzero. The maximum possible profit is $44.50.

This reasoning is sound. The obvious course was to invest the minimum permissible ($400) in the 3-1/2% bonds, the maximum possible in the highest yield 5-1/2% common stock, and the rest in the higher yield of the two preferred stocks. Multiplying $400 times 3-1/2%, plus $250 times 4-1/2%, plus $350 times 5-1/2% gives the same profit figure of $44.50; this should come as no surprise, but it is nevertheless reassuring to see that the algebra did arrive at the right answer.

More important, we have demonstrated that the algebraic approach does not depend on intuition. We have of course not proved that this method will always work; that belongs to mathematical texts. We have shown, for this example, that the algebraic approach resulted in the same answer as was obvious, and we further demonstrated that there could not possibly be any better policy. Assurance that there is nothing better is of great importance in practical applications, since, if we could see the best policy by looking at the problem, we would have no need for linear programming.

The method used in this example is essentially the simplex method, which is the basis for computer solutions of linear programming problems and which will be discussed in more detail in Chapter 4. For systematic hand calculation, the work is ordinarily done differently; in a computer program a number of things are done differently to conserve time and storage space, but the essential ideas are the same.

1.4: CHARACTERISTICS OF A LINEAR PROGRAMMING APPLICATION

Linear programming is a powerful tool of scientific management, but it is not a cure-all. It does have limitations in terms of the kinds of problems that can be solved and in the way the problems must be formulated. A listing of a problem's requirements will be both a necessary statement of the method's limitations and a good review of the basic ideas.

1. A problem for solution by linear programming methods must have a definite, identified, numerical goal. It is not adequate to say "Waste must be reduced", or "Somehow we have to improve customer service". The goal (objective function) has to be stated as the sum of a series of terms, each consisting of the product of the activity level and a constant multiplier which is the cost or profit per unit of that activity. The goal can only be to maximize the total profit or minimize the total cost, whichever is appropriate. ("Profit" and "cost" are generic terms, however. "Profit" includes total yield from a process, the total income including expenses, and so on; "cost" includes waste, amount of material used, total inventory, and so on.)

2. There must be separate and identifiable activities, and the level of each activity must be measurable in numerical terms. Examples of activities in this chapter have included the amount of a product to be manufactured, the amount of a vitamin supplement to be purchased, and the amount of money to be invested in a type of security. Identification of the activities that affect the objective is often one of the major parts of formulating a linear programming application.

3. The activities must be interrelated. In the investment example of section 1.3, for instance, it would not be possible to set an objective that would maximize profit and at the same time minimize risk -- unless the investor could state in numerical terms how much more risky a high-yield security is than a low-yield one.

4. The restrictions must be identified and stated in numerical terms. It is not enough to say, "The bottleneck in this process is the stamping mill." We must know just how many hours per day or week are available. If this availability can be altered by priority orders, or if it is necessary to include setup time, these factors must be included in the formulation. Failure to state all the constraints will generally lead to misleading results.

5. All activities must appear only in linear combinations, both in the constraints and in the objective function. This means that the only way a constraint may ever be written is as a sum of products, each product consisting of a constant multiplier and a single variable. It is not permissible, for example, for two variables to be multiplied together, for a variable to be squared, or for the square root of a variable to be taken. The same restrictions apply to the objective function.

Frequently, some of these limitations can be overcome. For instance, consider a petroleum pipeline application. The flow rate is not a linear function of pressure, because a small increase in pressure for high pressures would not produce the same increase in flow rate that the same small increase would produce at a lower pressure. In some cases of this kind it is possible to break the graph of flow versus pressure into a series of straight-line segments that approximate the true curve accurately enough. Therefore, even when a function is nonlinear (such as the above), it can often be represented satisfactorily in linear programming formulations. What is necessary is the skill to break the function into line segments or linear approximations. This technique, while not mathematically exact, is often useful and valid.

1.5: TYPICAL LINEAR PROGRAMMING APPLICATIONS

The examples presented in this chapter have necessarily been small and simple. Before going on to an exploration of the additional types of information that can be gained from linear programming methods, let us describe some application areas of more realistic size and see what power linear programming provides in each of them.

Refinery Scheduling

An oil refinery involves, among other things, the blending of some combination of materials to produce one or more different products. The goal can be either maximum profit per barrel, or maximum profit for the total operation (the two do not always lead to the same policy). The constraints describe the maximum supplies of available materials and several performance requirements on the different outputs, such as octane number and vapor pressure. If, as usual, the blending process includes chemical operations to convert some of the available materials into different compounds before blending, the constraints would also include the capacities of these intermediate processes.

A realistic problem would involve 400 to 600 activities and perhaps 200 constraints. With such a complex problem, we cannot accurately find an optimum unless linear programming methods are used. Naturally, oil refineries were in operation before linear programming came into wide use in the 1950s, and refinery managers have been making decisions on these matters for many years. The value of linear programming in such work lies in the large financial effect of small improvements. The manager's decision, reached by previous methods, may have been quite close to the optimum; but a few cents of profit per barrel is a great deal of money when multiplied by production rates in the tens of thousands of barrels per day. In fact, the petroleum industry is presently the largest user of linear programming. We shall explore a refinery scheduling problem in some detail in Chapter 3.

Paper Trimming

A common problem in the paper industry is deciding how to fill a group of orders so as to waste as little paper in trimming as possible. The manufacturer has roll stock in a variety of widths; his orders call for rolls differing in width and length. There may easily be hundreds of ways to cut the orders from the stock; each of these ways becomes one activity in a linear programming solution. The constraints are the widths of the stock rolls. (Length is not a constraint, assuming there is enough stock on hand to fill the orders.) The goal is to fill the orders with as little trim loss as possible.

Formulating this problem for linear programming is both easy and difficult. It is easy in the sense that unquestionably it can be done, and the mathematics involved is simple enough (although we will not go through it). It is difficult in the sense that every possible way of cutting the rolls is an activity, and it can take a long time simply to write down all the constraints. This latter problem can be greatly reduced with the aid of the computer itself, letting the computer enumerate the possibilities. This is just one example of a fairly common technique: let the computer help in preparing the input data, which is often voluminous.

Although the use of linear programming in the paper industry is not so extensive as in oil refining, the economic gains are significant. Trim losses can be held to approximately 1%, whereas without linear programming they are typically 5%. In an industry in which orders are expressed in tons, this can mean a great deal of money.

Production Allocation

In its general outline, the example in section 1.1 (standard and deluxe models) is quite typical--except, of course, for size. Even in that small problem, the optimum policy was not obvious from the problem statement, and naturally many problems are more complex than that one. There would ordinarily be more products, more manufacturing operations, and other types of restrictions. Examples of the last-mentioned would be limits on men and machines independently, when not all machines are used full-time; limits on storage of parts or final products; and different scrap percentages for different activities.

Many other examples could be cited, since linear programming can be utilized in any situation that meets the general requirements stated in the previous section, and in which there is sufficient potential economic payoff to offset the study and solution costs. Such applications have appeared in almost every segment of industry and the military, and the list is continually expanding.

One of the major reasons for the rapid growth of the use of linear programming is the fact that the technique can do a great deal more than simply provide the optimum policy for one objective and one set of constraints. Actually, it is possible to explore the effects of a number of different types of changes in the conditions underlying the problem formulation. Additional studies are often more valuable than the bare statement of an optimum policy. We shall consider these additional kinds of information in the following chapter.

CHAPTER 2: DERIVING ADDITIONAL INFORMATION ABOUT A SOLUTION

In the previous chapter we learned something about the general characteristics of a problem that can be solved with linear programming; we saw some of the mathematical methods by which the optimum solution can be found; and we received an indication of the economic value of the technique. We also saw some ways in which it might be desirable to investigate a problem beyond the finding of one solution to the original problem statement.

One of the most powerful concepts of scientific management is the investigation of several possible alternatives, using a method such as linear programming, before making a final decision. For example, before deciding to add new capital equipment, we would like to study the return on capital in several possible places in the production process where new equipment could be added. Before deciding where to place new sales emphasis, we would like to know which mix of products offers the largest profit potential. Before deciding where to put the most effort into improved contracts for the purchase of raw material, we would like to know which group of materials has the most influence on profit. Before deciding on an efficiency campaign, we would like to know which operations in the manufacturing process are the best candidates for reducing overall costs, considering that some kinds of changes may suggest entirely new approaches to the process.

In other words, the one "answer" giving the single best policy for the problem as stated is often only the beginning of obtaining the information needed about the system under study. It is often equally important to know how that policy would be changed if the problem statement changed; would the profit increase or decrease, and would there be a different set of activities in the optimum policy?

This information can be gained by appropriately changing the mathematical statement of the problem and then by rerunning it. This may take considerable computer time, which we would like to save if possible. Fortunately, methods have been developed to investigate the effects of changes without re-solving the whole problem. In this chapter we shall investigate some of these matters, concentrating on the value of the information derived rather than on the mathematical methods used. The mathematical techniques are not extremely difficult, but it would take considerable effort to learn them. Furthermore, it is practical to gain the benefit of most of these techniques without a detailed understanding of the intricacies of the mathematics.

2.1: THE FUNDAMENTAL "ANSWER"

We begin any linear programming problem with a set of constraints that describe either limitations on resources or minimum requirements that must be satisfied. We also have an objective function that describes either the total profit to be obtained from the set of activities or the total cost of the set of activities. The goal is to maximize or minimize (whichever is appropriate) the value of the objective function, subject to the constraints.

The fundamental "answer" states the level of each activity for a maximum profit (or least cost) and gives the value of that maximum profit. This answer is found by some form of the simplex method.

From the mathematics used, we learn that there will never be more nonzero activities in an optimum policy than there are constraints; there may be fewer. This means, for instance, that in a problem with two constraints and ten possible activities, at least eight of the possible activities will be at the zero level in the optimum policy. This would happen in Example 1 of section 1.1 if there were other products besides standard and deluxe models, but the only constraints were the grinder and polisher capacity. An optimum policy would nevertheless contain only two products. If this proves to be meaningless in a practical problem, it indicates that constraints were omitted in the problem formulation. In Example 1, for instance, we carefully stated the assumption that the manufacturer could sell all he could make of either the standard or deluxe model. In most realistic situations there would be limits on marketability that would require other constraints to be included in the formulation.

A basis consists of as many variables as there are constraints. A basic variable is usually nonzero, but is permitted to be zero; a nonbasic variable must be zero.

Constraints are most commonly expressed as inequalities: "less than or equal to" or "greater than or equal to". An inequality is converted into an equation by the introduction of a slack variable. A slack variable, like all variables in a linear programming problem, must be zero or positive. Any constraint that is not "tight" in an optimum solution (that is, some capacity is not fully utilized) will result in an optimum basis that includes a slack variable. The value of such a slack variable gives the amount by which the corresponding availability is underutilized, or by which a minimum specification is exceeded.

Let us restate the fundamental ideas in terms of an example, which will be used to illustrate all further developments in this chapter.

A manufacturer can make hex nuts, screws and bolts. Each pound of hex nuts requires four man-hours of labor and one hour of lathe time. Each pound of screws requires two man-hours of labor and one hour of grinder time. Each pound of bolts requires two man-hours of labor, one hour of lathe time, and three hours of grinder time. The manufacturer makes $3 profit on each pound of hex nuts, $2 on each pound of screws, and $2.50 on each pound of bolts; he can sell all he can make of each. How many pounds of each product should he make for maximum profit, and what is the profit?

The manufacturer produces other items, but he has decided that for the hex nuts, screws, and bolts he can allow twelve man-hours, two lathe-hours, and four grinder-hours each day.

Let H stand for the number of pounds of hex nuts, S for the pounds of screws, and B for the pounds of bolts. Then we can restate the problem in the usual mathematical form as:

$$\begin{array}{llcl}
\text{Maximize} & 3H + 2S + 2.5B & & \\
\text{Subject to} & 4H + 2S + 2B & \leq & 12 \text{ man-hours} \\
 & 1H + 1B & \leq & 2 \text{ lathe-hours} \\
 & 1S + 3B & \leq & 4 \text{ grinder-hours}
\end{array}$$

We have here three activities (production of hex nuts, screws and bolts) and three constraints (availability of man-hours, lathe-hours and grinder-hours).

The optimum policy can be found by application of the ideas presented in the previous chapter. Since we wish to concentrate now on results rather than on mathematical methods, we shall not go through the algebra, but shall simply state the result. The optimum policy is to produce one pound of hex nuts, four pounds of screws, and no bolts. One hour of lathe time will not be used. The maximum profit is $11.

The basis consists of the following variables: pounds of hex nuts, pounds of screws, and hours of unused lathe time. Thus there is a slack variable in the optimum basis.

A glance at the original problem statement will make this inclusion of a slack in the optimum basis seem plausible. Bolts were forced out because each pound of bolts requires three hours of grinder time, leaving little time for screws. If the manufacturer made a pound of bolts, grinder capacity would limit him to only one pound of screws instead of the optimum four pounds; and four pounds of screws clearly bring in more profit than one pound of bolts.

This justification of the plausibility of the answer is given to assure the reader that the answer is correct, not to suggest that such a justification is ordinarily possible. In most practical applications the problem is much too complex to see at a glance why the answer came out the way it did. If this were possible, there would probably be no need to use linear programming.

2.2: CHANGING LIMITS: RIGHT-HAND SIDE RANGES AND MARGINAL VALUES

Knowing that the optimum policy is to make one pound of hex nuts and four pounds of screws, and that this policy brings $11 profit, we will probably want to know the answers to questions such as the following. How would the profit be affected if another man-hour of labor were made available? How would it be affected if two more man-hours were made available? How many man-hours could be added before the optimum policy would be to make some combination of products other than nuts and screws?

The same types of questions might arise in another way. What if there were uncertainties in the estimates of the hours available? How much would profit be affected if only eleven man-hours were available, or if thirteen man-hours could be used?

Mathematically, all these questions relate to changes in the values on the right-hand sides of constraints. The term for the effect on profit of a change in a right-hand side is marginal value. This is defined as the change in the value of the objective function caused by a unit change (a change of one in the value of the right-hand side of some constraint).

Computing the marginal values of the right-hand sides takes little extra effort, once the optimum has been found (although we shall not take the time to investigate the method). The values for the sample problem are:

Man-hours	\$0.75
Lathe-hours	0.00
Grinder-hours	0.50

Taking man-hours first, one additional man-hour will raise the profit by 75¢; two will raise it by \$1.50; an added half hour will raise it by 37-1/2¢. It also means that removing man-hours will decrease the profit by 75¢ per man-hour removed. If the manufacturer has any other activity on which his profit on a man-hour is greater than 75¢, this marginal value would tell him immediately that he can make more money by shifting effort away from nuts and screws. It is precisely this type of information that is often of the most value in a linear programming application.

The zero marginal value of lathe-hours means that additional lathe-hours do not lead to any increase in profit. Since the manufacturer was not using all his lathe-hours initially, he could not make any additional profit by providing more capacity.

An additional hour of grinder time adds 50¢ to the profit. This may not seem reasonable, for how is it possible to increase grinder-hours without increasing man-hours, since a grinder presumably requires an operator? The answer is that it is not entirely reasonable; what the manufacturer might want to do is to investigate the effect of increasing man-hours and grinder-hours simultaneously. This is precisely the type of thing that is made possible by parametric linear programming, which will be discussed in section 2.6. For now, however, we must emphasize that marginal values here refer to changing one thing at a time, with all others held at their original values. We cannot, for example, legitimately add the marginal values of man-hours and grinder-hours and say that the effect of adding one unit of each would be \$1.25.

The marginal values nevertheless supply much useful information. Suppose that our goal were to provide a rational basis for deciding whether to buy a new lathe or a new grinder: the marginal values would indicate at once that for this part of the factory, at least, there would be no point at all in buying another lathe. Furthermore, the 50¢ marginal value of grinder time supplies information for computing the rate of return on a proposed capital expenditure for a new grinder.

The values given for the marginal values apply only as long as the optimum policy remains the making of hex nuts and screws and the leaving of some lathe-hours unused. Mathematically, we say that the marginal values are valid only as long as the right-hand sides are not changed enough to require a change of basis. (Recall that the optimum basis consists of hex nuts, screws, and unused lathe-hours.) If the number of man-hours is reduced by one, we will make fewer hex nuts (as it happens), but we will still make some of them. We will still make screws, and there will still be some unused lathe-hours. In other words, a reduction of one man-hour does not change the basis, although it does change the value of one of the variables in the basis, and it does change the total profit.

There is clearly going to be some limit on the amount by which the right-hand sides can be changed without requiring a change of basis. If we had only two man-hours, for instance, the optimum policy would be to make one pound of bolts, leaving unused lathe-hours and grinder-hours. The question is: What is the range of values for the right-hand sides, taken one at a time, that does not change the basis, and over which range the stated marginal values are valid?

Information about the range of the right-hand sides can be provided with little extra computation once the optimum has been found. The results for this example are:

Constraint	Lower Limit	Original Value	Upper Limit
Man-hours	8	12	16
Lathe-hours	1	2	∞ (infinity)
Grinder-hours	2	4	6

The table shows, for instance, that the number of man-hours available can range between eight and sixteen hours, with the other right-hand sides at their original values, and the optimum policy will still be to make hex nuts and screws, with some lathe-hours unused. The number of pounds and the profit will change, but the optimum policy will still be to make those two products. Furthermore, within this range, the effect on the profit of a change in the number of man-hours will be to add 75¢ profit for each man-hour added. In other words, the marginal value given earlier is valid throughout this range.

What does it mean that although lathe-hours cannot be reduced below one without changing the basis, lathe-hours can be increased indefinitely without any such change? It simply means that since we are not using all the lathe-hours available, our policy can never be affected by adding more.

What we are calling the marginal value is also called by other names, depending on the nature of the problem and on the writer: incremental cost, raw stock cost, break even price, simplex multiplier, and shadow price. Shadow price is perhaps the most common alternative name; we prefer to avoid it because the same term is also sometimes applied to variation of the coefficients in the objective function, leading to confusion as to the intended meaning.

2.3: CHANGING OBJECTIVE FUNCTION COEFFICIENTS: PROFIT RANGES

Just as we were interested in what would happen if the right-hand sides were changed, it is often valuable to have similar information about the values in the objective function.

For any activity that is in the optimal basis, the concept of marginal value is the same as with the right-hand sides. What is the effect on the optimum profit when there is a unit change in the profit coefficient for an activity? This information is generally not printed by the computer program, since it can be found by inspection of the objective function and from the levels of the optimum basic variables. For instance, if the profit on a pound of screws were $3 instead of $2, the profit would be increased by $4 since the optimum policy is to make four pounds of screws (and still one pound of hex nuts). For a nonbasic variable, changing the unit profit has no effect on the total maximum profit. Since any nonbasic variable is zero, increasing its profit cannot raise the total profit. (That is, it cannot do so unless the unit profit change is great enough to cause a change of basis. We are assuming throughout this section that the basis does not change.)

Since the marginal values of the objective function are so easy to find, they are not printed; in fact, we do not ordinarily use the term "marginal value" in connection with the objective function. When we say "marginal value", it is understood that right-hand sides are intended.

Just as with the right-hand sides, however, there will be a limit on the range over which objective function variations do not cause a change in basis. Since the ranges of validity are not obvious from the objective function and the optimum basis, they can be printed by the computer. They are called profit ranges (or cost ranges, where appropriate), and they give the range of variation for each profit term, taken one at a time, over which the basis does not change. The results for our example are:

Constraint	Lower Limit	Original Value	Upper Limit
Hex Nuts	$0.00	$3.00	$3.50
Screws	1.83	2.00	∞
Bolts	$-\infty$	2.50	3.00

In the case of hex nuts, this means that as long as there is any profit in them, we shall make some of them; if, however, the profit for a pound of hex nuts were greater than $3.50, we would change the basis. For example, if the profit per pound of hex nuts were $4, the optimum policy would be 1.5 pounds of hex nuts, 2.5 pounds of screws, and 0.5 pounds of bolts. There would be no unused capacity. Comparing with the optimum basis for the problem as stated, the activity of making bolts has entered the basis and lathe slack has left it.

The range on the making of screws indicates that if the profit per pound dropped below $1.83 (from the original $2), the optimum policy would involve a different basis, but no amount of increased profit on screws would make us change the basis. For bolts, we see that no matter how low the profit dropped, we would not make any. This is no surprise, since it was not profitable to make them anyway. If the profit for a pound of bolts rose above $3, however, the optimum basis would include some bolts.

This ranging information is useful management data in at least two ways: (1) to determine whether to change policy if the profit changes, by a market price variation or by a change in the cost of materials or production, and (2) to determine how sensitive the choice of policy is to uncertainty in the objective function coefficients. These profit figures for each activity naturally involve some prior calculations: what are the costs of materials, labor and overhead for each product? There is inevitably some estimating involved in determining these coefficients; would the optimum policy be greatly different if it were discovered that some of the figures were a bit inaccurate? A glance at the profit ranges shows that the profit on hex nuts and bolts would have to be more than 50¢ higher before the optimum policy would change, but if the profit on a pound of screws dropped by more than about 17¢ (a percentage change of only about 9%), the best policy would be different. If the profit on screws dropped by more than that much, or if there is more than that much uncertainty in the profit estimate, then the "optimum" policy is in some doubt. This kind of information can clearly be of great value in evaluating a policy suggested by a linear programming solution.

It is sometimes of interest to know what it would cost if we deliberately introduced some amount of an activity that is not in the optimum basis, in an amount that does not force any optimum basis variable out of the basis. The value of this number is that it tells us how much the profit on such a nonbasic variable would have to be increased before it would become profitable to introduce the variable into the basis. In our problem, this would reasonably be called the reduced profit, but since the concept first arose in connection with cost minimization problems, it is usually called the <u>reduced cost</u>.

The only nonslack activity not in the basis for the problem as stated is the manufacture of bolts, so this is the only variable that has reduced cost; the figure is 50¢ per pound. We cannot force more than half a pound of bolts without using up the lathe slack--which would be a change of basis. When a half pound of bolts is made, other variables have changed values. What is the total effect on the profit? The reduced profit is 50¢ per pound, so the reduced profit for half a pound is half of 50¢, or 25¢, making the new total profit $10.75.

The reduced cost is one example of added information that can be derived from a linear programming solution with little extra effort. It is at least useful secondary information; in some situations it becomes almost as important as the statement of the optimum policy.

2.4: TRADEOFFS: RATES OF SUBSTITUTION

We have so far seen two ways of asking questions about how the optimum policy would be changed if the data in the problem statement were modified. The marginal value of a right-hand side gives the effect on the profit of a unit change in the value of a restriction. The reduced cost gives the effect on the profit of the introduction of one unit of some activity not in the optimal basis. For both right-hand-side variations and for profit variations, ranging information tells the extent of each change that can be made without requiring a change of basis--that is, a shift to a different policy. These variations are all considered independently; only one thing is allowed to vary at a time. The only information given in either case is the effect on the profit. Although the basis will not change (we will still make the same products), the values of the variables in the basis and of the total profit will change.

This section somewhat generalizes these concepts. We will still consider only one change at a time, but now we can get a statement of the effect of the change on all the basic variables, as well as the effect on the profit. For example, if the number of man-hours available is reduced by one, what effect will this have on profit, on the pounds of hex nuts made, the pounds of screws made, and the number of lathe-hours used? The answer is given in terms of <u>substitution rates</u>, which are not limited to just one unit of change, but apply for any value within the range of validity of the substitution. The following table gives the substitution rates for the problem statement.

	Man-hours	Grinder-hours	Bolts
Profit	-0.75	-0.50	-0.50
Pounds of hex nuts	-1/4	1/2	1
Lathe-hours used	1/4	-1/2	2
Pounds of screws	0	-1	-3
Extent of this substitution	4	2	1/2

From these entries we see that a reduction of one man-hour (which is equivalent to changing the right-hand side of the first constraint to eleven) reduces profit by $0.75, reduces the number of pounds of hex nuts in the optimum policy by 1/4, increases the number of lathe-hours by 1/4, and has no effect on the pounds of screws produced. The increase in the number of lathe-hours is due to the reallocation of resources made necessary by the change in man-hours. If the number of man-hours were increased by one, profit would rise by $0.75; pounds of hex nuts would rise by 1/4; lathe-hours would drop by 1/4; and there would still be no effect on the pounds of screws. This substitution may be carried out to an increase of four hours.

For grinder-hours, a reduction of one would decrease profit by $0.50, increase hex nuts by 1/2 pound (again due to reallocation), reduce lathe hours by 1/2, and reduce the screws by one pound.

The pounds of bolts cannot be reduced (since we are making none), but they can be increased up to 1/2. If this is done, profit will be reduced by $0.50 per pound of bolts; half a pound will reduce profit by $0.25. The increase of 1/2 pounds of bolts will increase the pounds of hex nuts at a rate of one for one, giving an actual increase of 1/2. Lathe hours increase at a rate of two for one, leading to an extra hour. Screws go down three times as fast as bolts go up, leading to a decrease of 1-1/2.

2.5: SENSITIVITY TO TECHNOLOGY

Now we turn to a rather different type of investigation. So far we have assumed that the coefficients in the constraints (the left-hand sides) have not varied. For instance, none of the changes considered so far have altered the fact that it takes four man-hours to produce a pound of hex nuts.

This question can obviously be of considerable interest for the same two major reasons that the other items are of interest: because of possible changes in these figures, and because of possible uncertainty in the data.

This time we are asking for the sensitivity of the profit to a small change in a coefficient of a constraint. The following table gives this effect for all the coefficients.

	Hex Nuts	Screws	Bolts
Man-hours	-3/4	-3	0
Lathe-hours	0	0	0
Grinder-hours	-1/2	-2	0

The table shows, for instance, that total profit decreases 3/4 as fast as the number of man-hours per pound of hex nut increases; it decreases three times as fast as the number of man-hours per pound of screws increases. Since we are making no bolts in the optimum policy, any changes in the hours for making bolts will have no effect on profit.

Since we are not using all the lathe-hours available, a small increase there will have no effect on the profit either. This probably seems unreasonable, for if the lathe-hours required to make a pound of hex nuts were increased, surely the lathe operator would have to work longer. The answer to this apparent contradiction is that nothing in the system of constraints states that there is any relationship between lathe-hours and man-hours.

It will be instructive to make a change in one of the system coefficients and see if the entries in the preceding table are applicable. Let us see what happens to the profit if the number of grinder-hours per pound of screws is increased from one to two. Solving the problem, we find that the optimum policy now is to make two pounds of hex nuts and two pounds of screws, leading to a profit of $10. This is a reduction of only $1, whereas the table said to expect a reduction of $2. What happened?

Let us now see what happens if the grinder hours per pound were increased by 1/2 instead of one. Now the optimum is to make 1-7/9 pounds of hex nuts and 2-2/9 pounds of screws; the profit is 10-1/3 dollars. The profit was reduced 2/3 dollars by the increase of 1/2 hour of grinder time; the profit loss is now greater than the time increase, but still not twice as great.

Still, the ratio is greater than what we had for an increase of one. Let us now try a quite small change of 0.1 hour. This time, the optimum policy is to make 1.1818 pounds of hex nuts and 3.6364 pounds of screws; the profit is $10.82. Thus for 0.1 hour increase in time we got $0.18 drop in profit, which is getting closer to the 2:1 ratio the table predicts.

The answer to all this is that the table gives the rate of change of profit for coefficient values very close to the original values. For large increases, the rate is not the same. The relationship is said to be nonlinear. If it is important to know the effect of a large change, such as the 100% time increase we tried first, some other technique must be used.

Nevertheless, it should be clear that sometimes such small changes are all that we would want to investigate. As a matter of fact, the effects of such changes are occasionally the main purpose of a linear programming study.

2.6: CHANGING SEVERAL THINGS AT ONCE: PARAMETRIC PROGRAMMING

Everything we have discussed so far has involved two fundamental assumptions:

1. One thing is changed at a time. The level of a number of other variables may change as a result, but the cause of all the changes is the variation of the one.
2. The changes are not so great as to cause a change in the basis.

We have already seen situations in which we would like to go beyond such techniques. In connection with varying the availability of lathe-hours or grinder-hours, for instance, it is natural to want to vary man-hours simultaneously, since the machine operator must work as long as the machine is in operation. More generally, we might want to study the effect of an overall increase in capacity--for example, by working longer hours or by adding a few of all types of machines. Or we might want to study the effect of increasing one capacity and, at the same time, decreasing another.

When such sweeping changes are under study, it is unduly restrictive to consider only the one optimum policy--that is, to be limited to no change of basis. When a major management decision is under study, it is reasonable to expect that large changes in the structure of the system may make a different policy much more attractive than the one that is optimum under the initial conditions.

Parametric linear programming allows the freedom we need for this kind of study, since several factors can be varied simultaneously with no limit on the amount of basis-changing that may be required. The technique is most commonly applied to the right-hand sides of the constraints and to the objective function, but in principle it can be used with any row or column of the entire system.

Let us again consider the illustrative problem and investigate the effect of increasing man-hours and grinder-hours at the same rate, and decreasing lathe-hours at 1/3 that rate. How do the allocations change? What is the effect on the profit? When do changes of basis occur? How far can the variations be continued?

Keep in mind that we now want to increase the right-hand sides continuously--that is, to consider all new values, rather than selected and isolated ones. To describe this kind of variation we will assign for each right-hand side a rate of change, which may be positive, negative or zero. Then we will establish a parameter that increases smoothly and indefinitely from zero. For any one constraint, this can be written as:

New right-hand side = Old right-hand side
+ (parameter X rate of change)

We have such a relation for every constraint; the rate of change can vary with each new constraint, but the parameter is the same in all. As the parameter increases, it describes the simultaneous variation of all the right-hand sides.

In the example, the rates for man-hours and grinder-hours are both one, and the rate for lathe-hours is -1/3. As the parameter increases smoothly from zero to one, for instance, we are in effect asking what happens to the solution if instead of twelve man-hours we had 13; instead of two lathe-hours we had 1-2/3; and instead of four grinder-hours we had five. When the parameter increases to two, we have fourteen, 1-1/3, and six hours respectively. The parameter continues to increase; at some point a variable in the original basis would become negative if the parameter were increased further. If this happened, the solution would no longer be feasible, and thus would require a change of basis. At this point a parametric linear programming program in the computer would print the names of the changed basis variables and perhaps their levels at the point of change. It might also print the marginal values of all constraint right-hand sides, and then look for the next change of basis.

When that change is found, the new basis is printed, and so on.

The computer program does not actually do all this by experimentation. There are mathematical techniques that state exactly when a change of basis is required, in terms of the parameter. All the computer program actually does is compute this value of the parameter, print the appropriate data, and then go through some manipulations that determine what the new basis is.

Eventually, one of two things will happen. Either a point will be reached beyond which the parameter cannot be increased any further and still maintain feasibility, or a point will be reached beyond which the parameter can be increased indefinitely without ever requiring a change of basis. In the latter case the right-hand sides are said to be open.

In the illustrative problem, the first change of basis occurs when the parameter reaches four, which corresponds to sixteen man-hours, 2/3 lathe-hours, and eight grinder-hours. At that point, we are making eight pounds of screws and no nuts or bolts; the profit is $16. As the parameter increases further, it becomes profitable to begin making bolts instead of hex nuts; there is still slack in the lathes. After the parameter reaches 5-1/7, the lathes are "tight" (no slack), but there is slack in the grinders. (This reallocation of slacks is a change in basis. The basis consists of whatever variables have non-zero values.) As the parameter now increases further, we start cutting down on the pounds of bolts. When the parameter reaches six, we are again making no bolts--but we are not making hex nuts either. From here on, indefinitely, we make nothing but screws; there is slack in both lathe-hours and grinder-hours.

Through the use of parametric linear programming, a computer would print out the values of the right-hand sides at which basis changes occur, the new basis at each change, the levels of all basis variables at each change, and the rates of change of all basis variables from there to the next basis change. With this information in hand, the effect of the variation on profit and the basis variables can be plotted, as in Figure 13.

The graph, of course, does not show all that can be done with parametric linear programming. The rates of change for the constraints, for instance, may be zero. Suppose that we wanted to investigate the effect of increasing lathe-hours, with each additional lathe-hour requiring an additional man-hour. We would set the rate of change for man-hours at one, the rate for lathe-hours also at one, and the rate for grinder-hours at zero. Or suppose that we wanted to study the effect of increasing lathe-hours and simultaneously decreasing grinder-hours, with grinder-hours decreasing twice as fast as lathe-hours increase. We would make 0 the rate of change for man-hours; 1 the rate for lathe-hours; and -2 the rate for ginder-hours.

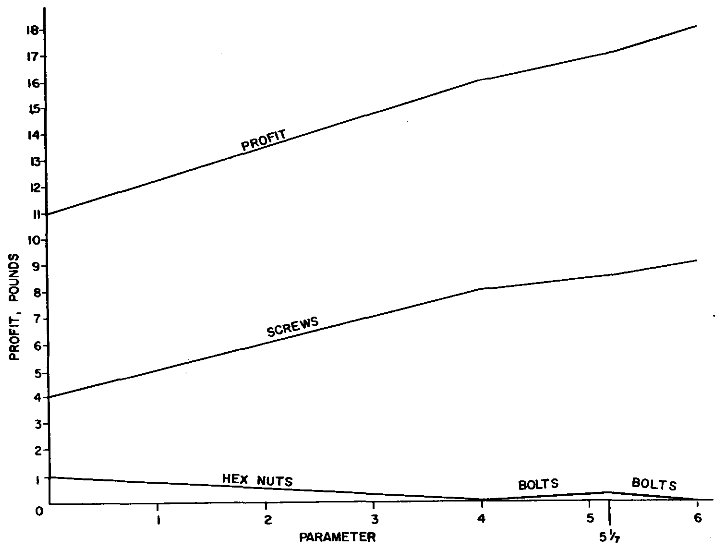


Figure 13. Graph of the effect on profit, pounds of screws, pounds of hex nuts, and pounds of bolts, of a simultaneous increase in the right-hand sides. One unit on the parameter scale corresponds to one unit in man-hours, lathe-hours and grinder-hours.

The possibilities are almost unlimited. Once the optimum solution to a problem has been found, getting the parametric programming results takes comparatively little extra time. It is therefore economically practical to run many parametric programming variations, leading in many cases to an extremely valuable analysis of the various alternatives open to management.

We have discussed only variation of the right-hand sides. Precisely the same things can be done with simultaneous variation of the coefficients in the objective function. This can be valuable, for instance, in determining the effects of possible simultaneous price changes, or in establishing what the price would have to be to make a potential course of action profitable.

Parametric programming is perhaps the best current example of the expanding usefulness of linear programming to management. As research goes on and as new features are incorporated into the computer systems for linear programming, management will find it even more helpful.

CHAPTER 3: A LARGER EXAMPLE -- OIL REFINERY SCHEDULING

We have now seen a number of examples of small problems that can be approached with the methods of linear programming. In this chapter a somewhat larger problem is discussed. We shall explore the physical situation in which the problem arises, the formulation of the linear programming model, the preparation of computer input, the presentations of results by the computer program, and some variations.*

3.1: THE PHYSICAL SITUATION

The basic material for an oil refinery is called crude oil, or simply crude. It is a liquid mixture of a large number of chemical compounds ranging from low-boiling-point ("light") components such as gasoline, to high-boiling-point ("heavy") components such as asphalt and residual fuel. A refinery normally has to process many different types of crude, with the types differing in the proportions of their constituent compounds and in price. The price is expressed in dollars per barrel ($/bbl). The refinery in this example is able to process two different types of crudes: crude 1 and crude 2.

A refinery consists of a number of process units for separating, changing or combining crude-oil components. (The crude-oil components are usually called fractions, since each represents a "fraction" of the crude.) A refinery also has a large amount of storage space for raw materials, intermediate products and final products. In our simplified example, the refinery has only two process units -- a pipe still and a catalytic cracking unit -- in addition to its storage tanks.

A pipe still is a device in which crude oil is distilled to separate it into fractions according to boiling point. The highest boiling fraction is called light naphtha; the middle boiling fraction is called medium distillate; and the lowest boiling fraction is called heavy residuum. In short, a pipe still takes crude oil as its input, heats it, and produces the three fractions as its outputs; each fraction appears at a separate pipe. The relative proportions of the three outputs depend both on the physical characteristics of the crude input and on the operation of the pipe still. In this problem the pipe still can be operated to produce either a high proportion of fuel (the heavy residuum) or a high proportion of medium distillate.

A catalytic cracking unit ("cat cracker") operates on the medium distillate fraction from the pipe still. Through a series of chemical reactions, it changes the medium distillate into three outputs called catalytic naphtha ("cat naphtha"), light cycle stock, and heavy cycle stock. Part of the cycle-stock output can be returned to the input of the unit, in an operation called recycling.

The essence of the cat cracker's operation is the "cracking" of the heavy molecules of the medium distillate into lighter and smaller molecules. The weight of the outputs cannot exceed the weight of the input, but since the small molecules take up more space than the large ones, the volume of the outputs exceeds that of the input.

From these two processes we obtain seven components that may be combined in various ways to produce four products. The seven components are:

1. The light naphtha from the pipe still produced while processing crude 1. Since the light naphtha is not run through the cat cracker or otherwise processed after leaving the pipe still, it is called virgin naphtha.
2. The light naphtha from the pipe still produced while processing crude 2. These two components are chemically the same, but they are stored separately and must therefore be accounted for separately.
3. The medium distillate from the pipe still, produced from either crude 1 or crude 2. (The distillate is not stored separately according to which crude is being processed.) Most of the medium distillate goes to the cat cracker, but some of it is used directly in blending final products. Once again, since this component is not further processed after leaving the pipe still, it is called virgin distillate.
4. The residuum from the pipe still. Some of this goes to the cat cracker, but the bulk of it is used in blending one of the final products.
5. Cat naphtha from the cat cracker.
6. Light cycle stock from the cat cracker.
7. Heavy cycle stock from the cat cracker.

These seven components are blended to make four products. The constituents of each product are the following:

1. Regular gasoline: virgin naphtha from crude 1, virgin naphtha from crude 2, and cat naphtha.

* The material in Chapter 3 is based upon the paper "Linear Programming Applied to Refinery Scheduling" (IPT-PR-0199), by David Smith, copyright 1963 C-E-I-R, Inc., and is used with their permission.

Name	Description
	VARIABLES
CR1F	Crude 1 run to fuel, MB/D
CR1D	Crude 1 run to distillate, MB/D
CR2F	Crude 2 run to fuel, MB/D
CR2D	Crude 2 run to distillate, MB/D
CATVD	Cat feed virgin distillate, MB/D
CATLC	Cat feed light cycle, MB/D
CATHC	Cat feed heavy cycle, MB/D
RSFUL	Residuum to fuel, MB/D
HCFUL	Heavy cycle to fuel, MB/D
LCFUL	Light cycle to fuel, MB/D
VDFUL	Virgin distillate to fuel, MB/D
VNAPD	Virgin naphtha to distillate (heating oil), MB/D
VDISD	Virgin distillate to distillate (heating oil), MB/D
LCCYD	Light cycle to distillate (heating oil), MB/D
VN1R	Virgin naphtha (crude 1) to regular gasoline, MB/D
VN2R	Virgin naphtha (crude 2) to regular gasoline, MB/D
CNPR	Catalytic naphtha (crude 1) to regular gasoline, MB/D
VN1P	Virgin naphtha (crude 1) to premium gasoline, MB/D
VN2P	Virgin naphtha (crude 2) to premimum gasoline, MB/D
CNPP	Catalytic naphtha to premium gasoline, MB/D
	RELATIONS
CPRRR	Change premium gasoline price row, $M/D
PROFT	Profit, $M/D
VNAP1	Virgin naphtha 1, material balance, MB/D
VNAP2	Virgin naphtha 2, material balance, MB/D
VDIS	Virgin distillate, material balance, MB/D
RESID	Residuum, material balance, MB/D
CATN	Catalytic naphtha, material balance, MB/D
LCCY	Light cat cycle oil, material balance, MB/D
HCCY	Heavy cat cycle oil, material balance, MB/D
CRBAL	Crude balance (crude 1 = 1/3 total), MB/D of crude 1
PSCAP	Pipe-still capacity, MB/D
CR2AV	Crude 2 availability, MB/D
TFCAT	Total feed cat capacity, MB/D
FREQ	Fuel oil volume requirement, MB/D
DREQ	Distillate (heating oil) volume requirement, MB/D
DSPEC	Distillate (heating oil) specification, M CON. B/D better than spec.
RREQ	Regular gasoline volume requirement, MB/D
RSPEC	Regular gasoline specification, M OCT. B/D better than spec.
PREQ	Premium gasoline volume requirement, MB/D
PSPEC	Premium gasoline specification, M OCT. B/D better than spec.
where	MB/D = thousands of barrels per day
	$M/D = thousands of dollars per day
	M CON. B = thousand contamination number of bbls.
	M OCT. B = thousand octane number bbls.

TABLE 1. Symbols used in Figure 14 and in text

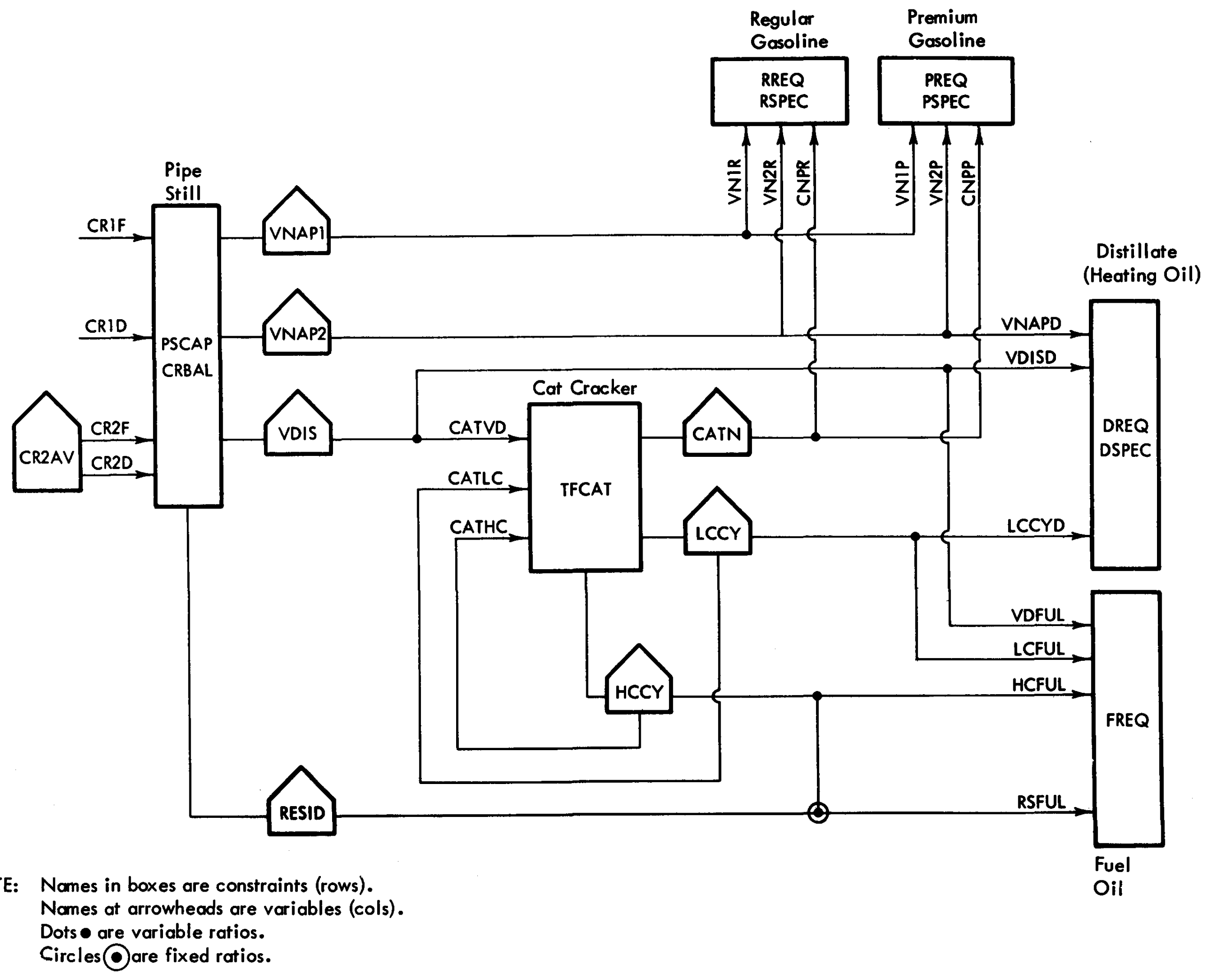


Figure 14. Refinery flowchart

2. Premium gasoline: same components as regular gasoline, but in different proportions to meet different product specifications.

3. Heating oil: virgin naphtha from crude 1, virgin distillate, and light cycle stock.

4. Fuel oil: virgin distillate, residuum, light cycle stock, and heavy cycle stock.

The flow of materials and intermediate products to form final products is shown in Figure 14, which should be read with Table 1. This diagram is not difficult to understand if it is studied with the preceding explanations.

Two conventions are employed in Figure 14. Names at the arrowheads are variables, which will correspond to columns in the linear programming formulation; names in boxes relate to constraints, which will correspond to rows. Rectangular boxes indicate processing or blending; boxes with peaked roofs indicate a point at which a material flow must balance -- that is, the sum of the input flow must equal the sum of the output flow. For instance, the box labeled VNAP1 refers to virgin naphtha from crude 1; the material flowing into this point must equal the material flowing out to the two types of gasoline. One of the constraints to be discussed later will express this material balance mathematically.

3.2: THE OPERATING DATA

Now that we have some idea of how an oil refinery works, let us investigate the factors that determine whether or not a scheme of operation is profitable. This will involve the costs of raw materials, the costs of operation, the characteristics of the various processes, and the prices of the final products. Many of the ways of running the refinery can be varied: choice of crudes, operating characteristics of the pipe still and the cat cracker, the proportions of components used in blending the final products, and "mix" of final products. The goal, of course, is to operate the refinery for maximum profit.

The remainder of this section presents the basic data and other information necessary to develop the mathematical statement of the problem.

Crude Supply

Crude 1 costs $2.75 per barrel delivered at the refinery gate. Because of quality problems, crude 1 can make up no more than one-third of the total crude input. Crude 2 costs $2.85 per barrel; its maximum availability is 75,000 barrels per day (75 MB/D).

Pipe-Still Operations

Each crude can be run either for maximum fuel or for maximum distillate. The pipe-still capacity is the same (100 MB/D) either way. The pipe still costs $.25 per barrel to operate, above the cost of the small amount of input crude that is burned to heat the still. This latter amount has been deducted from the yields in Table 2. The table shows the fractional yield from each crude, on each type of operation (maximum fuel or maximum distillate). The table indicates, for instance, that if one barrel of crude 1 is run for maximum fuel, the yield will be .10 barrel of virgin naphtha, .25 barrel of virgin distillate, and .60 barrel of residuum.

The sum of these fractions is less than one barrel because of the amount of crude used to heat the still. When crude 1 is run for maximum distillate, there is a higher proportion of virgin naphtha and virgin distillate, and a lower proportion of residuum. (It is primarily the residuum that goes into fuel.) Crude 2 costs more and produces higher percentages of the virgin naphtha, which is one of the components of the higher-priced gasoline.

	crude 1		crude 2	
	fuel	distillate	fuel	distillate
virgin naphtha	.10	.15	.20	.25
virgin distillate	.25	.40	.20	.35
residuum	.60	.40	.50	.30

TABLE 2. Fractional yields for both crudes, operating the pipe still for maximum fuel or maximum distillate

Catalytic-Cracker Operation

Table 3 gives the operating characteristics of the cat cracker. The three columns correspond to the three inputs. The primary input is the virgin distillate from the pipe still, but we must also consider that part of both the light-cycle stock and heavy-cycle stock outputs can be returned for recycling as input.

The first line of Table 3 gives the variable operating cost per barrel. The second line shows that the virgin distillate is contaminated with a little residuum; the light-cycle and heavy-cycle stocks are not. The third line refers to an internal recycle: part of each input is recycled within the cat cracker, in addition to the external recycle shown in the diagram of Figure 14. The last three lines give the fractions of output.

The table can best be understood by studying the entries in one column. One barrel of virgin distillate costs $.10 to process; it is contaminated with .1 barrel of residuum; and .6 barrel of this input is recycled within the cat cracker. If this were the only input, the output would be .7 barrel of cat naphtha, .3 barrel of light-cycle stock, and .5 barrel of heavy-cycle stock. The outputs add up to more than the input because the large molecules of the input are chemically broken down into smaller molecules of the outputs, with an increase in volume.

	virgin distillate	light cycle	heavy cycle
variable operating cost, $/bbl.	.1	.15	.16
direct feed	1.1	1.0	1.0
internal recycle, bbl.	.6	.2	.1
cat naphtha, bbl.	.7	.6	.3
light cycle, bbl.	.3	-	.7
heavy cycle, bbl.	.5	.5	-

TABLE 3. Catalytic-cracker operating characteristics: cost and yield

Gasoline Blending

Two grades of gasoline are blended: premium, which can be sold for $5 per barrel, and regular, which can be sold for $4.50 per barrel. An octane specification states that the three components of the premium gasoline must be blended so that they give an octane number of at least 89; the octane number of the regular gasoline must be at least 85. A volume specification states that there must be at least 25 MB/D of premium gasoline and no more than 10 MB/D of regular gasoline.

Table 4 gives the octane number of each of the gasoline components and summarizes the specifications that must be met.

	octane number	volume
virgin naphtha crude 1	85	
virgin naphtha crude 2	84	
catalytic naphtha	92	
premium specification	89	at least 25 MB/D
regular specification	85	at most 10 MB/D

TABLE 4. Octane numbers of gasoline components and gasoline-blending requirements

Distillate (Heating Oil) Blending

Home heating oil can be sold for $4 per barrel. It is subject to a specification on the maximum allowable contamination number, a quality of heating oils that is related to the amount of certain undesirable chemicals. There is the additional requirement to produce at least 30 MB/D. In this particular example there is the final specification that the virgin distillate from this day's operation will be supplemented by 11,000 barrels (11 MB) from storage, and that 11 MB of the day's production of light-cycle stock will be placed in storage. All of this is summarized in Table 5.

	contamination number	volume
virgin distillate	54	as produced plus 11 MB/D from storage
virgin naphtha 2	50	as produced
light-cycle stock	65	as produced less 11 MB/D to storage
specification	55	at least 30 MB/D

TABLE 5. Distillate (heating oil) blending characteristics

Fuel Blending

The only requirement on fuel blending is that the total production must not exceed 50 MB/D. It can be sold for $2.50 per barrel.

3.3: THE OBJECTIVE FUNCTION AND CONSTRAINTS

We now have all the necessary data, but it is not in a systematic form suitable for input to the computer. The next step is to write the objective function and the constraints (see Figure 15).

Objective Function

The objective function in this example includes not only profits, as expected, but also the variable and fixed costs. The objective function is the first row in the system, as displayed in Figure 15. The name PROFT has been given to the row. The objective function has been set up so that costs are positive and profits are negative.

The $3 under CR1F represents the cost of a barrel of crude 1 to fuel: $2.75 for the crude itself, and $.25 for its processing in the pipe still. The cost of a barrel of crude 1 to distillate is also $3. Crude 2, for either purpose, costs a total of $3.10: $2.85 for the crude, and $.25 for its processing in the pipe still. The costs under CATVD, CATLC, and CATHC are the variable processing costs for the three cat-cracker outputs, as given by Table 3. The remaining objective-function coefficients are sales prices per barrel for the four products: $2.50 per barrel for fuel, $4 per barrel for distillate, $4.50 per barrel for regular gasoline, and $5 per barrel for premium gasoline.

Material-Balance Constraints

The row VNAP1 is the material balance equation for virgin naphtha from crude 1. Plus signs are used with outputs, and minus signs are used with inputs. (This use is arbitrary; the reverse would be just as good.) The equation thus states that the virgin naphtha from crude 1 while running for maximum fuel (.1 barrel), plus the virgin naphtha from crude 1 while running for maximum distillate (.15 barrel), must equal the amount of virgin naphtha going to regular gasoline plus the amount of virgin naphtha going to premium gasoline. The row VNAP2 gives a similar equation for virgin naphtha from crude 2.

VDIS is a material-balance equation for virgin distillate. Its sources are given negative coefficients corresponding to the entries in Table 2. The three destinations are shown as 1s under the columns for input to the cat cracker, heating oil and fuel. (The validity of these relations should be checked against Figure 14, which shows the flow paths.) The 11 on the right-hand side reflects the 11,000 barrels per day of virgin distillate from storage. The rows RESID, CATN, LCCY, HCCY and CR2AV are similar material-balance equations for other points in the process.

Pipe-Still Constraints

CRBAL states the requirement that crude 1 be no more than one-third of the total crude. The logic here is that two times the amount of crude 1 must be less than or equal to the amount of crude 2. This can also be stated as a requirement that the amount of crude 1 cannot be more than half the amount of crude 2, which is what has been done here. PSCAP is the limitation on pipe-still capacity: the sum of the crudes, for both purposes, must not exceed 100,000 barrels per day.

Cat-Cracker Constraint

TFCAT is the constraint on total cat-cracker capacity. Because of the internal recycling, this is slightly different from the other capacity limitations. For each barrel of virgin distillate input, .6 barrel of the output is recycled internally. Therefore the input quantity (CATVD) must be multiplied by 1.6 to give the total flow through the cracker. Similar requirements apply to the amounts of light-cycle and heavy-cycle stock.

Fuel Oil-Blending Constraint

FREQ gives the total production requirement for fuel oil, which is the only constraint on this product: no more than 50 MB/D.

Heating Oil-Blending Constraints

Heating oil is subject to two restrictions. DREQ states that there must be at least 30 MB/D of heating oil produced. DSPEC states that the contamination number of the heating oil must be no greater

		Column Name																				
		Pipe-still activities				Cat-cracker activities			Fuel-blending activities				Heating oil-blending activities			Regular gasoline-blending activities			Premium gasoline-blending activities			
	Row Name	CRIF	CRID	CR2F	CR2D	CATVD	CATLC	CATHC	RSFUL	HCFUL	LCFUL	VDFUL	VNAPD	VDISD	LCCYD	VNIR	VN2R	CNPR	VN1P	VN2P	CNPP	RHS
Objective function	PROFT	3.0	3.0	3.1	3.1	.1	.15	.16	-2.5	-2.5	-2.5	-2.5	-4	-4	-4	-4.5	-4.5	-4.5	-5	-5	-5	= -70
Material-balance constraints	VNAP1	-.1	-.15													1			1			= 0
	VNAP2			-.2	-.25								1				1			1		= 0
	VDIS	-.25	-.4	-.2	-.35	1						1		1								= 11
	RESID	-.6	-.4	-.5	-.3	.1			.9													= 0
	CATN					-.7	-.6	-.3										1			1	= 0
	LCCY					-.3	1	-.7			1				1							= -11
	HCCY					-.5	-.5	1	.1	1												= 0
	CR2AV			1	1																	≤ 75
Pipe-still constraints	CRBAL	1	1	-.5	-.5																	≤ 0
	PSCAP	1	1	1	1																	≤ 100
Cat-cracker constraint	TFCAT					1.6	1.2	1.1														≤ 46
Fuel-blending constraint	FREQ								1	1	1	1										≤ 50
Heating oil-blending constraints	DREQ												1	1	1							≥ 30
	DSPEC												-5	-1	10							≤ 0
Regular gasoline-blending constraints	RREQ															1	1	1				≤ 10
	RSPEC															0	1	-7				≤ 0
Premium gasoline-blending constraints	PREQ																		1	1	1	≥ 25
	PSPEC																		4	5	-3	≤ 0

Figure 15. The oil refinery objective function and constraints

than 55 (as shown previously in Table 5) or that the maximum contamination must be less than 55 (VNAPD + VDISD + LCCYD). The variables in the parentheses constitute the allowed components. The contamination number of each component is:

50 for VNAPD
54 for VDISD
65 for LCCYD

where one unit of VNAPD would contribute 50 contamination units, and so forth.

This constraint can then be written: 50 VNAPD + 54VDISD + 65 LCCYD $\leq$ 55 (VNAPD + VDISD + LCCYD). Rewriting this constraint to move all variables to the left of the $\leq$ sign, we have: 50 VNAPD - 55 VNAPD + 54 VDISD - 55 VDISD + 65 LCCYD - 55 LCCYD $\leq$ 0 which reduces to -5 VNAPD - 1 VDISD + 10 LCCYD $\leq$ 0. This is what is given in Figure 15.

Gasoline-Blending Constraints

The last four lines state the capacity and octane requirements for regular and premium gasoline, with the octane requirements being derived by a method similar to that for the contamination number. The only difference is that with octane number the requirement must be met or exceeded; thus the signs are different.

3.4: COMPUTER INPUT

A linear programming system for a computer includes facilities for the entry of the problem data -- that is, the objective-function coefficients, the coefficients of the constraints, and the right-hand sides. Figure 16 shows typical input for one linear programming system, with some of the data for this example.

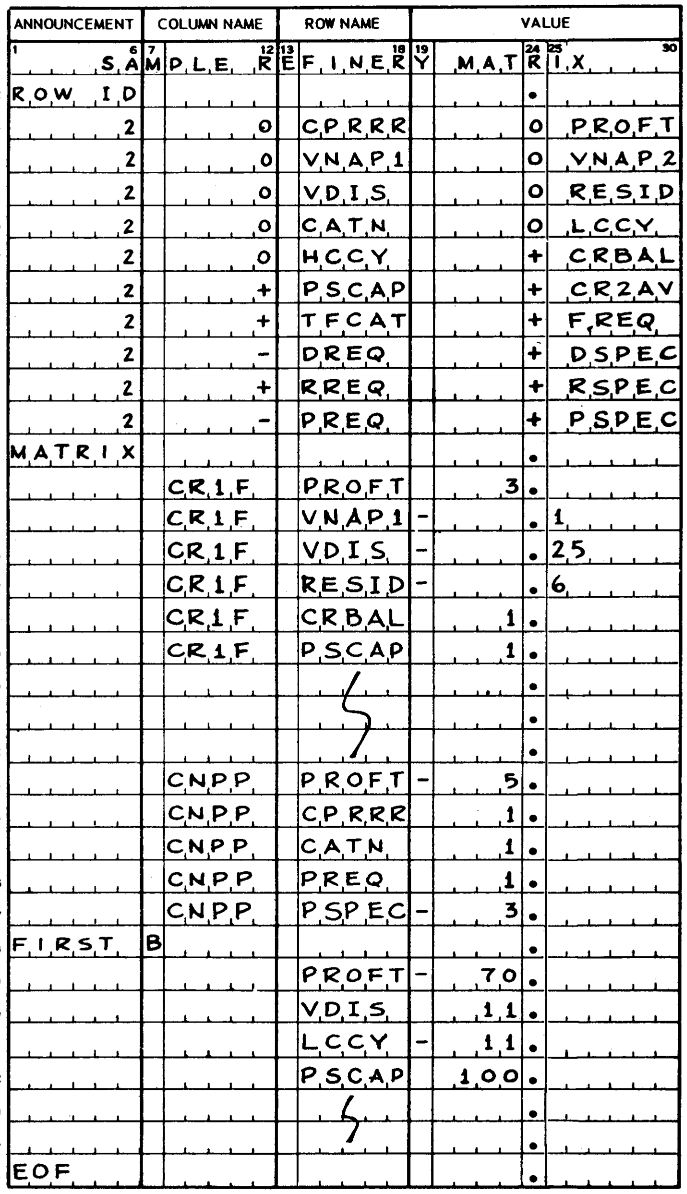
The first line states the name of the problem. The second line is a signal to the computer program that the names of the rows follow. The names themselves are written two to a line. Each name is preceded by either a zero, a plus sign, or a minus sign. The signs indicate, respectively, whether the restriction is an equality, a less-than condition, or a greater-than condition.

The word MATRIX signals that the constraint coefficients follow. Each is identified as to its row and column, and the system is designed to assume that if no coefficient is entered for a position, that value is zero.

FIRST B signals that the right-hand sides follow. The EOF signals the end of the input deck.

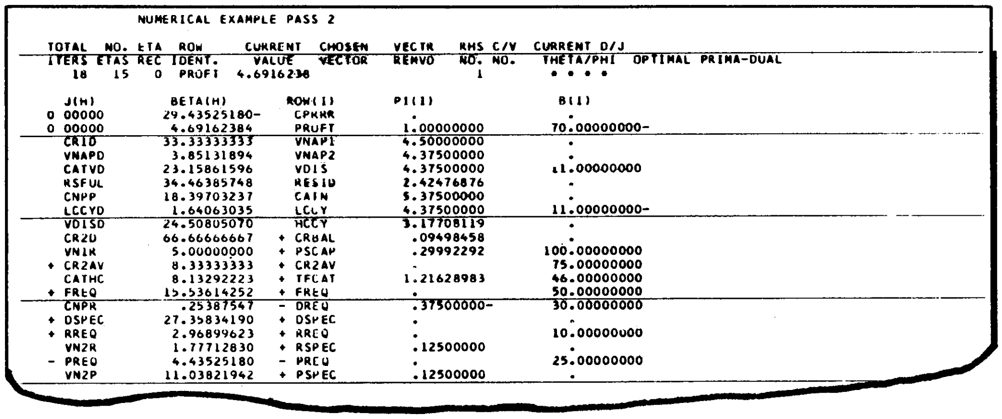
3.5: THE SOLUTION

The information provided by the computer falls into a number of categories, beginning with the statement of the optimum policy and continuing with a wide variety of data on changes in the system.



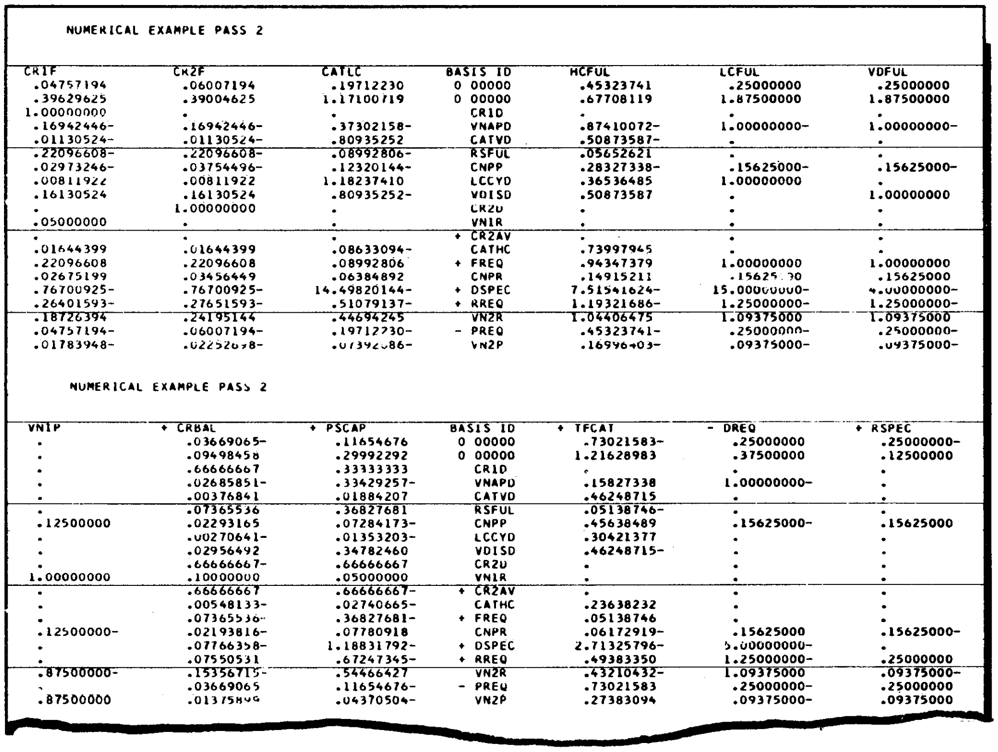
	ANNOUNCEMENT	COLUMN NAME		ROW NAME	VALUE			
	1 6	7	12	13 18	19		24	25 30
1	SA	M	PLE R	E FINER	Y	MAT	R	IX
2	ROW ID						.	
3	2		0	CPRRR			0	PROFT
4	2		0	VNAP1			0	VNAP2
5	2		0	VDIS			0	RESID
6	2		0	CATN			0	LCCY
7	2		0	HCCY			+	CRBAL
8	2		+	PSCAP			+	CR2AV
9	2		+	TFCAT			+	FREQ
10	2		-	DREQ			+	DSPEC
11	2		+	RREQ			+	RSPEC
12	2		-	PREQ			+	PSPEC
13	MATRIX						.	
14			CR1F	PROFT		3	.	
15			CR1F	VNAP1	-		.	1
16			CR1F	VDIS	-		.	25
17			CR1F	RESID	-		.	6
18			CR1F	CRBAL		1	.	
19			CR1F	PSCAP		1	.	
20							.	
21							.	
22							.	
23			CNPP	PROFT	-	5	.	
24			CNPP	CPRRR		1	.	
25			CNPP	CATN		1	.	
26			CNPP	PREQ		1	.	
27			CNPP	PSPEC	-	3	.	
28	FIRST	B					.	
29				PROFT	-	70	.	
30				VDIS		11	.	
31				LCCY	-	11	.	
32				PSCAP		100	.	
33							.	
34							.	
35	EOF						.	

Figure 16. A portion of an input form for the data of the refinery example



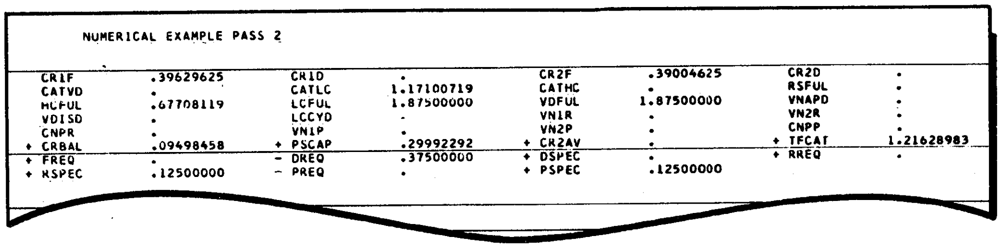
```
          NUMERICAL EXAMPLE PASS 2

  TOTAL  NO. ETA  ROW      CURRENT  CHOSEN   VECTR   RHS C/V  CURRENT D/J
  ITERS ETAS REC IDENT.     VALUE   VECTOR   REMVO   NO. NO.   THETA/PHI  OPTIMAL PRIMA-DUAL
    18    15   0  PROFT  4.6916238                      1         * * * *

    J(H)          BETA(H)        ROW(I)       PI(I)              B(I)
 0 00000        29.43525180-     CPRRR          .                  .
 0 00000         4.69162384      PROFT       1.00000000        70.00000000-
   CR1D         33.33333333      VNAP1       4.50000000          .
   VNAPD         3.85131894      VNAP2       4.37500000          .
   CATVD        23.15861596      VDIS        4.37500000        11.00000000
   RSFUL        34.46385748      RESID       2.42476876          .
   CNPP         18.39703237      CAIN        5.37500000          .
   LCCYD         1.64063035      LCCY        4.37500000        11.00000000-
   VDISD        24.50805070      HCCY        3.17708119          .
   CR2D         66.66666667    + CRBAL        .09498458          .
   VN1R          5.00000000    + PSCAP        .29992292       100.00000000
 + CR2AV         8.33333333    + CR2AV         .                75.00000000
   CATHC         8.13292223    + TFCAT       1.21628983        46.00000000
 + FREQ         15.53614252    + FREQ          .                50.00000000
   CNPR           .25387547    - DREQ         .37500000-       30.00000000
 + DSPEC        27.35834190    + DSPEC         .                  .
 + RREQ          2.96899623    + RREQ          .                10.00000000
   VN2R          1.77712830    + RSPEC        .12500000          .
 - PREQ          4.43525180    - PREQ          .                25.00000000
   VN2P         11.03821942    + PSPEC        .12500000          .
```

Figure 17. Computer output giving the optimum policy and marginal values



```
      NUMERICAL EXAMPLE PASS 2

 CR1F           CR2F            CATLC          BASIS ID      HCFUL          LCFUL          VDFUL
  .04757194      .06007194       .19712230     0 00000        .45323741      .25000000      .25000000
  .39629625      .39004625      1.17100719     0 00000        .67708119     1.87500000     1.87500000
 1.00000000      .               .               CR1D          .              .              .
  .16942446-     .16942446-      .37302158-      VNAPD        .87410072-    1.00000000-    1.00000000-
  .01130524-     .01130524-      .80935252       CATVD        .50873587-     .              .
  .22096608-     .22096608-      .08992806-      RSFUL        .05652621      .              .
  .02973246-     .03754496-      .12320144-      CNPP         .28327338-     .15625000-     .15625000-
  .00811922      .00811922      1.18237410       LCCYD        .36536485     1.00000000      .
  .16130524      .16130524       .80935252-      VDISD        .50873587      .             1.00000000
  .             1.00000000       .               CR2D          .              .              .
  .05000000      .               .               VN1R          .              .              .
  .              .               .             + CR2AV         .              .              .
  .01644399      .01644399       .08633094-      CATHC        .73997945      .              .
  .22096608      .22096608       .08992806     + FREQ         .94347379     1.00000000     1.00000000
  .02675199      .03456449       .06384892       CNPR         .14915211      .15625000      .15625000
  .76700925-     .76700925-    14.49820144-    + DSPEC       7.51541624-   15.00000000-    4.00000000-
  .26401593-     .27651593-      .51079137-    + RREQ        1.19321686-    1.25000000-    1.25000000-
  .18726394      .24195144       .44694245       VN2R        1.04406475     1.09375000     1.09375000
  .04757194-     .06007194-      .19712230-    - PREQ         .45323741-     .25000000-     .25000000-
  .01783948-     .02252698-      .07392086-      VN2P         .16996403-     .09375000-     .09375000-

      NUMERICAL EXAMPLE PASS 2

 VN1P          + CRBAL         + PSCAP          BASIS ID    + TFCAT         - DREQ         + RSPEC
  .              .03669065-      .11654676     0 00000        .73021583-     .25000000      .25000000-
  .              .09498458       .29992292     0 00000       1.21628983      .37500000      .12500000
  .              .66666667       .33333333       CR1D          .              .              .
  .              .02685851-      .33429257-      VNAPD        .15827338     1.00000000-     .
  .              .00376841       .01884207       CATVD        .46248715      .              .
  .              .07365536       .36827681       RSFUL        .05138746-     .              .
  .12500000      .02293165       .07284173-      CNPP         .45638489      .15625000-     .15625000
  .              .00270641-      .01353203-      LCCYD        .30421377      .              .
  .              .02956492       .34782460       VDISD        .46248715-     .              .
  .              .66666667-      .66666667       CR2D          .              .              .
 1.00000000      .10000000       .05000000       VN1R          .              .              .
  .              .66666667       .66666667-    + CR2AV         .              .              .
  .              .00548133-      .02740665-      CATHC        .23638232      .              .
  .              .07365536-      .36827681-    + FREQ         .05138746      .              .
  .12500000-     .02193816-      .07780918       CNPR         .06172919-     .15625000      .15625000-
  .              .07766358-     1.18831792-    + DSPEC       2.71325796-    5.00000000-     .
  .              .07550531       .67247345-    + RREQ         .49383350     1.25000000-     .25000000
  .87500000-     .15356715-      .54466427       VN2R         .43210432-    1.09375000      .09375000-
  .              .03669065       .11654676-    - PREQ         .73021583      .25000000-     .25000000
  .87500000      .01375899       .04370504-      VN2P         .27383094      .09375000-     .09375000
```

Figure 18. Computer output giving the reallocations caused by forcing one unit of activities that are at the zero level in the optimum policy



```
        NUMERICAL EXAMPLE PASS 2

   CR1F         .39629625     CR1D          .            CR2F        .39004625     CR2D          .
   CATVD        .             CATLC       1.17100719     CATHC       .             RSFUL         .
   HCFUL        .67708119     LCFUL       1.87500000     VDFUL      1.87500000     VNAPD         .
   VDISD        .             LCCYD         .            VN1R        .             VN2R          .
   CNPR         .             VN1P          .            VN2P        .             CNPP          .
 + CRBAL        .09498458   + PSCAP        .29992292   + CR2AV       .           + TFCAT       1.21628983
 + FREQ         .           - DREQ         .37500000   + DSPEC       .           + RREQ          .
 + RSPEC        .12500000   - PREQ          .          + PSPEC      .12500000
```

Figure 19. The reduced costs

The Optimum Policy

Figure 17 shows how the results might be printed by a linear programming system for a computer. We concentrate on the columns headed J(H) and BETA(H), which give, respectively, the variable names (activities) and their optimum values. Any variables not listed have optimum values of zero. We see, for instance, that CR1D (crude 1 to distillate) has an optimum value of 33.3 MB/D, and that CR2D (crude 2 to distillate) has a value of 66.7 MB/D. CR1F and CR2F (crude 1 to fuel and crude 2 to fuel) do not appear: the optimum policy is not to run the pipe still for fuel at all.

The maximum profit for this policy ($4,692 per day) is given at the top of Figure 17.

Changes in Allocation

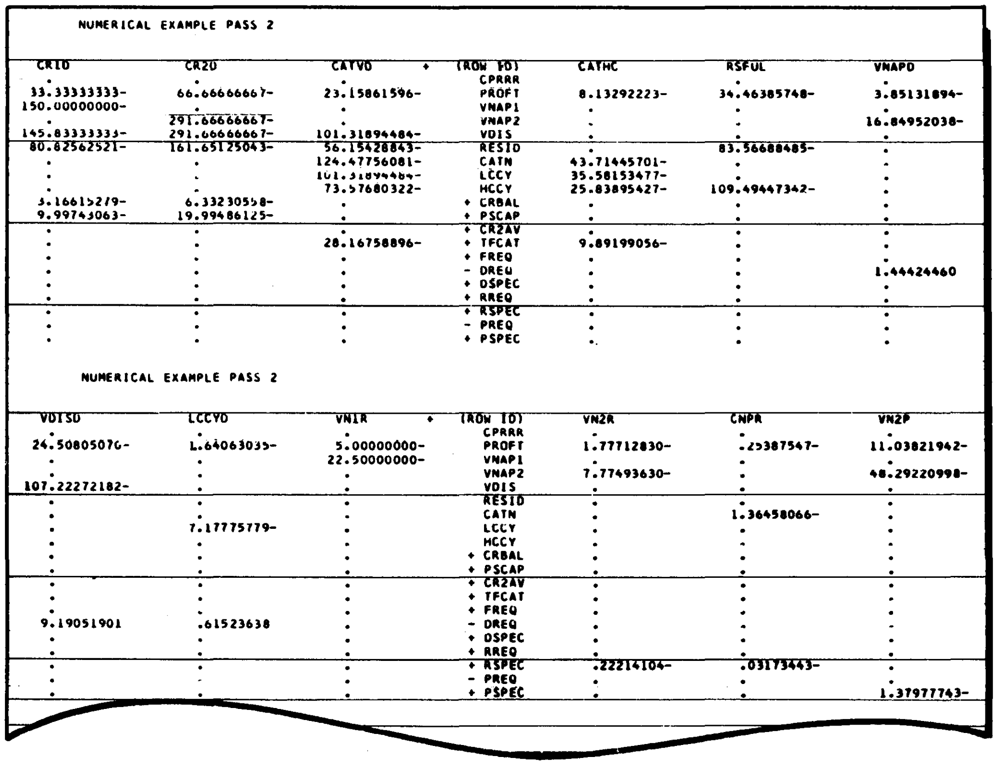
Should it be desirable to force a unit of some zero-level activity into the solution, other activity levels will change; we would like to decrease the profit as little as possible. The changes in allocation are shown in Figure 18. For example, forcing 1 MB/D of crude 1 to be run for fuel will cause, among other things, a decrease in crude 1 to distillate of 1 MB/D. This causes, in turn, a decrease in regular gasoline production of .264 MB/D, or 264 barrels per day. Heating oil production is maintained; fuel production goes up by 221 barrels per day; and premium gasoline is increased by 48 barrels per day.

Marginal Values

The marginal value of each restriction is computed; these are listed under the heading PI (I) in Figure 17. They are the changes in CURRENT VALUE (profit) per unit increase in the right-hand side constant of the corresponding constraint. The initial values of the right-hand sides are shown under B(I). For example, the pipe-still capacity (PSCAP) is initially 100 MB/D. If this were increased by 1 MB/D, the overall profit would be $.300 M/D, or $300, greater.

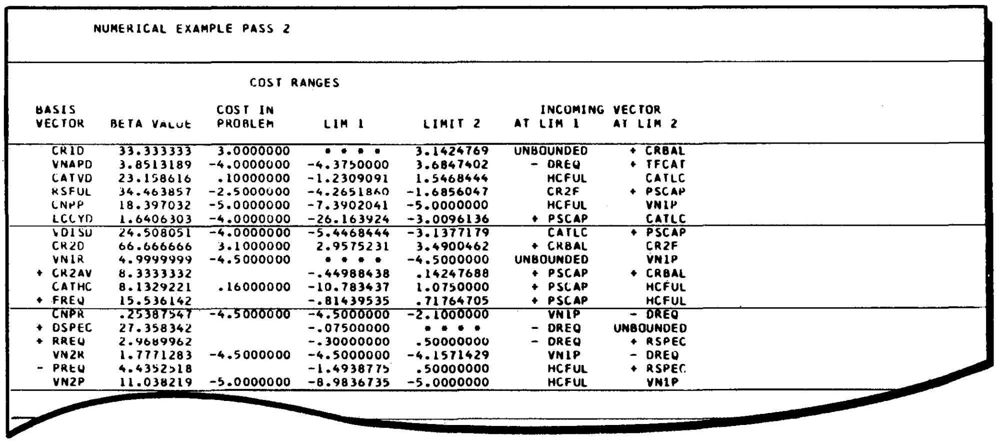
Reduced Costs

The reduced cost of alternative allocations is given in Figure 19 and is also a part of Figure 18. These values are the net cost of the exchanges required to satisfy all constraints when introducing a unit of an activity that is at the zero level in the optimum policy. For instance, the net effect of all the changes required to force 1 MB/D of crude 1 to be run for fuel is to increase overall costs (and therefore decrease profits) by $.396 M/D (.396 thousand dollars per day). These reduced costs can also be interpreted as the amount by which the cost of an activity would have to be reduced to make the introduction of that activity break even.



```
        NUMERICAL EXAMPLE PASS 2

   CR1D           CR2D           CATVD       +  (ROW ID)    CATHC          RSFUL          VNAPD
     .              .              .              CPRRR                       .              .
 33.33333333-   66.66666667-   23.15861596-       PROFT   8.13292223-   34.46385748-    3.85131894-
150.00000000-       .              .              VNAP1       .              .              .
     .         291.66666667-       .              VNAP2       .              .         16.84952038-
145.83333333-  291.66666667-  101.31894484-       VDIS        .              .              .
 80.82562521-  161.65125043-   56.15428843-       RESID       .         83.56688485-        .
     .              .         124.47756081-       CATN    43.71445701-       .              .
     .              .         101.31894484-       LCCY    35.58153477-       .              .
     .              .          73.57680322-       HCCY    25.83895427-  109.49447342-       .
  3.16615279-    6.33230558-       .            + CRBAL       .              .              .
  9.99743063-   19.99486125-       .            + PSCAP       .              .              .
     .              .              .            + CR2AV       .              .              .
     .              .          28.16758896-     + TFCAT    9.89199056-       .              .
     .              .              .            + FREQ        .              .              .
     .              .              .            - DREQ        .              .          1.44424460
     .              .              .            + DSPEC       .              .              .
     .              .              .            + RREQ        .              .              .
     .              .              .            + RSPEC       .              .              .
     .              .              .            - PREQ        .              .              .
     .              .              .            + PSPEC       .              .              .

        NUMERICAL EXAMPLE PASS 2

   VDISD          LCCYD          VN1R        +  (ROW ID)    VN2R           CNPR           VN2P
     .              .              .              CPRRR       .              .              .
 24.50805070-    1.64063035-    5.00000000-       PROFT    1.77712830-    .25387547-    11.03821942-
     .              .          22.50000000-       VNAP1       .              .              .
     .              .              .              VNAP2    7.77493630-       .         48.29220998-
107.22272182-       .              .              VDIS        .              .              .
     .              .              .              RESID       .              .              .
     .              .              .              CATN        .          1.36458066-        .
     .           7.17775779-       .              LCCY        .              .              .
     .              .              .              HCCY        .              .              .
     .              .              .            + CRBAL       .              .              .
     .              .              .            + PSCAP       .              .              .
     .              .              .            + CR2AV       .              .              .
     .              .              .            + TFCAT       .              .              .
     .              .              .            + FREQ        .              .              .
  9.19051901     .61523638         .            - DREQ        .              .              -
     .              .              .            + DSPEC       .              .              .
     .              .              .            + RREQ        .              .              .
     .              .              .            + RSPEC    .22214104-    .03173443-         .
     .              -              .            - PREQ        .              .              .
     .              .              .            + PSPEC       .              .          1.37977743-
```

Figure 20. Sensitivity of profit to changes in constraint coefficients



```
        NUMERICAL EXAMPLE PASS 2

                          COST RANGES

 BASIS                  COST IN                                  INCOMING VECTOR
 VECTOR    BETA VALUE   PROBLEM       LIM 1       LIMIT 2     AT LIM 1    AT LIM 2

   CR1D    33.333333   3.0000000    * * * *     3.1424769   UNBOUNDED   + CRBAL
   VNAPD   3.8513189  -4.0000000  -4.3750000    3.6847402     - DREQ    + TFCAT
   CATVD   23.158616   .10000000  -1.2309091    1.5468444       HCFUL     CATLC
   RSFUL   34.463857  -2.5000000  -4.2651860   -1.6856047       CR2F    + PSCAP
   CNPP    18.397032  -5.0000000  -7.3902041   -5.0000000       HCFUL     VN1P
   LCCYD   1.6406303  -4.0000000  -26.163924   -3.0096136     + PSCAP     CATLC
   VDISD   24.508051  -4.0000000  -5.4468444   -3.1377179       CATLC   + PSCAP
   CR2D    66.666666   3.1000000   2.9575231    3.4900462     + CRBAL     CR2F
   VN1R    4.9999999  -4.5000000    * * * *    -4.5000000   UNBOUNDED     VN1P
 + CR2AV   8.3333332              -.44988438    .14247688     + PSCAP   + CRBAL
   CATHC   8.1329221   .16000000  -10.783437    1.0750000     + PSCAP     HCFUL
 + FREQ    15.536142              -.81439535    .71764705     + PSCAP     HCFUL
   CNPR    .25387547  -4.5000000  -4.5000000   -2.1000000       VN1P    - DREQ
 + DSPEC   27.358342              -.07500000    * * * *       - DREQ    UNBOUNDED
 + RREQ    2.9689962              -.30000000    .50000000     - DREQ    + RSPEC
   VN2R    1.7771283  -4.5000000  -4.5000000   -4.1571429       VN1P    - DREQ
 - PREQ    4.4352518              -1.4938775    .50000000       HCFUL   + RSPEC
   VN2P    11.038219  -5.0000000  -8.9836735   -5.0000000       HCFUL     VN1P
```

Figure 21. Cost ranges of variables in the optimum policy

Sensitivity of Profit to Constraint Coefficients

Figure 20 gives the effect on profit of changes in the constraint coefficients. Look first at the row for PROFT: this is simply the effect on profit of a unit increase in costs. For instance, we see in Figure 17 that the optimum policy is to run 33.3 MB/D of crude 1 to distillate. If the cost of crude 1 were increased by one unit ($1 per barrel), the increase in cost would be $33.3 M/D (33.3 thousand dollars per day).

The reduction in profit would be the same amount. The number in Figure 20 appears with a minus sign, indicating a decrease in profit. Now consider the result given for the sensitivity of profit to the yield of residuum from crude when running for maximum distillate. The value of the coefficient (from Table 2) is 0.4, which appears with a minus sign in Figure 16 as the first entry in the row named RESID. Now consider Figure 20: looking at the column headed CR1D and the row named RESID, we see a sensitivity of -80.8. This means that the profit is reduced at the rate of 80.8 M/D per unit of increase in the constraint coefficient. This rate (as we saw in section 2.5) applies only to small increases. Suppose the yield of residuum from the crude 1 run for maximum distillate were increased by 1%; in that case, the profit would be increased by 1% of 80.8 M/D, or $808.

Ranges of Optimum-Policy Variables

The ranges of validity of the values for the variables in the optimum policy are shown in Figure 21. This exhibit shows, for each variable in the optimum policy, its name, the optimum value (BETA VALUE), the original cost, the lower limit (LIM 1) and the upper limit (LIM 2) of validity of this cost, the variable that would come into the basis if the cost of this one went below the lower limit (INCOMING VECTOR AT LIM 1), and the variable that would enter the basis if the cost of this one went above the upper limit (INCOMING VECTOR AT LIM 2).

For example, the cost of cracking virgin distillate (CATVD) was originally $0.10 per barrel. Figure 21 states that it could vary between -1.23 and 1.55 without changing the optimum policy. The policy is thus quite insensitive to the cost of this process. On the other hand, the policy is rather sensitive to the cost of virgin distillate sent to heating oil. In some cases, there is no lower or no upper limit to the range of validity; these cases are indicated appropriately in Figure 21.

Marginal Value Ranges

Figure 22 shows the ranges of validity of the marginal values of the right-hand sides (which were given in Figure 17). The column headed PI VALUE repeats the marginal value of each right-hand side for the row named; it then gives the MINIMUM VALUE and the MAXIMUM VALUE of the range of validity. It also lists the variables that would leave the basis if these ranges were exceeded.

For example, the amount of light-cycle stock sent to storage (LCCY) could vary from 9,176 B/D to 12,624 B/D without requiring a change in the solution to maintain feasibility. At the upper value, there is no more virgin naphtha 2 available to blend to regular gasoline; beyond the lower value, the contamination specification on heating oil would be violated.

3.6: PARAMETRIC PROGRAMMING

The final aspect of this example is a parametric programming run to determine the effect of a variation in the market price of premium gasoline. As we saw in section 2.6, this means that we let the price vary continuously, regardless of basis changes, to see how the optimum policy would change.

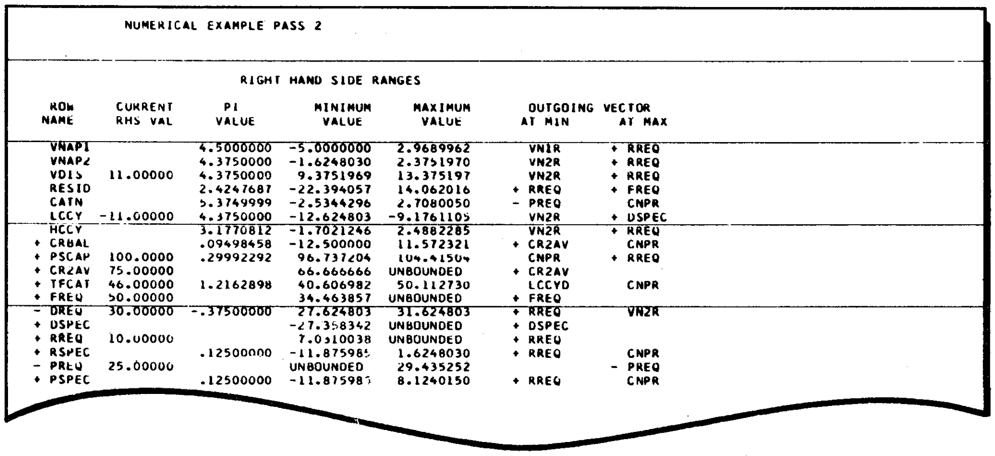
Calling for such a computer run requires filling out some appropriate forms, which we will not display since they vary for different computer systems. The computer output consists of a printout at each basis change. The printout lists the variables in the basis and their (new) optimum values, together with the right-hand sides and their (new) marginal values.

Thus far, parametric programming has been discussed primarily in terms of simultaneous variation of several right-hand sides. In fact, it may be used either for the right-hand sides or for the objective function, and it is entirely possible to vary only one thing at a time. In this example we wish to vary just one of the coefficients in the objective function: the price of premium gasoline. With one parametric programming run we can investigate the influence of a drop in price; with another, the influence of a rise. The results of such runs can be summarized as follows:

When the price of premium gasoline is reduced, there is no change in the solution until it reaches $4.50 per barrel, the same as the price of regular gasoline. At this point, 2,969 barrels per day are shifted from premium to regular, which fills the market demand for regular. A further reduction in premium price to $4 per barrel, the same as heating oil, causes premium volume to drop to the minimum allowable--25 MB/D. No further change in the policy occurs if the price drops below $4 per barrel. On the other hand, an increase in premium price to $6.49 per barrel brings out 772 barrels per day of added premium gasoline. At $6.50 per barrel, premium production goes to a maximum value of 31,805 barrels per day. No further changes occur in the solution, no matter how high the price goes.

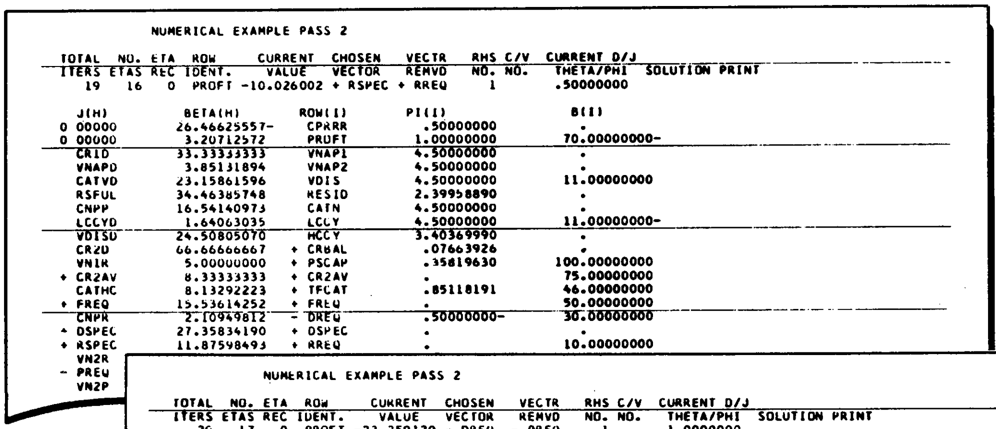
Figures 23 through 26 are the computer printouts at the various points of solution change. Figure 27 is a plot of the premium volume and the marginal values of several variables as a function of premium price.

A parametric programming run can be produced by the computer in a small percentage of the time required for the original solution. Considering the value to management in investigating a wide variety of possible changes in the operating system, it is not uncommon for many hundreds of parametric programming runs to be made.

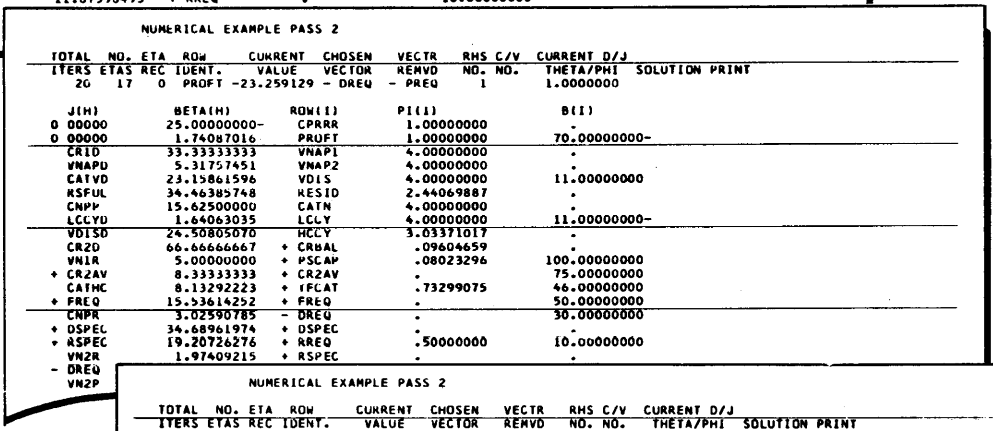


```
          NUMERICAL EXAMPLE PASS 2

                             RIGHT HAND SIDE RANGES

   ROW      CURRENT      PI         MINIMUM     MAXIMUM       OUTGOING VECTOR
   NAME     RHS VAL     VALUE        VALUE       VALUE        AT MIN     AT MAX

   VNAP1               4.5000000  -5.0000000  2.9689962       VN1R      + RREQ
   VNAP2               4.3750000  -1.6248030  2.3751970       VN2R      + RREQ
   VDIS    11.00000    4.3750000   9.3751969  13.375197       VN2R      + RREQ
   RESID               2.4247687  -22.394057  14.062016     + RREQ      + FREQ
   CATN                5.3749999  -2.5344296  2.7080050     - PREQ        CNPR
   LCCY   -11.00000    4.3750000  -12.624803  -9.1761105      VN2R      + DSPEC
   HCCY                3.1770812  -1.7021246  2.4882285       VN2R      + RREQ
 + CRBAL               .09498458  -12.500000  11.572321     + CR2AV       CNPR
 + PSCAP   100.0000    .29992292   96.737204  104.41504       CNPR      + RREQ
 + CR2AV   75.00000                66.666666  UNBOUNDED     + CR2AV
 + TFCAT   46.00000    1.2162898   40.606982  50.112730       LCCYD       CNPR
 + FREQ    50.00000                34.463857  UNBOUNDED     + FREQ
 - DREQ    30.00000   -.37500000   27.624803  31.624803     + RREQ        VN2R
 + DSPEC                          -27.358342  UNBOUNDED     + DSPEC
 + RREQ    10.00000                7.0510038  UNBOUNDED     + RREQ
 + RSPEC               .12500000  -11.875985  1.6248030     + RREQ        CNPR
 - PREQ    25.00000                UNBOUNDED  29.435252                 - PREQ
 + PSPEC               .12500000  -11.875985  8.1240150     + RREQ        CNPR
```

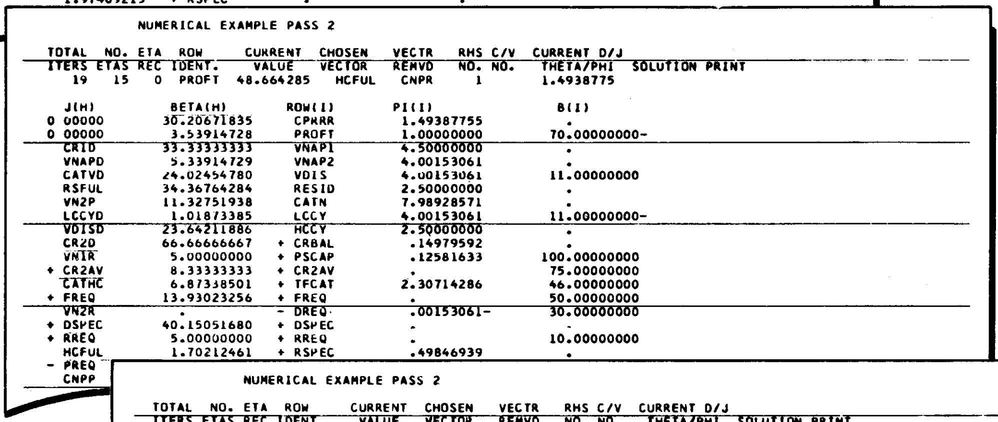
Figure 22. Ranges of validity of right-hand sides



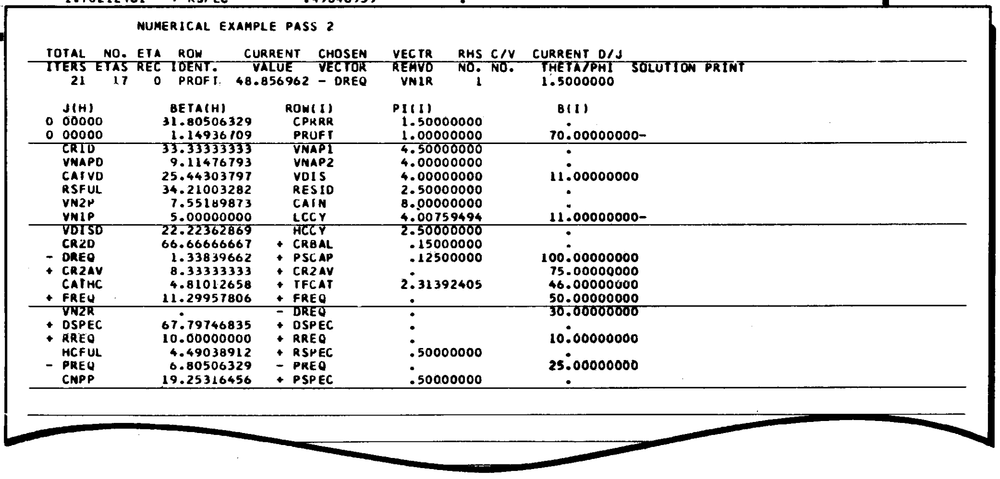
```
             NUMERICAL EXAMPLE PASS 2

TOTAL  NO. ETA  ROW     CURRENT  CHOSEN   VECTR   RHS C/V  CURRENT D/J
ITERS ETAS REC IDENT.    VALUE   VECTOR   REMVD   NO. NO.   THETA/PHI  SOLUTION PRINT
  19   16   0  PROFT -10.026002 + RSPEC + RREQ      1       .50000000

  J(H)          BETA(H)        ROW(I)        PI(I)             B(I)
0 00000       26.46625557-     CPRRR        .50000000           .
0 00000        3.20712572      PROFT       1.00000000        70.00000000-
  CRID        33.33333333      VNAP1       4.50000000           .
  VNAPD        3.85131894      VNAP2       4.50000000           .
  CATVD       23.15861596      VDIS        4.50000000        11.00000000
  RSFUL       34.46385748      RESID       2.39958890           .
  CNPP        16.54140973      CATN        4.50000000           .
  LCCYD        1.64063035      LCCY        4.50000000        11.00000000-
  VDISD       24.50805070      HCCY        3.40369990           .
  CR2D        66.66666667    + CRBAL        .07663926           .
  VN1R         5.00000000    + PSCAP        .35819630       100.00000000
+ CR2AV        8.33333333    + CR2AV          .                75.00000000
  CATHC        8.13292223    + TFCAT        .85118191         46.00000000
+ FREQ        15.53614252    + FREQ           .                50.00000000
  CNPR         2.10949812    - DREQ         .50000000-        30.00000000
+ DSPEC       27.35834190    + DSPEC          .                  .
+ RSPEC       11.87598493    + RREQ           .                10.00000000
  VN2R
- PREQ
  VN2P
```



```
             NUMERICAL EXAMPLE PASS 2

TOTAL  NO. ETA  ROW     CURRENT  CHOSEN   VECTR   RHS C/V  CURRENT D/J
ITERS ETAS REC IDENT.    VALUE   VECTOR   REMVD   NO. NO.   THETA/PHI  SOLUTION PRINT
  20   17   0  PROFT -23.259129 - DREQ  - PREQ      1       1.0000000

  J(H)          BETA(H)        ROW(I)        PI(I)             B(I)
0 00000       25.00000000-     CPRRR       1.00000000           .
0 00000        1.74087016      PROFT       1.00000000        70.00000000-
  CRID        33.33333333      VNAP1       4.00000000           .
  VNAPD        5.31757451      VNAP2       4.00000000           .
  CATVD       23.15861596      VDIS        4.00000000        11.00000000
  RSFUL       34.46385748      RESID       2.44069887           .
  CNPP        15.62500000      CATN        4.00000000           .
  LCCYD        1.64063035      LCCY        4.00000000        11.00000000-
  VDISD       24.50805070      HCCY        3.03371017           .
  CR2D        66.66666667    + CRBAL        .09604659           .
  VN1R         5.00000000    + PSCAP        .08023296       100.00000000
+ CR2AV        8.33333333    + CR2AV          .                75.00000000
  CATHC        8.13292223    + TFCAT        .73299075         46.00000000
+ FREQ        15.53614252    + FREQ           .                50.00000000
  CNPR         3.02590785    - DREQ           .                30.00000000
+ DSPEC       34.68961974    + DSPEC          .                  .
+ RSPEC       19.20726276    + RREQ         .50000000         10.00000000
  VN2R         1.97409215    + RSPEC          .                  .
- DREQ
  VN2P
```



```
             NUMERICAL EXAMPLE PASS 2

TOTAL  NO. ETA  ROW     CURRENT  CHOSEN   VECTR   RHS C/V  CURRENT D/J
ITERS ETAS REC IDENT.    VALUE   VECTOR   REMVD   NO. NO.   THETA/PHI  SOLUTION PRINT
  19   15   0  PROFT  48.664285   HCFUL   CNPR      1       1.4938775

  J(H)          BETA(H)        ROW(I)        PI(I)             B(I)
0 00000       30.20671835      CPRRR       1.49387755           .
0 00000        3.53914728      PROFT       1.00000000        70.00000000-
  CRID        33.33333333      VNAP1       4.50000000           .
  VNAPD        5.33914729      VNAP2       4.00153061           .
  CATVD       24.02454780      VDIS        4.00153061        11.00000000
  RSFUL       34.36764284      RESID       2.50000000           .
  VN2P        11.32751938      CATN        7.98928571           .
  LCCYD        1.01873385      LCCY        4.00153061        11.00000000-
  VDISD       23.64211886      HCCY        2.50000000           .
  CR2D        66.66666667    + CRBAL        .14979592           .
  VN1R         5.00000000    + PSCAP        .12581633       100.00000000
+ CR2AV        8.33333333    + CR2AV          .                75.00000000
  CATHC        6.87338501    + TFCAT       2.30714286         46.00000000
+ FREQ        13.93023256    + FREQ           .                50.00000000
  VN2R          .            - DREQ         .00153061-        30.00000000
+ DSPEC       40.15051680    + DSPEC          .                  .
+ RREQ         5.00000000    + RREQ           .                10.00000000
  HCFUL        1.70212461    + RSPEC        .49846939            .
- PREQ
  CNPP
```



```
             NUMERICAL EXAMPLE PASS 2

TOTAL  NO. ETA  ROW     CURRENT  CHOSEN   VECTR   RHS C/V  CURRENT D/J
ITERS ETAS REC IDENT.    VALUE   VECTOR   REMVD   NO. NO.   THETA/PHI  SOLUTION PRINT
  21   17   0  PROFT  48.856962 - DREQ    VN1R      1       1.5000000

  J(H)          BETA(H)        ROW(I)        PI(I)             B(I)
0 00000       31.80506329      CPRRR       1.50000000           .
0 00000        1.14936709      PROFT       1.00000000        70.00000000-
  CRID        33.33333333      VNAP1       4.50000000           .
  VNAPD        9.11476793      VNAP2       4.00000000           .
  CATVD       25.44303797      VDIS        4.00000000        11.00000000
  RSFUL       34.21003282      RESID       2.50000000           .
  VN2P         7.55189873      CATN        8.00000000           .
  VN1P         5.00000000      LCCY        4.00759494        11.00000000-
  VDISD       22.22362869      HCCY        2.50000000           .
  CR2D        66.66666667    + CRBAL        .15000000           .
- DREQ         1.33839662    + PSCAP        .12500000       100.00000000
+ CR2AV        8.33333333    + CR2AV          .                75.00000000
  CATHC        4.81012658    + TFCAT       2.31392405         46.00000000
+ FREQ        11.29957806    + FREQ           .                50.00000000
  VN2R          .            - DREQ           .                30.00000000
+ DSPEC       67.79746835    + DSPEC          .                  .
+ RREQ        10.00000000    + RREQ           .                10.00000000
  HCFUL        4.49038912    + RSPEC        .50000000            .
- PREQ         6.80506329    - PREQ           .                25.00000000
  CNPP        19.25316456    + PSPEC        .50000000            .
```

Figures 23, 24, 25, 26. Parametric programming run printouts

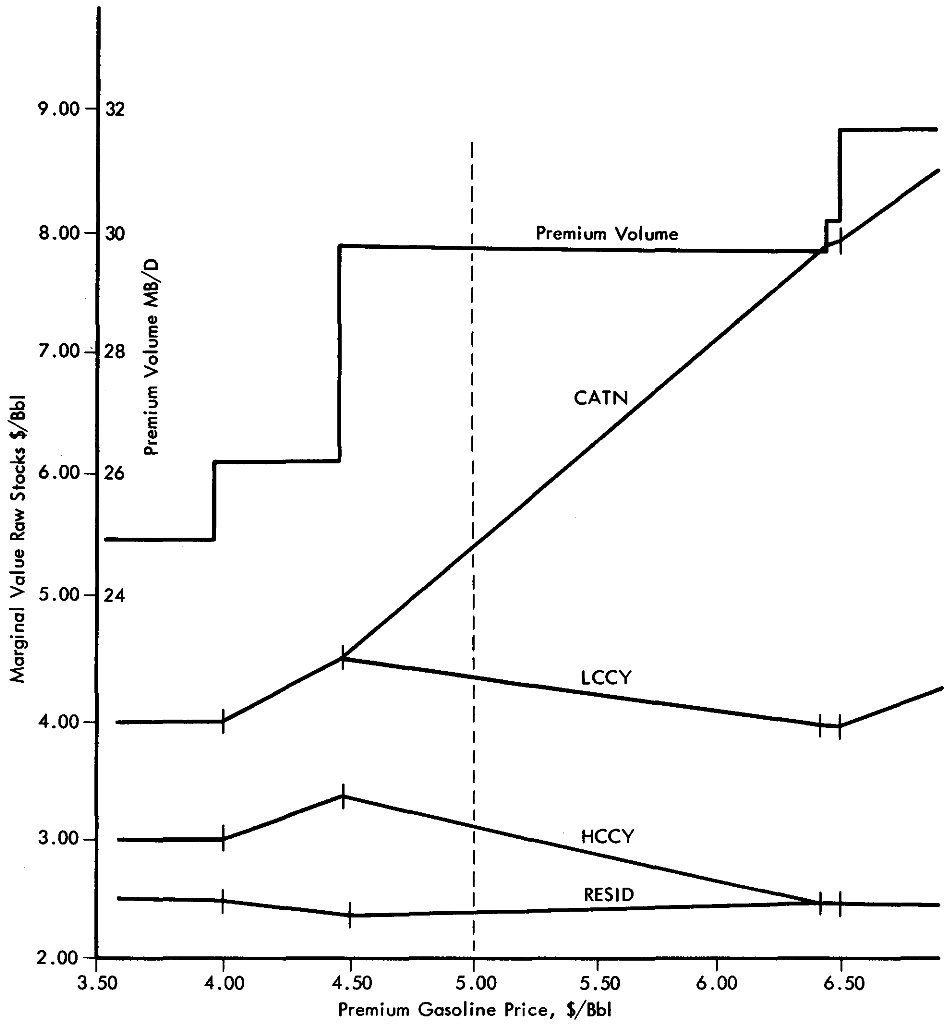


Figure 27. Graphic display showing some results of a parametric programming analysis of the effect of changes in the price of premium gasoline

CHAPTER 4: THE SIMPLEX METHOD

We have by now seen quite a bit about the usefulness of linear programming, and know something about how to apply the technique. In this chapter we shall investigate a little more closely how a linear programming problem can actually be solved by systematic algebraic methods.

The mathematics here will not be any more complicated than that used previously, namely, high school algebra. The reader who has a serious interest in linear programming will find a study of this chapter worth his effort, in that it will help develop a deeper insight into what the computer is doing in getting the kinds of results we have displayed previously.

Linear programming codes for computers do not do precisely what we shall show; short cuts and improvements are employed frequently. The fundamental ideas are the same, however.

4.1: GEOMETRICAL BACKGROUND

The techniques of the simplex method will be illustrated through a hypothetical two-activity, four-constraint problem. We make no attempt to show a particular business problem to which it might correspond; it should be clear by this point that the mathematical statement is representative of realistic problems that can arise in practice.

The problem:

1. Maximize $x_1 + 2x_2 = Z$
2. Subject to these constraints:

$$2x_1 + x_2 \leq 10$$

$$x_1 + x_2 \leq 6$$

$$-x_1 + x_2 \leq 2$$

$$-2x_1 + x_2 \leq 1$$

3. And to the implied constraints: $x_1 \geq 0$, $x_2 \geq 0$.

It will be useful to draw the graph of the constraints, because the fundamental approach to the solution can be nicely visualized thereby. (See Figure 28.)

The vertices in Figure 28 have been identified by letters, for comparison with some later algebraic results.

The first step is to convert the inequalities into equations by the introduction of slack variables. In prior examples, we called these slack variables s_1, s_2, etc.; as long as the meaning is clear, there is no reason not to use the same letter for slack

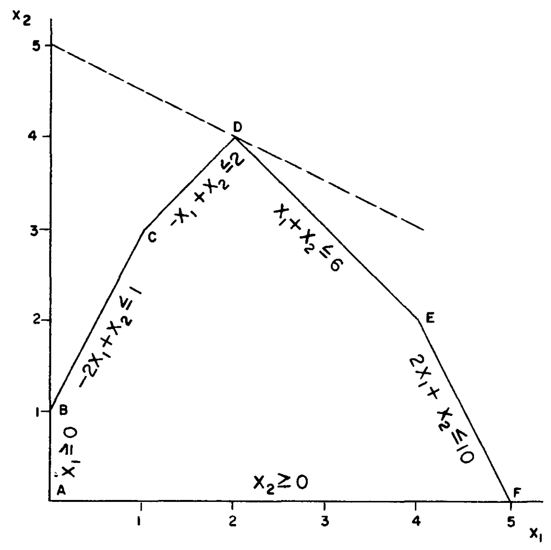


Figure 28. Graph of constraints for simplex example

variables as for the problem variables, using different subscripts.

Objective Function:

$$x_1 + 2x_2 \qquad = Z$$

Constraints:

$$\begin{array}{llr}
(1) & 2x_1 + x_2 + x_3 & = 10 \\
(2) & x_1 + x_2 \quad + x_4 & = 6 \\
(3) & -x_1 + x_2 \quad\quad + x_5 & = 2 \\
(4) & -2x_1 + x_2 \quad\quad\quad + x_6 & = 1
\end{array}$$

With four equations, a basis will consist of four variables that are permitted to be nonzero; the other two variables will be forced to be zero. Let us now perform an exercise that will in effect reveal the whole idea of the simplex method.

Table 6 is a list of the 15 possible ways to assign two variables to be zero and solve for the other four.

Solution Number	Variables x_1	x_2	x_3	x_4	x_5	x_6	Vertex
1	0	0	10	6	2	1	A
2	0	10	0	-4	-8	-9	
3	0	6	4	0	-2	-3	
4	0	2	8	4	0	-1	
5	0	1	9	5	1	0	B
6	5	0	0	1	7	11	F
7	6	0	-2	0	8	13	
8	-2	0	14	8	0	-3	
9	-1/2	0	11	13/2	3/2	0	
10	4	2	0	0	4	7	E
11	8/3	14/3	0	-4/3	0	5/3	
12	9/4	11/2	0	-7/4	-5/4	0	
13	2	4	2	0	0	1	D
14	5/3	13/3	7/3	0	-5/3	0	
15	1	3	5	2	0	0	C

Table 6. The 15 results of assigning two variables to be zero and solving for the other four. The letters under "Vertex" correspond to vertices in Figure 28.

Nine of the 15 solutions contain negative values of the variables and are therefore not feasible. The other six are not only feasible, but correspond exactly with the six vertices in Figure 28, and have been labeled accordingly.

Let us review. The six lettered lines in Table 6 are basic solutions: two variables were set equal to zero and the equations solved for the other four. They are also feasible solutions, since they satisfy the equations and involve only nonnegative values. Looking at the values of x_1 and x_2, which are plotted in Figure 28 (x_3 to x_6 are slacks), we see that basic feasible solutions correspond to vertices. This is equally true for problems with more than two activities, which we cannot graph in two dimensions. Naturally, in that case we must introduce an extended definition of what a vertex is, but this can be done.

Let us rearrange the basic solutions in Table 6 into order, from A to F, as in Table 7.

Vertex	Variables x_1	x_2	x_3	x_4	x_5	x_6
A	0	0	10	6	2	1
B	0	1	9	5	1	0
C	1	3	5	2	0	0
D	2	4	2	0	0	1
E	4	2	0	0	4	7
F	5	0	0	1	7	11

Table 7. The basic solutions of the illustrative problem

Inspect Table 7 carefully. Each pair of adjacent lines represents two adjacent vertices; how do the variables that constitute the basis in one vertex compare with those that constitute the basis in an adjacent vertex? The two "adjacent" bases have all but one variable in common. The basis for point A, for instance, is composed of x_3, x_4, x_5 and x_6; the basis for point B also contains x_3, x_4 and x_5, with x_6 in point A replaced by x_2 in point B.

4.2: ALGEBRAIC DEMONSTRATION

With this geometrical background, we can now state rather simply the broad outline of the simplex method.

1. Pick an initial feasible basis. If all the constraints are "less than or equal" types with positive right-hand sides, the slack variables immediately constitute such a basis; we shall see in section 4.3 what to do if this is not possible.
2. If necessary, perform some algebraic operations that will express the objective function entirely in terms of nonbasic variables. The variables that remain in the objective function after these operations will then all have the value zero, and the value of the objective function as transformed will therefore be constant.
3. Inspect the objective function to find the term with the largest positive coefficient. Giving this variable some positive value will increase the value of the objective function; this variable is to be introduced into the basis.
4. By a simple test that we shall see in a moment, determine which variable is to go out of the basis. Effecting the removal is a matter of some algebraic manipulations that constitute the bulk of the computational effort. These operations will modify the coefficients in the objective function,

so as once again to express it in terms of nonbasic variables only.

5. Inspect the objective function. If there are no terms with positive coefficients, there is no possible way to increase the value of the objective function by giving any nonbasic variable a positive value. The optimum has therefore been reached. If there are positive terms, repeat steps 3, 4 and 5.

This process is the algebraic equivalent of moving from vertex to vertex in such a way as to guarantee improvement at every step. Since there is a finite number of vertices, the process must terminate after a finite number of repetitions. Since improvement is guaranteed at every step, when finally there are no more positive terms in the objective function, the value of the objective function must be the largest it can possibly be. There can never be any question whether the result found by this method really is the best that could be determined.

Let us follow the process through in detail, in terms of the system of equations on page 37.

1. An initial feasible basis is provided by the slacks x_3, x_4, x_5 and x_6. Allowing them to have nonzero values and forcing x_1 and x_2 to be zero does satisfy the constraints. We have a basic solution (four nonzero and two zero variables) and it is feasible (the constraints are satisfied).

2. The objective function is then already expressed in terms of nonbasic variables. Its value is accordingly zero.

3. In the objective function, x_2 has the largest (positive) coefficient, so it is the variable that will be introduced into the basis to start improving it.

4. Making x_2 a basis variable is going to require transforming the system so that x_2 appears in only one equation. In doing so, with the system as it now stands, we will have to subtract some one equation from all of the others. But in this process we must not make the right-hand side of any equation go negative. The question is: Which equation could we subtract without making any right-hand side negative? Clearly, it will have to be the fourth equation, since it has the smallest right-hand side.

(As we shall see the next time around, things would be a trifle different if x_2 had different coefficients in the different equations.)

Again let us review. We are going to subtract equation (4) from equations (1), (2) and (3). After we have done so, x_2 will appear only in equation (4), but now x_6 will appear in all four equations. This transformation corresponds to introducing x_2 into the basis and removing x_6 from it.

We must also remove x_2 from the objective function. This is easily done by subtracting two times equation (4) from the objective function.

The result of all this is as follows:

$$\begin{array}{llllllll}
\text{(OF)} & 5x_1 & & & & & + 2x_6 & = Z - 2 \\
(5) & 4x_1 & & + x_3 & & & - x_6 & = 9 \\
(6) & 3x_1 & & & + x_4 & & - x_6 & = 5 \\
(7) & x_1 & & & & + x_5 & - x_6 & = 1 \\
(8) & -2x_1 & + x_2 & & & & + x_6 & = 1
\end{array}$$

Since everything on the left-hand side of the objective function is zero (nonbasic variables only), we have $0 = Z-2$, or $Z = 2$. The value of the objective function has been increased from zero to 2 by this change of basis. The basis is now $x_2 = 1$, $x_3 = 9$, $x_4 = 5$, $x_5 = 1$. Since x_1 is still zero, on the graph of Figure 28 we are at the point $x_1 = 0$, $x_2 = 1$, which is point B.

Looking for further improvement, we have in the objective function only one choice, since there is only one term with a positive coefficient: x_1.

Deciding which variable to remove from the basis is a little more complicated this time. Remember that we are going to have to eliminate x_1 from all equations except one. Equation (8) is not a candidate for the one to be left, since x_1 would then have an illegal negative value. In general, we ignore any equations in which the coefficient of the new basis variable is negative.

To see how we determine the variable to be eliminated from the basis, suppose that we divide each equation by the coefficient of x_1 in that equation: 4 in equation (5), 3 in (6), and 1 in (7). This means to divide every coefficient and the right-hand side in equation (5) by 4, etc. Having done this, all of the new coefficients of x_1 will be 1. Now we ask: Which right-hand side is the smallest? This is now the same question we asked the first time around, in order to determine which equation could be subtracted from all others without making any right-hand side negative.

The selection rule for the variable to be removed from the basis is usually stated in terms of these ratios. More formally: form the ratios of the right-hand sides to the coefficients of the variable to be introduced, not considering any negative coefficients. Eliminate the variable for which the ratio is the smallest.

In our case, equation (7) gives the smallest positive value, so the variable that is currently basic in that equation, x_5, will be removed. In terms of equations (5) to (8) before dividing by the coefficients of x_1, this means to multiply (7) by 4 and subtract from (5), multiply (7) by 3 and subtract from (6), multiply (7) by 2 and add to (8), and multiply (7) by 5 and subtract from the objective function.

The result is as follows:

$$\begin{array}{llr}
\text{(OF)} & \quad\quad\;\; - 5x_5 + 3x_6 = Z - 7 \\
(9) & \quad x_3 \;\; - 4x_5 + 3x_6 = 5 \\
(10) & \quad\;\; x_4 - 3x_5 + 2x_6 = 2 \\
(11) & x_1 \quad\quad\;\; + x_5 - x_6 = 1 \\
(12) & \; x_2 \quad\;\; + 2x_5 - x_6 = 3
\end{array}$$

The basis is now $x_1 = 1$, $x_2 = 3$, $x_3 = 5$, $x_4 = 2$. The value of the objective function has been increased to 7. We are at point C in Figure 28, and are evidently doing just what we said: approaching the optimum by a systematic process of moving from one vertex to an adjacent one, with guaranteed improvement at each step.

For the next improvement we shall add x_6 to the basis and remove x_4. (We had previously removed x_6 from the basis and now we are putting it back--but that is acceptable.) The result of the usual operations is as follows:

$$\begin{array}{ll}
\text{(OF)} & \quad\quad - \frac{3}{2}x_4 - 5x_5 \quad\quad = Z - 10 \\
(13) & \quad x_3 - \frac{3}{2}x_4 - \frac{1}{2}x_5 \quad\quad = 2 \\
(14) & \quad\quad\;\; \frac{1}{2}x_4 - \frac{3}{2}x_5 + x_6 = 1 \\
(15) & x_1 \quad\; + \frac{1}{2}x_4 - \frac{1}{2}x_5 \quad\quad = 2 \\
(16) & \; x_2 \quad + \frac{1}{2}x_4 + \frac{1}{2}x_5 \quad\quad = 4
\end{array}$$

We are now at point D on the graph, the optimum. The objective function now also tells us that this is the optimum: there are no positive terms, so that there is no variable we could introduce that would increase the objective function from its present value, 10. (Incidentally, the original objective function was $x_1 + 2x_2$: the present values of $x_1 = 2$, $x_2 = 4$ give this same value of 10, which checks.)

4.3: SUMMARY

This brief glimpse of the simplex method may be closed with a few indications of some practical computational aspects of the method.

We assumed in the exposition above that it was possible to pick an initial feasible basis by inspection, and suggested that the slacks would supply such a basis. This, of course, is true only if all the constraints are of the "less than or equal" type with nonnegative right-hand sides. If there are any actual equations (in practice, this is uncommon), there will naturally be no slack variable for that constraint. If, as occasionally happens, there are "greater than or equal" constraints, then the slacks are subtracted. Such a slack cannot be used as a basis variable because it would have a negative value.

The solution to this impasse is to assign additional variables to any constraint that does not already have a slack with a positive coefficient, this new variable being called an artificial variable. It is called artificial because the first step, in a preliminary phase called the feasibility solution, will be to eliminate such variables from the basis. This can be done easily enough by giving the artificial variables very large "costs" in the objective function. This will force them out of the basis, leading to some other basis that will be feasible. With the artificial variables out of the basis (their values are zero) and with the large costs keeping them out permanently, we have essentially the same problem we started with, except that now we have computed a feasible basis. From this point on we can ignore the artificial variables and carry on with the normal simplex procedures.

The computational techniques we have outlined in this chapter are not necessarily precisely those used in computer programs for linear programming. The simplex method, as we have shown it, is somewhat inefficient from a computing point of view: there are operations that can be eliminated, and there are ways to reduce the amount of computer storage required. Finally, the procedure as it has been revised for these reasons also turns out to be much more convenient when it comes to providing the kinds of additional information about a solution that were described in Chapter 2.

****

The reader who has digested the material in this manual, even if not quite completely, has gained a fairly thorough understanding of the power and methods of linear programming. With the guidance of manuals on particular computer programs for linear programming, he is ready to try applying linear programming to his own problems.

The reader who wants more discussion of application areas or of the mathematical techniques may find some of the references listed in the bibliography useful.

GLOSSARY

Activity.--A structural variable whose level is to be computed in a programming problem.

Activity level.--The value taken by a structural variable in an intermediate or final solution to a programming problem.

Agenda.--The set of control language statements used to prescribe a solution path or run procedure; an ordered list of the major operations constituting a procedure for solution or computer run. (This usage corresponds roughly to the ordinary "agenda" for a meeting.)

Agendum. The body of code called for execution by a control language statement (agendum call card).

Agendum call card.--A single agendum name and its parameters punched on one card in a stylized form; one item of the agenda --a control language statement calling for the execution of the named agendum. A set of agendum call cards is used to control a linear programming system, thus forming an agenda.

Algorithm.--A mathematical procedure whose routine application (iteratively or recursively) yields the solution to a particular class of problem.

Alpha (α).--Notation used to express a tableau or portion of it in terms of the current (as opposed to original) basis.

Alternate optima.--Distinct solutions to the same optimization problem.

A - matrix. --Notation used to denote the original coefficient matrix, $[a^i_j]$ or $[a_{ij}]$, possibly extended by some or all of the logical vectors. The functional row is usually included, as in the modified simplex array.

Artificial variables. --Auxiliary or logical variable restricted or bounded to zero for equation constraints, and unrestricted as to sign for objective function or other null constraints. The logical variables, artificials and slacks provide the "identity" basis needed for the start of the simplex algorithm. Artificial vectors may appear physically in the problem statement and/or machine representation, but are more often only a logical construction (device). Once the zero-valued artificial variables have been driven out of the basic set, or have achieved a zero value by simplex pivot steps, the simplex procedure assures that they will not become nonzero in subsequent iterations.

Artificial vector (primal).--The logical column vector associated with a primal artificial variable. It is a unit vector, with +1 in the row associated with the artificial variable, and zero elsewhere; hence, a logical vector as opposed to structural, and it may not be physically present in the computer matrix.

Assignment problem.--The problem of placing n elements into n cells, one element to a cell, at minimum cost, where the individual cost of putting element i in cell j is a given constant c_{ij}. The problem can be stated as a linear programming problem, or a transportation problem, and so solved. However, a much more efficient special-purpose algorithm exists for this problem, in which operations on the cost matrix (c_{ij}) lead to an increasing series of partial assignments. This algorithm, called the Hungarian method, terminates at the first complete assignment which is optimal.

Augmented matrix. --The coefficient matrix augmented by the column of right-hand-side constants; the same as the constraint matrix.

Balance equations.--Those constraints on a production process which express the equality of inflows and outflows of one sort of material at one point in the system. They are usually in the form of equations with a right-hand side of zero.

Basic feasible solution (primal).--A basic solution to the constraint equations in which the nonzero values of the primal variables (both structural and logical) meet all the variable-type and bound restrictions. The lowest-cost basic feasible solution is the optimum. See also "feasible solution".

Basic solution (primal).--A solution to the row constraint equations in which the number of nonzero primal variable values (both structural and logical) is at most equal to the number of equations. The row constraint equations include bounds, and for modified or Kuhn-Tucker simplex arrays, costs.

Basic variables (primal).--Those m primal variables, in an m-constraint linear programming problem, which are permitted to be nonzero and nonbounded at an iterative stage in the solution. The remaining variables (nonbasic variables) are fixed at zero. In a bounded variable code some variables may be fixed at their bounds.

Basis (dual).--A square submatrix (nonsingular) of the structural coefficient matrix extended vertically by an n x n identity matrix (which provides a logical unit row for each primal structural variable) whose rows are the row vectors associated with the current set of dual basis variables. The logical unit rows are seldom carried explicitly in the computer. In common usage only the structural rows are implied in reference to "dual basis".

Basis (primal).--A square submatrix (nonsingular) of the coefficient matrix, of maximum size, whose columns are the column vectors associated with the current set of basic variables. A presumed basis may be singular. Some codes substitute logical vectors for the dependent columns.

Basis headings list (primal).--The names or indexes of the vectors in the current basis in the same order as the activity levels (beta values); used to identify the levels with their corresponding variables.

Basis inverse.--The inverse of the basis matrix.

Basis matrix.--Same as "basis".

Basis row.--A row of the coefficient matrix corresponding to a potentially nonzero structural variable in the current dual solution.

Basis variables.--Same as "basic variables".

Basis vector.--A column of the coefficient matrix corresponding to a potentially nonzero variable (either structural or logical) in the current primal solution.

Beta value.--The level of a basis primal variable; same as "activity level".

Beta vector.--A notation for the B-vector in terms of the current basis--for example, the basic or potentially nonzero portion of the solution only. Variables at their bounds are usually excluded.

Bounded variable.--A primal variable x_j in a linear programming problem which is required to satisfy a constraint of the form

$$0 \leq x_j \leq b, \text{ or } -b \leq x_j \leq 0, \text{ or } b_1 \leq x_j \leq b_2$$

where b is some positive constant and $b_1 \leq b_2$ is implied. Some linear programming codes can handle bounded variables directly; others require that these column-bound restrictions be explicitly stated as constraint rows of the matrix. Dual variables are not ordinarily bounded directly. However, note that a non-negative structural variable, consisting of a minus one in a "less than or equal" row i and a plus b_i in the cost row, effectively bounds the dual variable, π^i, for that row as $0 \leq \pi^i \leq b_i$.

Break-even price.--Same as "marginal value" (q.v.), but usually only applied to balance equations.

B-vector.--Notation often used for the right-hand side (constant column) in a linear programming matrix.

Capacity.--The right-hand side of some constraint of the form

$$a_1x_1 + a_2x_2 + \ldots + a_kx_k \leq b$$

where the constraint applies to a measure of the capacity of a process unit.

Change row.--A row vector, Q, specifying the proportionate changes in the cost-row vector in parametric programming on the objective function. If C is the original cost row, and t is the parameter, the moving value of the cost row is expressed as C + tQ.

Change vector.--A column vector, C, specifying the proportionate changes in the right-hand side (B-vector) in parametric programming. If B is the original column and t is the parameter, the moving value of the column is expressed as B + tC.

Code.--A set of computer instructions which carry out a particular calculation. A computer program.

Coefficient.--A constant multiplying a variable, as the "3" in 3x.

Coefficient matrix.--The matrix of left-hand-side coefficients in a system of linear equations. It is to be distinguished from the matrix obtained by appending the right-hand side, which is called the "augmented matrix" of the system. It may be thought of as including a full set of logical vectors to convert inequality constraints to equations, and in the case of the modified simplex array it also contains the objective function coefficients.

Col ID. --Generally, a name, index or mark to distinguish a particular column of a matrix.

Column. --All the elements of a single vertical line of a matrix, taken in given order. An m x 1 submatrix of an m x n matrix.

Column error. --The numerical value of

$$(\pi_1, \ldots, \pi_m)\, a_k + c_k,$$

or in the modified simplex array

$$(\pi', \ldots \pi_m)\, \bar{a}_k$$

where

$$(\pi_1, \ldots, \pi_m)$$

is the vector of simplex multipliers, a_k is the column for the basic variable x_k, and c_k is the coefficient of x_k in the objective function; in the modified array, π' is the scale factor and $\bar{a}_k$ is extended by the objective row entry, c_k. With no rounding error, the above expressions would have a value of zero.

Column index. --A numerical equivalent of the column identification, assigned by the computer for the purpose of saving space and making tests easier.

Column relation. --A relation that is equal, less than or equal, greater than or equal, or null for each constraint (matrix column) of the dual problem. The column relations correspond one-to-one with the complementary primal variable-type restrictions as follows:

$$\sum_i \pi_i a_j^i = c_j \text{ (equal)} \Leftrightarrow x_j \gtreqless 0$$

$$\sum_i \pi_i a_j^i \geq c_j \text{ (less than or equal)} \Leftrightarrow x_j \leq 0$$

(nonpositive)

$$\sum_i \pi_i a_j^i \leq c_j \text{ (greater than or equal)} \Longleftrightarrow x_j \geq 0$$

(non-negative)

$$\sum_i \pi_i a_j^i \gtrless c_j \text{ (null)} \Longleftrightarrow x_j = F^L \text{ (fixed)}$$

where

π_i is a dual variable

a_j^i is the coefficient matrix

c_j is the row of objective coefficients

x_j is a primal variable

F^L = fixed level

Column slack. --The amount by which the dual restriction is exceeded--for example, dual slack. This is also the evaluator of this column in the primal problem--that is, the reduced cost, d_j, etc.

Column vector. --One column of a matrix or a matrix consisting of a single column. The elements of the column are interpreted as the components of the vector.

Complementary relation. --See complementary variable.

Complementary slack. --See complementary variable.

Complementary variable. --A relation (row) of the primal problem has an associated primal logical variable (slack or artificial). This primal relation or slack is said to be complementary to the structural dual variable for this row, or its type restriction. Similarly, a relation (column) of the dual problem has an associated dual logical variable (slack). The dual relation (or its slack) is said to be complementary to the structural primal variable for this column. (See "column relation".)

Composite algorithms. --When neither the basic solution nor the dual solution generated by its simplex multipliers remains feasible, the corresponding algorithm is called composite. (Dantzig)

Concave function. --(1) Geometrically, a function whose graph is never below the chords joining any two points on the graph. (2) A concave constraint (with respect to the objective function) is one where the functional value on the constraint is never "better" than the value on the chord joining two points on the constraint. Thus, only the end points of a piecewise linear approximation will be in the basis no matter how many subdivisions there are between them. (The efficiency of separable programming for this case provides the prime justification for that method.)

(3) A function f such that

$$t_1 f(x_1) + t_2 f(x_2) \leq f(t_1 x_1 + t_2 x_2) \text{ for all}$$

$$t_1 \geq 0,\ t_2 \geq 0,\ t_1 + t_2 = 1 \text{ and all } x_1 \text{ and } x_2$$

in the convex domain of definition of f. (4) The negative of a convex function.

Condition of a matrix.--A convenient measure is the absolute ratio of the largest to smallest nonzero element--the larger the ratio, the less well-conditioned the matrix. An ill-conditioned matrix may behave digitally as if it were singular. Mathematically, a matrix is said to be ill-conditioned, or badly scaled if the absolute value ratio of the largest to smallest eigenvalue is large.

Constant vector.--The right-hand sides of the set of linear inequalities; the B-vector. By convention the linear relations of an LP problem are so arranged that all variables and their coefficients appear on the left, and the column of constants appears on the right.

Constraint.--An equation or inequality relating the variables in an optimization problem. A feasible (primal) solution must satisfy all the constraints including column-type restrictions (bounds, non-negativity, etc).

Constraint matrix.--In linear programming, the augmented matrix of the constraint equations. It is the matrix formed by the coefficient columns, or left-hand sides, and the column of constants.

Control language.--The language used to prescribe the course of a linear programming run on a computer. The language consists mainly of verbs (agendum names), which call in a body of code (program or subroutine) embodying the desired algorithm for execution.

Convergence of an algorithm.--An algorithm is said to converge if it is certain to yield its solution in a finite number of steps. It is a much stronger requirement than the mathematical convergence of the sequence of obtained function values.

Convex function.--(1) Geometrically, a function whose graph is never above the chords ioining two points on the graph. (2) A convex constraint with respect to the objective function is one where the value on the constraint is never "worse" than the value on the chord. Thus, any degree of accuracy can be obtained by making a sufficiently fine piecewise linear approximation. (Note that some functions are neither concave nor convex.) (3) A function f such that

$$t_1 f(x_1) + t_2 f(x_2) \geq f(t_1 x_1 + t_2 x_2) \text{ for all}$$

$$t_1 \geq 0,\ t_2 \geq 0,\ t_1 + t_2 = 1 \text{ and all } x_1 \text{ and } x_2$$

in the convex domain of definitions of f. (4) The negative of a concave function.

Convex programming.--Optimization of a convex function over a convex region (set). Linear programming and quadratic programming are special cases of convex programming.

Convex set.--Geometrically, a set of points (region) that contains all the points on the line segment joining any two points of the set. A point which does not lie on any line segment joining two other points of the set is called an extreme point or vertex of the set.

Cost function.--The objective function of a cost minimization program.

Cost range.--The objective function interval of a basic solution variable in which the basis is optimum. A change in the objective function of the variable which is outside that range would force a change in the basis to preserve optimality.

Cost row.--The row of objective function coefficients (the cost of a unit of activity) of a cost minimization linear program matrix.

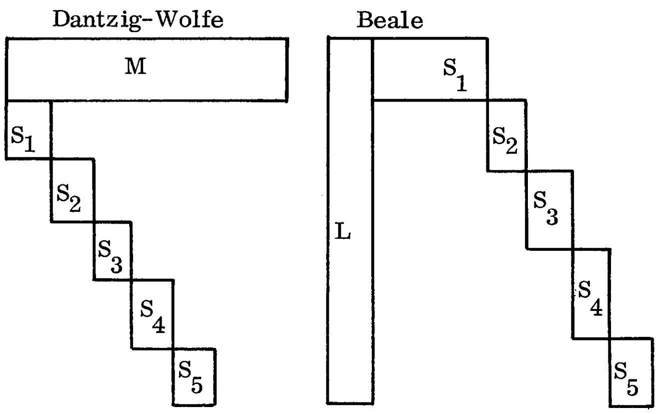
Coupling equations.--When some of the constraints involve only the variables $\{X\}$ and some others involve only the variables $\{Y\}$, the remaining equations involving both $\{X\}$ and $\{Y\}$ are called coupling equations. The master problem would be composed of coupling equations in Dantzig-Wolfe decomposition.

Curtain.--A method to give preferential treatment to a selected group of vectors which are thought to comprise an advanced or optimal basis. The selected group of vectors lies within "curtains." Curtains are usually activated by a control card.

Cycling. --A degenerate situation can arise (but is very rare) in an application of the simplex algorithm. The computations proceed endlessly through a cycle of values with the same level of objective function.

d_j. -- Notation occasionally used for an evaluator or "reduced cost" (q.v.) as contrasted with c_j, which represents the actual cost. When feasible and when using the modified or Kuhn-Tucker simplex array $d_j = \alpha_j^F$ where F is the functional row and α_j^F is the functional row of coefficients in terms of the current basis. The d_j's are also the values of the slack variables of the dual problem. Note that the d_j of a primal logical vector is $\pm$ the π - value of its associated row.

Decomposition. --A means of increasing the capacity and at times the efficiency of computers for certain types of linear programs. Two main schemes of decomposition have been developed:



where S_i stands for the ith subprogram, M stands for the master problem consisting of the equations coupling the subproblems, and L stands for the "linking matrix" of variables linking the subproblems. Thus, linear programs which qualify for decomposition are those in which subprograms can be split off from the main body of constraints or variables, with each subprogram involving only some of the variables and with no variables common to any two subprograms. At each step of the algorithm each subprogram is solved as an individual linear program, relative to a suitably chosen objective function, or RHS. These solutions are averaged or "blended" into a solution for the total program. The overall solution is tested for feasibility and optimality, and if nonfeasible or nonoptimal, a new stage is developed with improved feasibility or lower cost.

Degeneracy. --A special or limiting state or condition, usually produced by a zero value for some parameter or combination of parameters. In an m-constraint linear programming problem, the term is reserved for the case where a basic solution has fewer than m nonzero components. Such degeneracy arises directly from a tied situation in the previous iteration, in which the chosen entering variable has caused the value of two basic variables to reach zero simultaneously, although only one would leave the basis.

Degenerate solution. --In linear programming, a basic solution in which at least one basic variable has zero activity.

Delta. -- The "Kronecker delta" (q.v.) -- the Rth row or column of a unit matrix.

Density of a matrix. --The proportion of nonzero elements in the matrix. In an m x n matrix (a_{ij}), the formula is

$$\text{Density} = \frac{\text{count of nonzero } a_{ij}\text{'s}}{mn}$$

Structural density accounts for the structural matrix only, whereas gross density adds the nonzero elements of the logical vectors and divides by $m(m+n)$.

Departing variable. --The basic variable that becomes nonbasic in the LP iteration; also called "leaving vector", or "vector out". It is the vector that corresponds to the pivot row position in the basis.

Dickson-Frederick pricing criterion. --A procedure for choosing the pivot column in the performance of the simplex method. It eliminates the effects of scaling of the problem data by choosing that column j, $(d_j, \alpha_{ij}, \ldots, \alpha_{mj})$ where d_j is the reduced cost and α_{ij} the corresponding transformed entries, for which d_j is negative and $d_j^2 / \sum_i (\alpha_{ij}^+)^2$ is maximal, where (α_{ij}^+) denotes the "positive part" of α_{ij}.

Digital error. --Rounding, or truncation, error in a digital computer which becomes especially serious with ill-conditioned matrices.

Direct problem. --The primal linear programming problem as originally stated, as contrasted with its dual.

Double iteration. --A stage in a multiple-pricing algorithm when two vectors enter the basis with but a single pass through the matrix.

Double pricing. --An algorithm which chooses two candidates to enter the basis on a single pass through the matrix.

Duality theorems for linear programming. --

1. Main theorem: If both the primal and dual problems have a finite optimum, then the optimum values are equal.
2. Corollary: If either problem has a feasible finite optimum, then so does the other, and the optimum values are equal.
3. Corollary: A feasible but unbounded solution to one problem implies no feasible solution for the other.
4. Corollary: No feasible solution to one problem implies that the other is either unbounded or infeasible.
5. Weak theorem of the alternative: A variable and its complementary slack are not both nonzero.
6. Strong theorem of the alternative: Among all alternate optima, at least one solution exists in which a variable and its complementary slack are not both zero, and the one of the pair that is zero in this solution is zero in all alternate solutions.

Dual problem. --Let the original or direct linear programming problem be that of determining nonnegative values of $x_1, x_2, \ldots, x_n$ so that $z = c_1x_1 + \ldots + c_nx_n$ is minimized subject to

$$\sum_{j=1}^{n} a_j^i x_j \geq b^i \quad (i = 1, 2, \ldots, m)$$

The dual problem is defined to be that of determining values of $w^1, w^2, \ldots, w^m$, so that $z = b^1w^1 + \ldots + b^mw^m$ is maximized subject to

$$\sum_{i=1}^{m} a_j^i w^i \leq c_j \quad (j = 1, 2, \ldots, n)$$

Dual simplex algorithm.--Starting with a dual-feasible solution, the algorithm selects the most infeasible vector to leave the basis and then computes which vector must enter to maintain dual feasibility. As a consequence of this change of basis, other variables may become primal infeasible, but since the objective value changes monotonically toward optimum, the algorithm is finitely convergent. It is exactly the primal algorithm applied to the dual problem.

Dual slack. --The level of the slack variables of the dual problem is given by the primal evaluators (reduced costs, d_j's).

Dual solution. --The optimal solution to the dual problem (which is necessarily obtained in solving the primal problem, but which may not be printed out). Dual solution may be interpreted as the optimum set of marginal values and reduced costs (d_j) for the primal problem.

Dual variables. --The variables of the dual problem. Their coefficients are the rows of the primal problem. The optimum values of the structural dual variables and the column relation slacks are made up, in the language of the direct problem, of the marginal values and the reduced costs, respectively.

Dual variable-type restriction. --A type of restriction-- free, non-negative, nonpositive, fixed -- on each variable (matrix row) of the dual problem. The dual variable-type restrictions correspond one-to-one with the row relations as follows:

$\pi^i \gtrless 0$ (free) $\Longleftrightarrow \sum a_j^i x^i = b^i$ (equal)

$\pi^i \geq 0$ (non-negative) $\Longleftrightarrow \sum a_j^i x^i \leq b^i$ (less than or equal)

$\pi^i \leq 0$ (nonpositive) $\Longleftrightarrow \sum a_j^i x^j \geq b^i$ (greater than or equal)

$\pi^i = S^F$ (fixed) $\Longleftrightarrow \sum a_j^i x^j \gtrless b^i$ (null)

Dynamic programming.--A method for optimizing a set of decisions which must be made sequentially. Characteristically, each decision must be made in the light of the information embodied in a small number of observables called state variables. The incurred cost for each period is a mathematical function of the current state and decision variables, while future states are functions of these variables. The aim of the decision policy is to minimize the total incurred cost, or equivalently the average cost per period. The mathematical treatment of such decision problems involves the theory of functional equations, and usually requires a digital computer for implementation.

Econometrics.--The application of mathematical estimation and inference techniques to economic data, in order to set the levels of economic parameters and to verify or disprove economic theorems.

Element. --(1) A member of a population or set; (2) an individual coefficient, such as a_{11} or a_{27}, in a matrix.

Elementary matrix. --A unit or identity matrix except for one unique column or row. The inverse of a nonsingular elementary matrix is also an elementary matrix. This inverse is often called a "transformation matrix" (q.v.).

Entering variable.--The nonbasic variable chosen to become basic in the iteration. Its associated vector, when expressed in terms of the current basis, is the pivot column.

Epsilon (ϵ).--(1) Notation for a tolerance or appropriately small number; (2) notation for a change in a matrix element, as in certain degeneracy-correcting procedures.

Epsilon perturbation.--A technique for avoiding cycling (in a degenerate problem).

Equality relation.--A constraint relation of either the primal or dual problem which is an equality. The logical (artificial) variable associated with the relation has an element of one in the row (primal) or column (dual). The logical (artificial) variable is fixed at zero level. The complementary variable of a primal logical variable is a free (may take on any positive, zero or negative value) dual variable. Conversely, the complementary variable of a dual logical vector is a free primal vector.

Equation (equality). --A proposition (or relation) that a given mathematical function shall be equal to another function (often a constant). The process of identifying the values of the variables (functional arguments) which make the proposition true is called solving the equation.

Equivalent equations.--Two equations or equation systems in the same unknowns which have the same set of solutions.

Essential zero. --A"tolerance" (q.v.) about zero such that all numbers within that tolerance are set equal to zero.

Eta record.--A physical record consisting of eta vectors; usually packed to eliminate zero elements.

Eta (η)-vector. --The "product form" (q.v.) of the basis inverse consists of a product of elementary column matrices. Columns of these matrices which are not unit vectors are called η -vectors. The successive η -vectors and a notation of their position are all that is required by the computer to maintain the inverse information since the rest of the matrix is an identity matrix. When dealing with sparse matrices, the product form usually contains fewer nonzero elements than the explicit form. Furthermore, since the inverse itself is almost never desired, but only the product of the inverse and a vector, it is more efficient to recompute the required data than to maintain an explicit inverse.

Evaluator.--A number associated with a nonbasic variable at an iterative stage in the simplex method of linear programming. Numerically, the evaluator Δ_k is the change in the objective function which results when the variable x_k is increased from 0 to 1 and the basic variables are shifted in value so as to maintain functional equality. Note that not all constraints will necessarily remain satisfied with x_k at the 1.0 level and no change of basis. If all the evaluators are non-negative, the current basic solution is optimal (yields a true minimum). Also called reduced cost, d_j, simplex evaluator, dual slack, etc.

Expanded tableau.--The linear programming matrix as it appears after the introduction of logical (both slack and artificial) vectors. It is to be compared with the original matrix, which includes the structural vectors only. The expanded matrix has the same length (vertical dimension) as the original matrix, but its width (horizontal dimension) is increased by the number of logical vectors (one per row).

Explicit tableau.--A tableau with a specific numerical value listed for every element (as opposed to a function for computing the values of the tableau, such as the product form of the inverse or the explicit inverse). A tableau is always in terms of a basis; the current basis is understood unless otherwise stated.

Extreme point.--A point of a convex set which does not lie on a line segment joining any other two points of the set; also called a "vertex".

Feasible basis.--A basis such that postmultiplying the inverse of such a basis by the constant column yields a solution which satisfies the constraints. This usually requires that structural and slack variables be non-negative and that artificial variables (other than objective row artificials) be zero.

Feasible solution.--A solution to the constraint equations in which all variables satisfy their sign restrictions (see "feasible basis"). A feasible solution which corresponds to a feasible basis is a basic feasible solution.

Feasibility vector.--A row vector for premultiplying a tableau so that feasibility evaluators (in the composite algorithm) are weighted to choose vectors which will reduce the sum of the absolute values of the infeasibilities when brought into the basis.

File.--A group of stored data on a tape, disk, etc., usually contiguous.

Fixed variable.--(1) A variable in the problem (logical, structural, primal, or dual) fixed at zero level for feasibility; (2) a variable to be bounded away from zero is sometimes "fixed" at its bound in a bounded variable algorithm, so that the transformed variable associated with it is then feasible at zero level, thus permitting arbitrary upper and lower bounds.

Forced entry.--The deliberate introduction of a nonbasic variable into an optimum basis (displacing one of the optimal basis variables), which yields the activity level of the forced variable and the change in total optimum functional value.

Ford-Fulkerson problem.--The "maximum network flow problem" (q.v.).

Formulation of a problem.--The translation of a physical problem (for example, how to schedule a refinery) into a mathematical problem (maximize this function subject to these constraints). It includes the identification of all the variables, the synthesis of the objective function, and the discovery of all pertinent constraints, but not necessarily the assignment of numerical values to the coefficients and other parameters. However, the relationships of variables to constraints is usually established, indicating the presence or absence of a nonzero coefficient for each matrix position.

Fractional programming.--A class of mathematical programming problems in which the objective function is the quotient of linear functions. (Clearly a nonlinear programming problem.)

Free variable.--A primal or dual variable whose feasible values have no restriction as to sign.

Function.--A dependent variable related to independent variables such that for some sets of given values of the independent variables, values of the dependent variables correspond. See also "linear function" and "quadratic function".

Functional.--(1) Generally, a function whose argument is another function. (2) In linear programming, the cost function or profit function or, more generally, the objective function.

Game theory.--The theory of games is a branch of mathematics that aims to analyze various problems of conflict by abstracting common strategic features for study in theoretical "models"--termed "games" because they are patterned on actual games such as bridge and poker. By stressing strategic aspects--that is, aspects controlled by the participants--it goes beyond the classical theory of probability, in which the treatment of games is limited to aspects of pure chance. Zero-sum, two-person games can be solved by linear programming methods.

Gamma (γ).--The updated negative of a change vector for parametric programming on the right-hand side.

Gaussian elimination.--A reduction method for systems of simultaneous linear equations, in which one of the equations is solved for one of the variables in terms of the remaining variables. When these expressions for the solved variable are substituted into the remaining equations, the result is an equivalent system with one less equation and one less variable. A Gaussian elimination step is exactly equivalent to a pivot step. It is a single change of basis and can be expressed functionally as premultiplying by the inverse of an appropriate elementary column matrix. Sufficient repetition of this procedure can yield the numerical solution in case of a nonsingular square system, and a solution of parametric form (a linear function of a subset of the variables) if the number of variables exceeds the number of equations.

Geometric solution.--A graphic method of solving a linear programming problem, by plotting the half-planes determined by the constraints and the lines of constant value for the functional. Its use is restricted to problems with, at most, two structural variables.

Global optimum.--A feasible solution which gives a value to the objective function at least as optimal as any other in the feasible region. It is contrasted with a local optimum, which yields the best objective function value of all points in its subset of the feasible region. In linear programming a local optimum is a global optimum.

Gradient of a function.--A vector at a point, whose direction is the direction of most rapid change of some function f, and whose magnitude is the rate of change of f in that direction.

Graphic solution.--A solution obtained with graphs or other pictorial devices, as contrasted with solutions obtained by the manipulation of numbers.

Hyperplane.--An (n-1)-space in n-space; the set of all points $(x_1 \ldots x_n)$ in n-space whose coordinates satisfy some linear equation $a_1x_1 + \ldots + a_nx_n = b$. For example, a plane is a hyperplane in three-space; a line is a hyperplane in two-space; a point is a hyperplane in one-space.

Identity matrix.--(1) A square matrix with diagonal elements all unity and all other elements zero. The identity matrix (which is frequently denoted by the symbol I) plays the same role in matrix algebra that unity does in the algebra of real numbers--that is, $AI = IA = A$ for any (conforming) square matrix A; also, it is the element with respect to which matrix inverses are defined--for example, $A^{-1}A = AA^{-1} = I$. (2) Colloquial: In linear programming, a square matrix with diagonal elements all plus or minus one, with all other elements zero.

Implicit prices.--Same as marginal values, shadow prices, dual variable levels, etc.--that is, numbers giving the incremental worth of a relaxation of one unit in the right-hand side of a constraint.

Increased profit.--The analog of "reduced cost" (q.v.) in a situation where the objective function is expressed in terms of profit. Numerically, it is the evaluator associated with a nonbasic variable at optimality. It is the amount by which the corresponding unit profit would have to be increased before this variable could profitably be introduced into this basis.

Incremental cost.--(1) Variable cost (as contrasted with fixed costs). It is the increase in total cost due to adding, buying, or making one additional unit. (2) Applies only to constraints originally stated as ($\geq$) inequalities. It is the increase in the optimal (maximal) value of the objective function which would be produced by a decrease of one in the right-hand side. It is the negative of the corresponding "marginal value" (q.v.).

Independent equations.--A set of equations none of which can be expressed as a linear combination of the others. With linear equations, the condition for independence is that the matrix (coefficient columns) shall be nonsingular or, equivalently, have rank equal to the number of equations.

Inequality.--A proposition (or relation) which relates the magnitudes of two mathematical expressions or functions A and B. Inequalities are of four types; A is greater than B ($A>B$); A is less than B ($A<B$); A is greater than or equal to B ($A\geq B$); A is less than or equal to B ($A\leq B$). The first two types are called "strict" and the last two "relaxed" or "weak". The process of identifying a functional argument or range of arguments which makes the proposition true is called solving the inequality, or obtaining a feasible solution to the inequality.

Inequality relation.--A constraint relation of either the primal or dual problem which is an inequality. The associated logical variable (of the same problem) having its one in the row or column of an inequality relation is non-negative if the relation is $\leq$ and non-positive if the relation is $\geq$. (Note that in some codes, with explicit slack vectors in the A-matrix, the negative of the logical unit vector is generated.)

These logical vectors are often called slacks, as contrasted with artificials. A slack level has the interpretation: amount of unused capacity, amount by which a specification is exceeded, etc.

Inequation.--Colloquial for "inequality".

Infeasible basis.--A basis corresponding to an infeasible solution. Postmultiplying the inverse of such a basis by the constant column, one obtains a solution in which at least one variable value is outside its prescribed range.

Initial basis.--The set of column vectors associated with the basic variables for which a starting solution will be obtained in linear programming. It is often an identity matrix or a matrix with plus and minus ones on the diagonal, zeros elsewhere, consisting only of logical vectors.

Initial feasible basis.--The set of column vectors associated with the basic variables of the first feasible solution to a linear programming problem.

Initial solution.--The first trial solution used at the beginning of an iterative-type solution to an optimization problem.

Instability.--See "numerical instability".

Integer programming.--A class of optimization problems in which the values of all of the variables are restricted to be integers. Normally, the optimization problem without this integer restriction is a linear program; additional adjectives indicate variations--for example, integer quadratic programming.

Inverse of a matrix.--An inverse of the square matrix A is another matrix B of the same dimension such that AB = BA = I, where I is the identity matrix. A matrix has at most one inverse. It has exactly one inverse if it is square with a nonzero determinant. The inverse of A may be obtained by extending A by an identity matrix and performing pivot steps on A to reduce it to an identity matrix. If the same operations were applied to the extension, it would end up as the inverse of A. See "product form of inverse".

Inversion.--(1) An operation on a matrix yielding the matrix's inverse. The inversion operation is usually denoted by a superscript -1. (2) In linear programming, the inversion of a specified basis matrix.

Iteration.--A single cycle of operations in a solution algorithm made up of a number of such cycles.

J (H).--Notation for the basis headings list.

Kronecker delta (δ^R).--A row or column vector of the unit matrix where the (+1) element occupies the Rth position in the vector.

Kuhn-Tucker simplex array.--As in the modified simplex array, the objective function is changed to an equation and added to the rows of the matrix. The right-hand side is also brought into the matrix as a variable whose level will be fixed at exactly minus one. This array has the virtue of symmetry and is convenient from a computing standpoint since every row and column is treated alike. However, row and column relations must be made explicitly available; these relations are equivalent to the sign restrictions on the corresponding dual and primal variables.

Lagrange multipliers.--Auxiliary variables λ_1, $\lambda_2 \ldots, \lambda_h$ used in optimizing an objective function $f(x_1, x_2, \ldots, x_n)$ subject to h constraints $\phi_1 = 0, \phi_2 = 0, \ldots, \phi_h = 0$ on the x_i. The procedure is to construct an auxiliary objective function

$$f + \lambda_1\phi_1 + \lambda_2\phi_2 + \cdots + \lambda_h\phi_h ,$$

obtain the first partial derivatives with respect to each x_j and also with respect to each λ_i, and set these partial derivatives all equal to zero. The result will be a set of (n + h) equations in the (n + h) variables $(x_1, \ldots, x_n, \lambda_1, \ldots, \lambda_h)$ which, when solved, will include the optimal coordinates of the x_j subject to the given restrictions. In linear programming, this procedure does not directly yield the solution, basically because the full set of restrictions for an LP problem includes some inequalities (in particular, the common restriction that all variables be non-negative). But by substituting $(\omega_j)^2$ for $\{x_j$ subject to $x_j \geq 0\}$ a new problem in ω^2 can be stated in which all restrictions are equalities. This problem can be solved by Lagrangian methods, and the values of ω_j^2 will be the optimum values of x_j and the values of λ_i will be the optimum solution to the dual problem.

Left-hand side.--The mathematical expression to the left of the equality or inequality sign in an equation or inequality. In linear programming, by convention, the left-hand side of each constraint is the complete linear function, while the right-hand side is the constant term.

Length of a vector.--The square root of the sum of the squares of the vector coefficients; also called the Euclidean norm of the vector.

Lexicographic ordering.--Dictionary ordering; may be used in place of epsilon perturbations to insure that cycling does not occur when a vector choice is degenerate; not ordinarily incorporated in linear programming codes since practical evidence of cycling is very rare.

Limitation.--A constraint or restriction on a variable or set of variables.

Linear combination.--An expression of the form

$$a_1z_1 + a_2z_2 + \ldots + a_nz_n$$

where the a_i are coefficients and the z_i are symbols, such as variables or vectors, which can be multiplied by numbers.

Linear dependence.--The state or condition of a set of n vectors $z_1, z_2, \ldots, z_n$ which are related by an equation of the form

$$a_1z_1 + a_2z_2 + \ldots + a_nz_n = 0$$

Where 0 is a vector consisting of all zero coefficients, and the a_i are a set of coefficients not all zero.

Linear equation.--An equation whose left-hand side and right-hand side are both linear functions of the variables. Such an equation can always be put in the form $f(x, y, z, \ldots) = c$, where f is a linear function and c is a constant.

Linear function.--A function of the form $A_o + A_1 x_1 + A_2x_2 + \ldots A_nx_n$, where A_i's are coefficients not all zero and x_i's are variables. The graphic representation of a linear function is a straight line, plane, or hyperplane.

Linear independence.--The state or condition of a set of n vectors $z_1, z_2, \ldots, z_n$ for which all linear combinations (except for the trivial combination with all coefficients zero) are vectors of nonzero length; that is, the resultant vector has at least one nonzero coefficient. No linear combination of a subset of the vectors will yield a vector of the set which is not in the subset.

Linear inequality.--An inequality whose left-hand side and right-hand side are both linear functions of the variables. By transposition, such an inequality can always be thrown in the form $L(x, y, z, \ldots) R c$, where L is a linear function, c is a constant, and R is the given inequality relation. An inequality can always be converted to an equality by the addition or subtraction of a non-negative "slack" variable.

Linear program.--See "linear programming problem".

Linear programming.--A technique for finding the best solution from among all solutions of a system of linear inequalities. The variables are usually processing or scheduling variables in some physical situation; the inequalities are obtained from the physical constraints on these variables; and the criterion for "best solution" is the value of some given linear function of all the variables. As the term is used today, linear programming includes the formulation of the problem in linear programming terms, algorithms for finding the best solution, and the analysis of the effect of changes in the values of problem parameters. When a solution fails to exist, the system is said to be infeasible or to have no feasible solution. When the best solution is infinite in one or more variables, the system is said to be unbounded.

Linear programming problem.--The mathematical problem of minimizing or maximizing a linear function of n variables, subject to n independent restrictions, such as requirements that each variable be non-negative, and also subject to a finite number of other linear constraints. The latter are either equalities or weak inequalities ($\leq$ or $\geq$); strict inequalities of the form $<$ or $>$ are not admissible. An exact solution or other termination to this problem is furnished by the simplex method or one of its variants.

Linking variables.--In a situation where certain subsets of the variables have nonzero coefficients only in corresponding distinct subsets of rows, the remaining variables having coefficients in more than one subset of rows are called linking variables. These variables comprise the linking matrix in Beale decomposition.

Local optimum.--A feasible solution which gives a value to the objective function at least as low (or high) as any of its immediate neighbors in the feasible region. It is contrasted with a global optimum, which gives the lowest (or highest) value obtainable in the feasible region.

Logical vector.--A unit vector used to extend an LP tableau to provide an "identity" matrix. Unit vectors in $\leq$ rows are positive slacks. Unit vectors in $\geq$ rows are negative slacks (either a unit positive vector restricted to nonpositive values, or a unit negative vector restricted to non-negative values). Unit vectors in = rows are artificial zero-level variables. Unit vectors in functional or other non-constraint rows are free (artificial) variables.

Lower bound (of a set of numbers).--Let A = (a) be a set of real numbers. Then the real number c is a lower bound for a, provided $c \leq a$ for every a in A.

Lower bound (of a variable value).--To be feasible, the value of the variable must be greater than or equal to its lower bound. In linear programming, variables are normally lower-bounded at zero--that is, $x_i \geq 0$.

Machine loading.--Given a set of jobs (i), i = 1, 2, ..., n; where job i consists of a set of operations (i, j), j = 1, 2, ..., n_i. Also, given a set of machines (k), k = 1, 2, ..., N; as well as the information that operation (i, j) can be performed on a subset Ω (i, j) of these N machines at a rate of r_{ij}. The machine loading for this set of jobs is the selection of a particular machine (k) from the relevant subset (Ω) for each operation (i, j). The machine-loading problem is to select that particular machine loading which satisfies all constraints on the system at least cost. The "transportation problem" (q.v.) is a subset of this problem. In transportation problems, the $r_{ij} = 1$ for all i, j; and the sum of demands is usually constrained to equal the sum of supplies.

Marginal.--The economists' term for the derivatives, or rate of change of a function with respect to quantity. "Incremental" and "variable" are often used in an exactly synonymous sense. Thus, the composite terms: "marginal cost" (of production), "marginal revenue" (from sales), "marginal value" (of a capacity, of sales, of supplies, etc.). The coefficients of a linear programming model are themselves all marginal figures; for example: the cost coefficient of an activity is the marginal cost of performing the activity; the coefficient in a material-balance row is the marginal consumption or production of the material.

Marginal cost.--The rate of change of cost as a function of quantity. If C(x) denotes the total cost of x units bought or produced, the marginal cost is another function c(x) equal to C(x+1)-C(x).

Marginal revenue.--The rate of change of income as a function of quantity. If R(x) denotes the income from the sale of x units, the marginal revenue is another function r(x) equal to R(x+1)-R(x).

Marginal value.--The dual solution to a linear programming problem can be interpreted, in the economic sense, as a set of marginal values since a dual-solution value is the change in the objective function per unit increase in the corresponding right-hand-side constant, assuming that a change of basis is not required to maintain feasibility. The economic interpretation is complicated by alternate optimum solutions (corresponding to different bases) which may yield different values for the dual variables. Thus, there may be two or more marginal values for the same constraint; such multiple values must be interpreted in terms of the operating procedure indicated by the corresponding basis; that is, the basis determines how the relaxation of the constraint would be utilized. Unfortunately, a change of basis (to an alternate optimum) may be required to change an RHS constant in a given direction even infinitesimally, and aside from parametric programming on the right-hand side, there is no convenient way to assure that the basis at hand is one which will permit the desired change. (The right-hand-side ranging algorithm will only indicate the amount and direction of RHS change possible before needing an alternate basis.)

Mathematical programming.--Techniques of finding an optimum value of a function of many variables when these variables are subject to restrictions in the form of equations or inequalities. The term is usually restricted to problems so complex that they require a digital computer for their solution. Special types of programming are linear programming, quadratic programming, and nonlinear programming, all discussed separately in this glossary.

Matrix.--A rectangular array for the compact display of the mn numbers.

$$a_{ij} \ (i=1 \ldots m; j=1 \ldots n)$$

The typical element a_{ij} is placed in the ith row and the jth column. Another notation for an element is a^i_j, which is useful in that the position of the index can indicate an element of a row or column and can distinguish rows or columns--for example, b^i is an entry in a column; π_i is an entry in a row; etc.

Matrix element.--One of the mn numbers, symbols, or other objects which have been arrayed in the form of a matrix.

Matrix generator.--A computer code whose input is other stored information and whose output is a set of mn numbers (a_{ij}), so indexed, or whose output is only the nonzero element values located or positioned by column identification and row identification.

Maximize.--Find values of the arguments which give the largest possible value to a function.

Maximum network flow problem.--Given a capacitated network with one node distinguished as the source, and another node distinguished as the sink. To find a maximal flow from source to sink, subject to the following conditions: (1) the flow in each arc must be in some preassigned direction and cannot exceed the arc's capacity; (2) the flow into a node may not exceed its capacity; and (3) the sum of the flows into any intermediate node (not source or sink) must equal all the flows out of it.

Minimize.--Find values of the arguments which will give the smallest possible value to a function.

Mixed-integer programming.--Integer programs in which some, but not all, of the variables are restricted to integral values.

Model.--A synthetic system which is patterned after an actual system but which is easier to manipulate. The synthetic system is designed to resemble the actual system in ways considered important, so that its manipulation can provide useful information.

Modified simplex array.--A computing array for the simplex procedure in which the objective function plus an artificial variable is set equal to a constant (usually zero) and appended to the rows of the matrix. This artificial variable is unrestricted as to sign, and in a cost minimization problem, its value is to be maximized. This array also permits simultaneous evaluation of a number of functionals while optimizing one of them, or a prescribed linear combination of them. When using this array for computation, the sign of the objective function coefficients is frequently changed before data input so that the sign of the value of the corresponding free (artificial) variable as printed out will correspond to the true functional.

Network flow problem.--See "maximum network flow problem".

Node.--One of the defining points of a network.

Nonbasic variable.--A variable in a linear programming iterative stage whose value is fixed at zero, or is bound, and whose column vector therefore does not appear in the current basis.

Nondegenerate solution.--A basic solution in which all of the basic variables are nonzero valued.

Nonlinear equation.--An equation at least one of whose terms is a nonlinear function of the variables.

Nonlinear function.--A function defined as something other than a sum of terms consisting of a constant times a single variable plus a final constant. For example, $ax + bx^2 + c$ (where x is a variable and a, b and c are constants) is a nonlinear function.

Nonlinear programming.--An inclusive term covering all types of constrained optimization problems except those where the objective function and the constraints are all linear. Special types of nonlinear programming for which some theory has been developed are convex programming, concave programming, and quadratic programming.

Non-negativity restriction.--A restriction that a value or a set of numbers must be positive or zero, but in no instance negative.

Nonsingular matrix.--A square matrix whose determinant is not zero, and thus whose rank (number of linearly independent columns) is equal to its dimension (number of rows). Any nonsingular matrix has an inverse.

Null constraint.--A row of the primal problem which is not a constraint--that is, symbolically $A \lesseqgtr B$. The objective function, cost-change rows for parametric programming on the cost row, and alternate objective functions are all null constraints. In addition, it may be desirable to include some other functions to determine the activity level of the free (artificial) variable of that row.

Numerical instability.--A characteristic of computations or computational methods in which small deviations in initial input data tend to produce large deviations in the results. Ill-conditioned matrices are numerically unstable.

Objective function.--That function of the independent variables whose maximum or minimum is sought in an optimization problem.

Objective function coefficient range.--See "cost range".

Objective function element.--One of the coefficients in the objective function row of a linear programming matrix.

Objective value.--The value of the objective function at a particular solution.

Operations research.--"The attack of modern science on problems of likelihood (accepting mischance) which arise in the management and control of men and machines, material and money in their natural environment. Its special technique is to invent a strategy of control by measuring, comparing, and predicting probable behavior through a scientific model of a situation." (Stafford Beer)

Opportunity cost.--The money or other value sacrificed by choosing a nonoptimal course of action.

Opportunity loss.--Same as "opportunity cost".

Optimize.--Maximize or minimize, as the case may be.

Optimum.--A set of values of the variables which optimizes (gives the most favorable value to) the objective function.

Original simplex array.--An array for computation by the simplex method in which both the objective function and RHS are kept separate and handled differently from the coefficients in the matrix, as distinguished from such variants as the modified, or Kuhn-Tucker, arrays.

Original simplex method (computing form).--Using an original simplex array, the functional row c_j is stored separately from the matrix and is never updated. The evaluators are calculated as

$$d_j = c_j - \sum_i C^i \alpha_j^i$$

where C^i are the cost coefficients of the vectors in the basis, and α_j^i is the matrix tableau expressed in terms of the current basis. This is in contrast to standard simplex method.

Packing.--Condensed storage of sparse vectors or matrices, accomplished by storing only nonzero elements and their identifications or indexes. In certain cases, it is more efficient to store zeros but not infinity; this would be termed infinity packing as distinguished from (zero) packing. It is also occasionally possible to pack out certain coefficients equal to one.

Parameter.--A secondary variable in an application. For example, the analytic geometry description of a line, y = ax + b, can be replaced by the parametric representation

$$\left\{ \begin{array}{l} y = at + b \\ x = t \end{array} \right\}$$

where t is regarded as a parameter. The constants, a and b, and the dependent variables, x and y, are not considered as parameters.

Parametric programming.--A method for investigating the effect on an optimal linear programming solution of a sequence of proportionate changes in the elements of a single row or column of the matrix. Most commonly, the method is applied to either the objective function row or the right-hand-side column. If it is applied to the latter, for example, the original right-hand vector B is replaced by a new vector B + Qt, where Q is a constant vector and t is a parameter. A parametric programming program may print out any or all of the following: the values of t at which basis changes occur, the new basic set at each change, the optimal levels of the basic primal and dual variables at each change, and the rate of change of each basic variable (with respect to t) between the current t and its value at the next basis change.

Partition.--To separate a linear program into related subprograms.

PHI.--The notation used for the parameter in the objective function parameterization or the critical price ratio in the dual algorithm and its variants.

Piecewise linear approximation.--The division of the domain of definition of a function into subregions, and the replacement of the function by some close-fitting linear function in each subregion.

PI values.--Notation often used for the simplex multipliers.

Pivot column.--The column of the matrix containing the pivot element. In a linear programming iteration, it is the column associated with the entering variable (nonbasic variable picked to become basic).

Pivot element.--The pivot-row element of the pivot column.

Pivot rejection.--Rejection of the entering variable turned up by the pricing criterion, because the corresponding pivot element would be too close to zero. In such a case, the next best nonbasic variable is examined, and so on.

Pivot row.--The row of the matrix containing the pivot element. In a linear programming iteration, it is the row associated with the departing variable (basic variable picked to become nonbasic).

Pivot step.--A step consisting of a single transformation of the matrix in a pivotal method of reduction of a set of linear equations.

Pivotal method.--The most common of the many methods used in the solution of sets of linear equations. A particular equation (pivot row) is solved for a particular (pivot) variable. The pivot coefficient plays a dominant role in the elimination or condensation to zero of all other coefficients for the chosen vector (by substitution in the other equations).

Postmultiply.--To multiply a matrix A by some conforming matrix B--that is, by a matrix with as many rows as the given matrix has columns, A x B.

Postoptimum analysis.--Development of the right-hand-side ranges, cost or profit ranges, trade-offs between pairs of basic variables, and/or parametric programming.

Precision.--The number of words, or storage positions, used to denote a number in a computer. Single-precision arithmetic is the use of one word per number; double-precision, two words per number, and so on. For variable word-length computers, precision is the number of digits used to denote a number. The higher the precision, the greater the number of decimal places that can be carried.

Premultiply.--To multiply a matrix B by some conforming matrix A--that is, by a matrix with as many columns as the given matrix has rows, A x B.

Pricing.--The choice of a nonbasic variable to enter the basis using an appropriate pricing criterion.

Pricing criterion.--The criterion for choosing an entering variable. The most common (though perhaps not the best) criterion in the primal algorithm is to choose that entering variable whose evaluator is most negative.

Pricing vector.--(1) In modified simplex this refers to the updated cost row. (2) In revised simplex this refers to the "simplex evaluators" or "PI values".

Primal simplex algorithm.--Refers to a type of linear programming algorithm in which a basic feasible solution to the expanded primal (direct) problem is maintained at each step of the iterative process, with a feasible solution to the dual problem appearing only at optimality. This is in contrast to a "dual simplex" method (q.v.).

Primal solution.--(1) The basis of the primal variables (both logical and structural) at optimum; (2) the activity levels of these variables.

Primal variable-type restriction.--A type restriction, free, non-negative, nonpositive, or fixed, on each variable (whose coefficients are a matrix column) of the primal problem. The primal variable-type restrictions correspond one-to-one with the column relations as follows:

$$x_j \gtrless 0 \text{ (free)} \quad \Longleftrightarrow \quad \sum_i \pi_i a^i_j = c_j \text{ (equal)}$$

$$x_j \le 0 \text{ (non-negative)} \quad \Longleftrightarrow \quad \sum_i \pi_i a^i_j \geq c_j \text{ (greater than or equal)}$$

$$x_j \ge 0 \text{ (nonpositive)} \quad \Longleftrightarrow \quad \sum_i \pi_i a^i_j \le c_j \text{ (less than or equal)}$$

$$x_j = F^L \text{ (fixed)} \quad \Longleftrightarrow \quad \sum_i \pi_i a^i_j \substack{\le \\ >} c_j \text{ (null)}$$

Problem formulation.--See "formulation of a problem".

Product form of inverse (computing form).--A computationally efficient (for sparse matrices) way of updating only the required portion of the inverse; often used in "revised simplex" (q.v.) codes due to its calculation and input-output characteristics. The inverse π of the basis at stage t is carried as the product of elementary transformation matrices, one for each pivot step, rather than being computed and carried explicitly.

$$\pi = E_t \, E_{t-1} \cdots E_1$$

where E_t is an identity matrix except for one column (see "eta-vector"); otherwise, the computing form is the same as the revised simplex. The subscripts on E are an index on pivots. The product form is used recursively to compute an updated row or column of the tableau as follows:

$$\pi^R_j = (\ldots(\delta^R_j E_t) E_{t-1})\ldots) E_1);$$

$$\alpha^i_S = (E_t \ldots (E_2(E_1 a^i_S)\ldots);$$

$$d_j = \pi^0_i a^i_j$$

π^R_j is the R^{th} row of the basis inverse.
E_t is the t^{th} "transformation matrix" (q.v.).
a^i_J is the J^{th} column of the LP matrix.
α^i_S is the S^{th} column of the LP matrix expressed in terms of the current basis.

Profit range.--The interval $(c - \Delta_1,\ c + \Delta_2)$ where c is a profit coefficient in the objective function, and the two endpoints are points where a change of basis must first occur to maintain optimality. See also "cost range".

Programming.--Arranging or furnishing a program of or for (Webster's Dictionary). In computer usage, it usually refers to computer programming, which is a stylized technique for making a computer carry out a particular calculation. To avoid semantic difficulty when speaking in context of mathematical programming, computer programming (or programs) is termed "coding" or "codes". Compound terms such as linear programming, quadratic programming, and dynamic programming are subsumed under the general heading of mathematical programming -- the optimization of a given function subject to given restrictions on the variables.

Quadratic function.--A function which contains second-degree terms in the variables, e.g., $a + bx + cx^2$.

Quadratic programming.--Maximization, subject to linear constraints, of an objective function which is a quadratic function. This is one of the few convex programming problems, aside from linear programming, which have solution algorithms that terminate in a finite number of steps.

Rank of a matrix.-- The maximum number of linearly independent rows of the matrix. Equivalently, the maximum number of linearly independent columns, or the size of the largest square submatrix with an inverse.

Raw-stock value.--Marginal value, in a situation where the right-hand-side element is a capacity for a stock-balance equation.

Reduced cost.--The evaluator associated with a non-basic variable at optimality. It is the amount by which the corresponding unit cost would have to be reduced before this variable could profitably be introduced into the basis.

Redundant equations.--A set of equations, one of which may be expressed as a linear combination of the others. Such an equation can be omitted from the system without affecting the solution. With the inclusion of a full set of logical vectors in an LP matrix, there are no redundant equations in the strict sense of the term; however, very much the same situation applies when one relation or constraint dominates another; for example, the constraint equation $X_1 = 0$ dominates the constraint $X_1 \geq 0$ so that the latter constraint could be removed without affecting the solution.

Regularization.--A technique for augmenting a matrix with (1) a row that constrains the sum of absolute values of the structural variables to be $\leq R$, where R is large enough not to affect the solution if not otherwise unbounded; and (2) pseudo-artificial vectors (unit vectors with a large penalty cost M) for each non-null row relation, where M is large enough so that the pseudo artificials will not be used unless the problem is otherwise infeasible. A regularized problem, then, always has an optimum feasible solution.

Reinversion.--A means of avoiding the buildup of round-off errors in an explicit tableau or inverse, or in a product-form inverse. However, in product form the most important effect of reinversion is to reduce the number of eta-vectors required to express the inverse, and hence the number of nonzeros in the eta records. Reinversion may be called for when digital error is suspected, at a presumed optimum, every K iterations (usually $30 \leq K \leq 75$), or in product form by clock or nonzero control to minimize the total time required for solution. Unlike the normal iterating agenda where the pivot elements are taken in the order prescribed by vector in-and-out choices, the reinversion algorithm can, and usually does, make a special selection of pivots to maintain digital accuracy, and in product form to also preserve sparseness.

Report generator.--A computer code whose input is data, operating instructions and format information. Linear programming report-generator data may include solution activity levels, input coefficients, etc. Linear programming report-generator instructions may include vector multiplications, additions, etc. The output of a report generator may include instructions to a printing unit as well as output information.

Requirement.--The right-hand side of a constraint of the form

$$x_1 + x_2 + \dots + x_k = b, \text{ or } \leq b$$

where x_i are interpreted as inflows to some sink; also used for "constraint" (q.v.) and "restriction" (q.v.).

Resource.--The right-hand side of a constraint of the form

$$x_1 + x_2 + \dots + x_k \leq b$$

where x_i are interpreted as outflows from some source.

Restriction.--An equation or inequality limiting the feasible range of variation of a variable. Also used for constraint (q.v.).

Revised simplex method (computing form).--A variant of the simplex method, especially suitable for LP problems in which the number of variables is much larger than the number of equations. The method employs (1) an implicit inverse, "product form of the inverse" (q.v.), or (2) an explicit inverse of the current basis ^i_h (often without zero packing) to compute an updated row or column of the tableau as required.

$$d_J = (\delta^R_i \pi^i_h)\, a^h_J, \quad \alpha^i_S = \pi^i_h\, a^h_S, \quad \pi_T = E_T\, \pi_{T-1}$$

where:

R is the functional row.

E_T is the transformation matrix of iteration T.

π_T is the basis inverse at iteration T.

π^i_h is the current basis inverse.

The simplex multipliers comprise the profit row of the inverse; they are used to test for optimality and to select a new entering variable, if needed.

RHS.--Notation for the right-hand side.

Right-hand side.--The mathematical expression to the right of the equality or inequality sign in an equation or inequality. In linear programming, by convention, the right-hand side of each constraint is merely the constant term, with the complete linear function as the left-hand side. Same as B-vector, column of constraints, etc.

Right-hand-side element.--An element of the column vector comprising the right-hand sides of the constraints.

Right-hand-side range.--The interval $(b-\Delta_1,\ b+\Delta_2)$ where b is the original value of the right-hand-side element and the two endpoints are the first points where a change of basis must occur to maintain feasibility. Within this interval, the optimal objective value is a linear function of the right-hand-side element, with rate of change equal to the marginal value of its row.

Rounding error.--The error in the final result of a calculation which is ascribed to the rounding used for input factors and intermediate results. In a computer, rounding error can be reduced almost to any desired extent, but only at the expense of speed, memory capacity, and program complexity.

Round-off error.--Same as "rounding error".

Row.--All the elements of a single horizontal line of a matrix, taken in the given order. A (1 x n) submatrix of an (m x n) matrix.

Row error.--The numerical value of

$$a_{i1}x_1 + a_{i2}x_2 + \dots + a_{in}x_n - b_i$$

where $(x_1, x_2, \dots, x_n)$ are the levels of the variables in the current solution; $(a_{i1} \dots a_{in})$ is the left-hand side of row i; and b_i is its right-hand side. With no rounding error, the above expression would have a value of zero.

Row ID.--The same as "row identification". The Row ID list (or the Row ID's) refers to the set of all row identifications to be used in a particular problem.

Row identification.--Generally, a name or mark to distinguish a particular row of a matrix; specifically, a six-character identification of a row in the SHARE standard input format.

Row index.--A numerical equivalent of the row identification, assigned by the computer for the purpose of saving space.

Row operations.--One of the following operations on a matrix: (1) interchanging any two of its rows; (2) multiplying a row by a nonzero constant; (3) adding a multiple of a row to some other row. Operation (1) is equivalent to premultiplying by an identity matrix with two columns permuted. Operations (2) and (3) combined are equivalent to premultiplying by an elementary column matrix. A succession of row operations will invert a nonsingular matrix.

Row relation.--A relation that is equal, less than or equal, greater than or equal, or null for each constraint (matrix row) of the primal problem. The row relations correspond one-to-one with the complementary dual variable-type restrictions as follows:

$$\sum a_j^i x_j = b^i \text{ (equal)} \Leftrightarrow \pi^i \gtrless 0 \text{ (free)}$$

$$\sum a_j^i x_j \leq b^i \text{ (less than or equal)} \Leftrightarrow \pi^i \geq 0 \text{ (non-negative)}$$

$$\sum a_j^i x_j \geq b^i \text{ (greater than or equal)} \Leftrightarrow \pi^i \leq 0 \text{ (nonpositive)}$$

$$\sum a_j^i x_j \lesseqgtr b^i \text{ (null)} \Leftrightarrow \pi^i = S^F \text{ (fixed)}$$

where π^i is a dual variable

a_j^i is the coefficient matrix

b^i is the column of RHS coefficients

x_j is a primal variable

S^F is a fixed scale factor.

Row vector.--A matrix consisting of a single row. A given row of a matrix. The elements of the row are interpreted as the components of the vector.

Scaling of a matrix.--Multiplication of each row and each column of the matrix by different factors chosen to improve the computational properties (condition) of the matrix. By judicious scaling, the precision and efficiency of linear programming computations can be increased greatly, without changing in any material way the definition of the problem. A frequently used scaling algorithm is to divide each row by the largest absolute element in it, and then divide each resulting column by the largest absolute element in it. This ensures that the largest absolute value in the matrix is 1.0 and that each column and row has at least one 1.0 in it.

Scheduling.--(1) The placement of a given set of objects (A, B, C, D, ...) on the time axis, as opposed to sequencing, which is merely ranking them in temporal order. Sometimes the sequencing result is assumed before the scheduling problem is given. (2) Scheduling of a continuous process manufacturing facility, such as an oil refinery, usually means the allocation of streams to units and the setting of processing conditions, since the flows are assumed to be constant during the scheduling period.

Sensitivity. The responsiveness of a solution to changes in the numerical values of the coefficients.

Separable programming.--A class of nonlinear programming problems in which each function appearing may be expressed as the sum of separate functions of single variables. For example, the function

$f(x_1, \ldots, x_n)$ is separable if

$$f(x_1, \ldots, x_n) = \sum_{i=1}^{n} f_i(x_i),$$

for appropriate functions f_i.

When all the functions have been separated, it is possible to make a piecewise linear approximation to each one. For convex constraints, this is all that is required, since the LP algorithm will always choose the two nearest points surrounding the desired value, giving accuracy dependent only on the fineness of the approximation. For concave constraints, however, the ordinary LP algorithms will choose from the two most widely separated points, ignoring those in between and giving generally a most unsatisfactory accuracy. However, the separable programming (LP) algorithm assures that the points chosen are contiguous, and thus accurate. Of course, only a local optimum solution is assured.

Set.--A collection or family of objects of some kind. These objects are called "elements," or "members" of the set.

Shadow price.--Same as "marginal value" (q.v.); also used incorrectly for "reduced cost" (q.v.).

SHARE.--A user's group for IBM large-scale scientific computers (comprising in 1963 the 704, 709, 7090, 7094, 7094-II, 7040, 7044 equipments).

SHARE standard input format.--A format for the input card, specifying a single element of the matrix. In this format, columns 1-6 are blank, reserved for control information; columns 7-12 identify the matrix column in which the element appears; columns 13-18 give the row; column 19 gives the sign of the element; and columns 20-30 give its numerical value, with decimal point in column 24.

Sigma Σ.--Notation for the change row in parametric programming on the objective row.

Simplex.--A convex polyhedron in a Euclidean space of dimensions equal to or greater than the number of linearly independent vertices. Simplexes in the plane are points and line segments; simplexes in three-space are points, line segments, and triangles; and so on.

Simplex evaluators.--See "evaluator".

Simplex method.--A computational routine for obtaining the optimal solution to a linear programming problem. It is an iterative elimination procedure at each stage yielding a basic solution, and it rests primarily on the following two principles:

1. Elementary row operations on the constraint matrix leave the set of feasible solutions unchanged.
2. The number of nonzero values in an optimal solution is never more than the number of constraint equations. The following steps comprise one iterative stage:

a. A test of whether the current solution is optimal and/or feasible.

b. If not both optimal and feasible, a choice of an entering variable and a departing variable.

c. A pivot step so as to read off the new solution; then back to a.

An inversion algorithm is sometimes provided to express the problem in terms of a given initial basis. The problem may be formulated in any of three arrays (original, modified, or Kuhn-Tucker) and may be solved by a number of computing forms (original, standard, revised explicit inverse, revised product form of inverse), each of which can employ a variety of choice criteria for the entering and departing variables--that is, primal, composite, Dickson-Frederick, dual, suboptimization, symmetric, etc. Any simplex method is contrasted with approximate techniques, such as logarithmic gradient, relaxation, etc., which do not yield a basic solution.

Simplex method, original.--See "original simplex method (computing form)."

Simplex method, revised.--See "revised simplex method (computing form)."

Simplex multipliers. A set of numbers (Π_1, Π_2, ..., Πm), one for each row of a linear programming problem. Their values are given by the vector equation:
Original array

$$(\Pi_1, \ldots, \Pi_m) = -(c_{i_1} \ldots c_{i_m}) B^{-1}$$

modified or Kuhn-Tucker array

$$\pi_1^R = \delta_1^R \pi$$

where B is the current basis, C_{i_R} is the objective function element of the R^{th} basic variable, and $i_1 \ldots i_m$ are the subscripts of the basic variables.

π_1^R is the R^{th} (profit) row of the inverse.

δ_1^R is a unit row vector with 1 in position R and zeros elsewhere.

π is the inverse of the basis (including the cost row).

The main application of these numbers is in computing evaluators for the nonbasic variables.

Singular matrix.--A square matrix whose rank is less than its dimension. A singular matrix has no inverse.

Sink.--That single node in a capacitated network at which all flow is assumed to terminate. It can be recognized, on a directed graph, as that node at which all flows are directed inward.

Slack.--The difference between the left and right members of an inequality. If $u \leq v$, the slack s is defined as $u + s = v$ or $s = v - u$; such slacks are restricted to be non-negative. If $u \geq v$, the slack may be defined as $u + s = v$ or $s = v - u$ in which case it is restricted to be nonpositive, or it can be defined as $u - s = v$ or $s = u - v$ (note the -1 coefficient of s in $u - s = v$), in which case it is restricted to be non-negative.

Slack level.--The current value of a slack variable.

Slack variable.--An auxiliary variable introduced to convert an inequality constraint to an equation.

Slack vector.--The column vector associated with a slack variable. It is a unit vector with +1 or -1 in the row in which the slack appears, and zeros elsewhere.

Solution.--A set of values of the variables which satisfy the given constraints. Solutions may be classified as basic or nonbasic, feasible or infeasible, optimal or nonoptimal. These qualifiers are all defined under their appropriate alphabetical headings.

Solution level.--The set of values taken by the variables in a solution.

Source.--That single node in a capacitated network from which all flow is assumed to originate. It can be recognized, on a directed graph, as that node from which all flows are directed outward.

Sparse vector or matrix.--A vector or matrix whose elements are mostly zeros.

Standard simplex method (computing form).--Using the modified simplex array (where the objective function has been incorporated in the matrix as a null constraint), the entire explicit tableau is kept up-to-date at each iteration. This method is particularly efficient for computing ranges, or when the structural matrix is dense. The only computation required at a pivot step is

$$\alpha_T = E_T \alpha_{T-1}$$

where E_T is an elementary matrix (q.v.). Since the complete tableau is available at all times, more complicated pricing criteria (such as Dickson-Frederick or maximum change in objective) are feasible.

Starting basis.--The set of column vectors associated with the basic variables of a starting solution in linear programming.

Starting solution.--The first trial solution used at the beginning of an iterative-type solution to an optimization problem. The means for providing this initial solution is sometimes an algorithm in its own right.

Steppingstone method.--A name for a special form of the simplex method used for solving the transportation problem. Given a feasible solution, the algorithm tells how to improve this solution by introducing a new source-destination allocation and reallocating among existing allocations so that the constraints remain satisfied. Those existing allocations whose values are modified in this way are called steppingstones. If no such reallocation can improve the solution, the current solution is optimal.

Stipulations.--The right-hand sides and row relations.

Stochastic programming.--A generalization of linear programming in which any of the unit costs, the coefficients in the constraint equations, and the right-hand sides are random variables subject to known distributions. The aim of such programming is to choose levels for the variables which will minimize the expected (rather than the actual) cost.

Structural vector.--A column of a linear programming matrix corresponding to a physical variable in the original problem.

Submatrix summary.--The partial sum of left-hand-side terms evaluated at the current solution for $p(\leq n)$ variables and $q(\leq m)$ constraints. It takes the form of a column vector with q elements.

Suboptimal.--(1) Not yet optimal. (2) Optimal over a subregion of the feasible region.

Suboptimization.--When using a multiple-pricing algorithm, suboptimization enters that variable first into the basis which gives by itself the greatest improvement in objective; the next variable entered gives the greatest additional improvement, and so on. This technique tends to prevent the first entering variable from being removed by the second, etc.

Substitution rate.--The change in the basic variables (both logical and structural) required to maintain equality as a nonbasic variable is increased from 0 to 1.0 in value. The feasibility restrictions may be violated if the new variable were actually set at the 1.0 level; the extent of the substitution is the largest (absolute) value the new variable can take on without causing an infeasibility.

Sum of infeasibilities.--The (usually negative) sum of the absolute amounts by which the primal variables are outside their type restrictions. When the sum of infeasibilities reaches zero, the corresponding solution becomes feasible. The "Phase I" of certain linear programming codes was simply a derived linear programming problem -- that of maximizing the negative sum of infeasibilities to zero.

Symmetric parametric programming.--The simultaneous parameterization of the right-hand side and the objective function. This is useful in economic studies when both costs and requirements change a linear function of a single parameter -- for example, time. Another use is as a sort of primal-dual algorithm to go from a pseudo RHS and objective function (for which the starting basis is optimal and feasible) to true values of the RHS and objective function. In some problems, this approach requires far fewer iterations than a more conventional algorithm.

Tableau.--The current matrix, with auxiliary rows and/or columns, as it appears at an iterative stage in the standard simplex method computing form of solution.

Technology matrix.--Loosely, the coefficient matrix, or (a_{ij}), as distinguished from the cost coefficients (c_j) and the right-hand sides (b_j).

Theta (θ).--Notation often used for the new value of the entering variable in the updated basic set (in the primal algorithm this is equal to the critical right-hand-side ratio), or for the parameter in right-hand-side parameterization.

Tolerance.--The half-width of a band of numbers centered at zero, all of which are taken as equivalent to zero for decision purposes. Some form of tolerance is required in digital computer applications because of rounding error. Linear programming codes generally have several different zero tolerances for different purposes.

Transformation matrix.--An elementary matrix representing a single change of basis. It is the inverse of a matrix formed from an identity matrix (which is the present basis in terms of itself) by replacing one unit vector (representing the departing variable) with another vector (representing the entering vector). The entering variable is expressed in terms of the present (old) basis.

Transformation of a matrix.--A change in the appearance of a matrix which leaves the solutions of the corresponding set of linear equations unchanged; a change of basis.

Transportation problem.--A linear programming problem of considerable historical and practical interest, for which very efficient solution algorithms are available. In this problem, there are m origins (with origin i possessing a_i items) and n destinations (with destination j requiring b_j items), and with sum of the a_i equal to the sum of the b_j. We are given the mn costs associated with shipping a single item from any origin to any destination, and are asked to empty the origins and fill the destinations in such a way that the total cost is minimized. The problem can be stated as a linear programming problem whose variables are the mn individual allocations, with the summed cost as the objective function and with $m+n-1$ independent constraints, and so solved. However, much more efficient special-purpose algorithms are available; these are called transportation algorithms or codes, in which the special structure of this problem is exploited. A special case of the "machine loading" problem (q. v.).

Traveling-salesman problem.--A special type of assignment problem. A traveling salesman wishes to start in one city, visit each of n other cities, and return to the start in such a way that the total distance traveled is minimized. This can be regarded as an assignment problem in which each city is assigned a successor city, but with the added restriction that the journey from city one back to city one be continuous. The traveling-salesman problem is much more difficult in solution than the corresponding unrestricted-assignment problem, even though there are fewer feasible solutions to consider. A number of solution algorithms have been brought forward, most of them based on integer linear programming or on dynamic programming techniques. However, none as yet are regarded as generally satisfactory: the efficient methods are only approximate, and the exact methods require too much computer time.

Unbounded set of numbers.--A set A = (a) of real numbers which has no lower bound and/or no upper bound. That is, if A has no lower bound, for any real number x, it is possible to find a number (a) in A such that $a < x$. If A has no upper bound, for any real number x, it is possible to find a number (a) in A such that $a > x$.

Unbounded solution.--An optimal solution with some infinite or unbounded values, and therefore with an infinite optimum for the objective function.

Unique.--One and only one (in mathematics).

Unique solution.--The optimal solution, in a problem which has one and only one optimal solution.

Unit negative vector ("UN").--A unit (positive) vector except that the sign of the 1.0 is minus.

Unit vector.--The term is used in two senses, one more special than the other. (1) A vector of unit length with one coefficient (+1) and the other coefficients all zero. (2) A vector of unit length. In the context of linear programming, meaning (2) is never used.

Upper bound (of a set of numbers).--Let A = (a) be a set of real numbers. Then the real number c is an upper bound for A, provided that $c \geq (a)$ for every (a) in A.

Update.--To change a matrix, vector, etc., from stage n into a form suitable for stage $n+1$, usually by routine operations on the former.

Variable.--A symbol which can stand for any member of a given set, as distinguished from a constant, which is a symbol standing for only one such member. An argument of a function whose value is to be established.

Variable cost.--Marginal cost--that is, includes no fixed charges.

Vector.--An ordered set or array of quantities, usually n real numbers $(x_1, x_2, \ldots, x_n)$. A vector arrayed horizontally is called a row vector; one arrayed vertically is called a column vector. For example, the rows of a matrix are row vectors, the columns of a matrix are column vectors.

Vector in.--Entering variable's associated vector.

Vector out.--Departing variable's associated vector.

Vertex.--A point of a convex set which does not lie on a line segment joining any other two points of the set; also called extreme point.

BIBLIOGRAPHY

The following books are listed in order of difficulty, starting with the most elementary and ending with most advanced.

Meisels, K., A Primer of Linear Programming, New York University Press, New York, 1962. An elementary general introduction. The first half is at about the same level of mathematical difficulty as this general information manual; the second half extends the mathematical methods slightly. The book contains a number of additional applications.

Metzger, Robert W., Elementary Mathematical Programming, John Wiley and Sons, New York, 1958. An elementary general introduction. The exposition begins with the transportation problem (which is related to linear programming but is in some ways simpler), continues with the simplex method, and concludes with a number of additional applications and techniques. Mathematical difficulty is little greater than that of this manual.

Vazsonyi, Andrew, Scientific Programming in Business and Industry, John Wiley and Sons, New York, 1958. A longer treatment, covering other topics besides linear programming and offering many applications and examples. The mathematical level is slightly higher than that of this manual.

Ferguson, Robert O., and Lauren F. Sargent, Linear Programming, McGraw-Hill Book Co., New York, 1958. This longer treatment, containing many fairly extensive applications, starts with the transportation problem and presents good discussions of the methods and applications of linear programming.

Vajda, S., Readings in Linear Programming, John Wiley and Sons, New York, 1958. A number of short chapters introduce standard techniques, plus about two dozen applications of many different sorts. Several techniques besides simple linear programming are discussed: the transportation problem, quadratic programming, game theory. The mathematical level is somewhat above that of this manual.

Garvin, W. W., Introduction to Linear Programming, McGraw-Hill Book Co., New York, 1960. This college-level text contains additional techniques and many applications. For the reader with two to three years of college mathematics, it is perhaps one of the most readable presentations of linear programming.

Gass, S. I., Linear Programming: Methods and Applications, McGraw-Hill Book Co., New York, 1958. This graduate-level text places somewhat more emphasis on mathematical and computational techniques than on applications, although there are examples.

Hadley, G., Linear Programming, Addison Wesley, Reading, Massachusetts, 1962. This graduate level text offers a thorough treatment for the reader with adequate mathematical preparation: theory, computational techniques, applications, many exercises.

Charnes, A., and W. W. Cooper, Management Models and Industrial Application of Linear Programming, 2 volumes, John Wiley and Sons, New York, 1960. This graduate-level text by two of the pioneers in the field offers complete and rigorous coverage of linear programming and many related subjects.

INDEX

Activity 7, 8, 9, 11, 13, 15, 31
Algebraic methods 4, 9, 37
Allocation 2, 6, 14, 31
Artificial variable 40

Basic feasible solution 10, 11, 38
Basic solution 10, 38, 39
Basic variable 10, 11, 15, 38
Basis 10, 15, 22, 37
Breakeven price--See marginal value

Change of basis 17, 21, 38
Coefficient 7, 11, 22, 39
Column 7, 21, 24
Constraint 1, 3, 7, 8, 13, 15, 24, 27, 37
Cost range--See profit range

Equation 9

Feasible solution 4, 10, 38
Feasibility solution 40
Feed blending example 7
Formulation of problem 1, 7

Graphical methods 4, 5, 9, 37

Improving the basis 12, 38, 39
Incremental cost--See marginal value
Inequality 3, 9, 16, 37
Input to computer 1, 29, 34
Investment policy example 9

Level of activity 7, 11, 13
Limitation--See constraint
Linear 3, 13

Marginal value 16, 17, 18, 19, 31, 34
Maximization 6, 7, 8, 13, 15
Minimization 7, 8, 13, 15

Nonbasic variable 10, 11, 38, 39
Nonnegative 6, 7, 8, 38

Objective function 1, 7, 11, 13, 15, 18, 21, 22, 27, 39
Optimum policy 1, 2, 8, 10, 15
Output from computer 30, 31

Paper trimming example 14
Parametric programming 17, 21, 34
Production capacity allocation example 2, 14
Profit range 18

Ranging 18, 19, 33, 34
Rate of substitution 19
Raw stock cost--See marginal value
Reduced cost 19, 31
Reduced profit--See reduced cost
Refinery scheduling example 13, 23
Restriction--See constraint
Right-hand side 7, 16, 17, 19, 21, 34
Row 7, 21, 24

Sensitivity 20, 33
Shadow price--See marginal value
Simplex method 2, 5, 12, 37
Simplex multiplier--See marginal value
Slack variable 9, 16, 22, 37
Substitution rate 19

Variable 24
Vertex 6, 9, 37, 38, 39

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]

E20-8171

IBM®

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601
[USA Only]

IBM World Trade Corporation
821 United Nations Plaza, New York, New York 10017
[International]