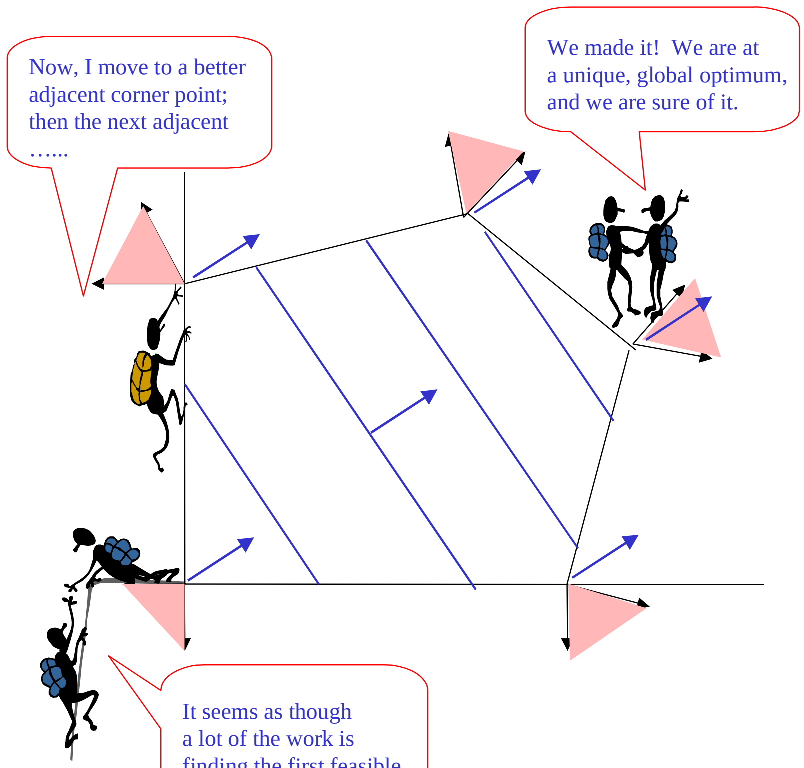
Linear Programming
Key Terms, Concepts, & Methods for the User

Notes for Chemical Engineering 4G03


Now, I move to a better adjacent corner point; then the next adjacent …...
We made it! We are at a unique, global optimum, and we are sure of it.
It seems as though a lot of the work is finding the first feasible corner point.

Thomas Marlin
McMaster University
Hamilton, Ontario, Canada

Forward

This material has been prepared for the student who wishes to learn the basic concepts about linear programming. The material will prepare the student to use linear programming in engineering practice. It should also provide a basis for further study into the mathematics, algorithms and numerical implementation of linear programming.

"The Purpose of Mathematical Programming is Insight, not Numbers."

Arthur Geoffrion (1976)

Linear Programming:
Key Terms, Concepts, & Methods for the User

Table of Contents

Linear Programming
Key Terms, Concepts & Methods for the User

1.0 Linear Programming

We start our studies of optimization methods with linear programming. Basically, we select linear programming because it

- is used widely in engineering practice
- enables us to practice problem formulation and results analysis, including inequality constraints and variable bounds
- gives insight to the power of optimization (versus brute force simulation of many alternatives)
- builds a foundation for other major categories of optimization algorithms

This first section explains why linear programming is a useful method and a good introduction to optimization.

1.1 The Meaning of Optimization

Fourth-year students already have considerable experience with mathematical modelling for simulation, so what is new? In simulation, the results are defined by the user-selected values of the variables and parameters. Thus, we often say that the system of equations used for simulation has zero degrees of freedom; for example, a simulation model with 20 (linearly independent) equations has 20 variables. In optimization, the model has more variables than linearly independent equations; therefore, for a properly formulated optimization problem the user-selected values of variables do not define the results. In optimization, the objective function is used to guide the selection of values for the degrees of freedom, i.e., the "extra" variables.

Example 1.1 Gasoline Blending: A simple process example of simulation and optimization is given in a blending problem. Simulation of the blending problem is depicted in Figure 1.1a, where the flow rates of all streams are defined by the user and the total flow rate, physical properties of the product stream and profit are determined using the model and the user-selected values. Optimization is depicted in Figure 1.1b, where the objective is profit achieved when the product satisfies quality specifications. Profit can be maximized by adjusting the five input flow rates, which have different costs, within user-defined limits.

The optimization result could be estimated by a grid search approach that evaluated the behavior of the blend process for many combinations of component flow rates. If we selected a rather imprecise grid of ten values of flow per component, the grid would have the 10^5 cases to evaluate; if each case required 0.1 second, the rough search would require over 150 minutes. Clearly, a better approach is required. The linear programming method that we will learn in this chapter can optimize the blending problem to high precision with a computing time of less than one second.

For further details on problem definition, please see the lecture notes on “Formulating the Optimization Problem”.

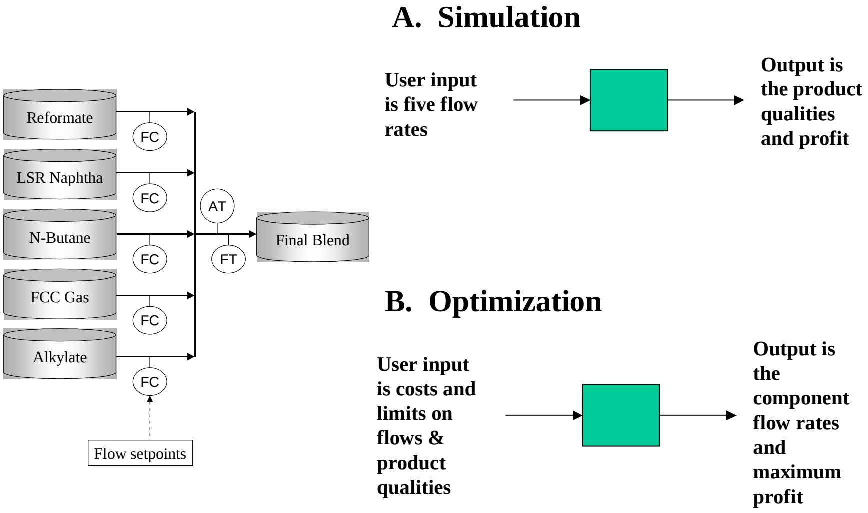

A. Simulation
User input is five flow rates
Output is the product qualities and profit
Reformate
LSR Naphtha
N-Butane
FCC Gas
Alkylate
FC
FC
FC
FC
FC
AT
FT
Final Blend
Flow setpoints
B. Optimization
User input is costs and limits on flows & product qualities
Output is the component flow rates and maximum profit

Figure 1.1 Gasoline blending example.

1.2 The Importance of Linear Programming

Since linear programming (LP) technology can solve large problems reliably, it was the first method widely used for optimization using digital computation. It remains one of the most important – likely ***the most important*** – optimization method. Linear programming is used in a wide range of applications, such as design, manufacturing, personnel planning, investment management, statistics, public health, national public policy, and many more.

A linear programming (LP) problem involves many variables and equations. Current software can solve 100s of thousands to millions of equations and variables in a reasonable time. How can we solve such large mathematical problems? The key feature is in the name – ***linear*** programming. After several years of engineering study, you have seen that most models involve non-linear expressions, and therefore, you might be dubious about the value of linear model. Please keep an open mind, because we will see many useful applications and learn model formulations that enable us to solve realistic problems with linear programming.

Optimization in general, and linear programming in many instances, is a natural way to formulate and solve engineering problems. In the past, problems requiring fast solution could not be solved using optimization, so that ad hoc solution methods were developed that gave rapid, but sub-optimal, solutions. An example is automatic control, whose development predated digital computation and linear programming. However, linear programming can solve some problems very fast and is replacing older methods in selected real-time applications.

Example 1.2 Optimizing transportation costs: This example will demonstrate the importance of having a systematic mathematical method for optimization. We will design a transportation system between the plants, warehouses and customers in Figure 1.2. The manufacturing costs in the plants are the same, as are the storage costs in the warehouses; therefore, our goal in this problem is to satisfy the customer demands at minimum transportation costs. If faced with this challenge (and not knowing optimization) you would likely apply a heuristic to find a solution.

The word "heuristic" in this context means a "a set of rules or procedures based on experience and qualitative analysis that is not rigorous". Using reliable heuristics that provide nearly optimal solutions is not a bad approach, if possible. However, we seldom have a bound on the gap between the best (optimum) and the result achieved using a heuristic; thus, we will learn optimization in this chapter. But first, let's apply two reasonable heuristics to this problem.

Heuristic 1, Sequential decision making: We will first decide the best policy for the warehouse-to-customer flows; then, we will decide the best plant-to-warehouse flows. In the first step, we rank the costs of satisfying the customer demands from the warehouses, from which we select the flows that give the lowest cost alternates.

From W2 to C1	50,000 units
From W2 to C2	100,000 units
From W2 to C3	50,000 units

In the second step, we determine the flows that give the minimum cost for the plant to warehouse, which are given below.

From P2 to W2	60,000 units	(Note that the maximum production in P2 is 60,000)
From P1 to W2	140,000 units	

This result satisfies all strict customer requirements and does not exceed the capacity of plant 2; we will call this a feasible solution. The total cost is **$1,200,000**. Is this good; is this the best? Without optimization, we do not know.

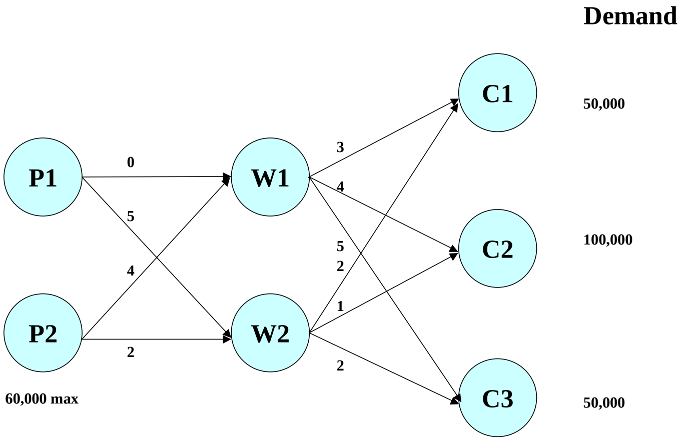


Figure 1.2 Transportation problem from Example 1.2. Costs ($/unit) are noted for each path. (This example and figure are from Geffrion and Van Roy (1979); see this reference for further examples and interesting discussion.)

Heuristic 2, Decision making with some look ahead: The first heuristic did not consider the plant-warehouse costs when making the first decisions. In this heuristic, we will first find the plant-warehouse-customer paths that give the lowest costs and decide on the best warehouse-customer flows. The lowest cost paths are given in the flowing

For C1	P1-W1-C1
For C2	P2-W2-C2
For C3	P2-W2-C3

Observing these paths gives the following flows from the warehouses.

From W1 to C1	50,000 units
From W2 to C2	100,000 units
From W2 to C3	50,000 units

Second, we select the lowest cost for plant-warehouse flows to satisfy the above decisions, which are given in the following.

From P1 to W1	50,000 units	
From P2 to W2	60,000 units	(Note that the maximum production in P2 is 60,000)
From P1 to W2	90,000 units	

This result satisfies all strict customer requirements and does not exceed the capacity of plant 2; it is also a feasible solution. The total cost is **$920,000, less than the first solution**. Is this good; is this the best? Without optimization, we do not know.

Optimization: This problem can be solved to determine the optimum using methods introduced in this chapter. The computing time will be less than one second, and the minimum cost is **$740,000!** That is a big improvement achieved with fast computation, so let's keep reading to learn optimization.

The previous example has given some insight into the complexity of an optimization problem. For the complex problems, even this small example, why do heuristics often fail?

- Complete enumeration of alternatives is usually impossible. For example, selecting 15 variables from 30 candidates has over 155,000 possibilities.
- Sequential decision making will not find correct solutions because of interactions among decisions.
- Capacity limits (for example, the maximum production in plant 2) are very difficult to include in heuristics
- Problem data, especially costs and limits, change frequently. Therefore, the "same" problem with different parameters has to be solved often.

In this chapter, we will learn linear programming to quickly and efficiently solve many optimization problems.

1.3 Learning Goals

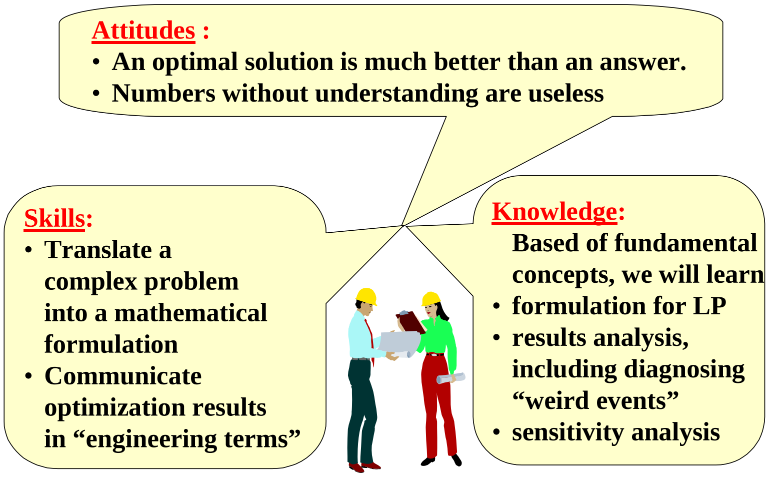
Optimization via linear programming is a vast topic, which for mastery requires sophisticated mathematical analysis, advanced numerical methods, computer coding, mathematical modelling and results analysis. Well, that is too much for 70 pages and too much for an introduction! However, we want to be sure to learn what most engineers need to know for engineering practice.

One common method for explaining learning goals is to address three key categories; attitudes, skills and knowledge (Rugarcia, et al, 2000). The key learning goals for linear programming are given in Figure 1.3.

After you have completed this chapter, you will be able to

- explain the basic concepts of linear programming along with advantages and limitations
- sketch the feasible region in two dimensions and demonstrate the simplex algorithm procedure
- formulate appropriate linear programming models of technical and economic applications
- analyze the results, including sensitivity and diagnosis of unusual events
- explain an optimization study from formulation to results analysis, including preparing a formal report.

This chapter does not fully prepare you for developing a computer program to solve linear programming or to extend the technology through research. However, it will provide a good basis for engineering practice and if your interest is piqued, further studies.



Attitudes :

- An optimal solution is much better than an answer.
- Numbers without understanding are useless

Skills:

- Translate a complex problem into a mathematical formulation
- Communicate optimization results in "engineering terms"

Knowledge:

Based of fundamental concepts, we will learn

- formulation for LP
- results analysis, including diagnosing "weird events"
- sensitivity analysis

Figure 1.3 Learning Goals for engineering optimization.

2.0 Key Modelling Assumptions and Limitations

We begin with some key assumptions that limit the types of models used in linear programming. We must understand and abide by these limitations. When first encountering these model limitations, the engineering student might conclude that few realistic problems could the represented. However, many model formulations have been developed for use with LPs, as we will see in Section 8.

2.1 Linearity

This is the key feature that enables the impressive performance of LP methods. It also places severe restrictions on the model; ***both*** the constraints and the objective function must be linear. Therefore, the engineer must understand linearity. Linearity consists of the following two properties.

Proportionality: The contribution of a variable to the objective function or constraint function is proportional to the value of the variable.

Additivity: The value of an objective function or constraint function is the sum of the contributions of each variable. Note that proportionality does not exclude cross-product terms, so that the additivity property is required.

2.2 Divisibility

We assume that any variable can be divided into any small value, in other words, variables are continuous. Other variables we encounter often can assume only specific values, such as 0.0 or 1.0; these we call discrete variables. Some examples of continuous and discrete variables are given in the following.

Continuous

- Temperature
- Pressure
- Flow of liquid
- Mole fraction
- Weight of granular material in a bin
- Enthalpy

Discrete

- Number of automobiles manufactured per shift
- Number of trays in a distillation column
- Pipe diameter, because only specific sizes are manufactured
- One of several mutually exclusive investment decisions

We might argue that some of these continuous variables are really discrete, because of a finite number of molecules in a system or of quantum effects, but the divisibility assumption is excellent for engineering problems.

We have two choices when discrete variables are present.

- We can assume that the all variables are continuous and round off the answer to the closest integer for the variables that are not continuous.
- We can use a model with integer variables, which requires an entirely different solution method, integer programming (Williams, 1999).

When we consider LP methods, we must have all continuous variables or we must be able to model approximately using continuous variables and round off the answer to the nearest discrete value after solution. This round-off method is not always appropriate, for example, when selecting one of mutually exclusive investments.

2.3 Certainty

Often, we assume certainty without stating it, which is not a good practice. Here, we will expressly acknowledge that we are assuming that all information used in the LP is known exactly. We will see that we can evaluate the effects of changes in some of the data easily using sensitivity (or post-optimal) analysis. Therefore, we typically report the optimization results for the base case, or best estimate, of uncertain parameters and also provide how much the solution changes for small changes in the uncertain parameters.

Using only the best estimate of the parameter is not appropriate for all problems. For example, we might want to make a decision that is profitable (or safe or meets product requirements) for all values of uncertain parameters within their range. If the uncertainly is large and has a strong influence on the results, we will have to use linear programming solution methods that explicitly consider uncertainty, such as stochastic linear programming (Sen and Hingle, 1999).

2.4 Formulating a Linear Program

We formulate a linear programming problem by tailoring the general optimization problem. We begin with the general optimization problem.

$$\begin{aligned} &\min_{x} z = f(x) \\ &s.t. \\ &\mathbf{h(x)} = \mathbf{0} \\ &\mathbf{g(x)} \geq \mathbf{0} \\ &\mathbf{x_{min}} \leq \mathbf{x} \leq \mathbf{x_{max}} \end{aligned} \tag{2.1}$$

with x a vector of variables, h(x) equality constraints (equations) and g(x) inequality constraints. The variables can be bounded between upper and lower limits. For a linear program, the optimization problem is the following.

$$\begin{aligned} &\min_{x} z = c^T x \\ &s.t. \\ &A_h x = b_h \\ &A_g x \geq b_g \\ &\mathbf{x_{min}} \leq \mathbf{x} \leq \mathbf{x_{max}} \end{aligned} \tag{2.2}$$

with $\mathbf{c^T x} = c_1 x_1 + c_2 x_2 + \mathbf{....} = \sum_{i=1}^{n} c_i x_i$

A = matrices of constant *left-hand side* coefficients multiplied by the variables **x**
$\mathbf{b_j}$ = vectors of *right-hand side* constants
c = vector of cost coefficients

In general, the equations and inequalities define a region in which the optimum can exist, which we call the feasible region. The point (or points) where z is minimized within the region is the optimum.

The reader should recognize that a user of LP software defines the problem by inputting the coefficients **c, A, b, $\mathbf{x}_{\mathbf{min}}$** and **$\mathbf{x}_{\mathbf{max}}$.** The user does not perform the calculations explained in Sections 4 and 5. However, informed users of linear programming *must understand* the solution method so that they can properly select the LP method, formulate an appropriate linearized model, and interpret the numerical results from a computer program.

3.0 Linear Programming Properties and Advantages

The properties introduced in the previous section enable us to greatly simplify our mathematical models and to use very efficient solution methods. The solution method (algorithm) for LP uses these following properties that result from the assumptions in Section 2.

3.1 Convexity

Convex set: The feasible region for a linear program has an important property that greatly simplifies the problem solution, convexity. A region is convex if all points on a straight line connecting any two points within the region are also in the region. A sketch of a general convex region is given in Figure 3.1. Importantly, we will see that a problem stated as an LP, abiding by the standard formulation in equation (2.2), involves a convex set

Convex objective: The objective function is linear, which is also convex. A convex function satisfies the following expression.

$$f[\gamma x_1 + (1-\gamma)x_2] \leq \gamma\ f(x_1) + (1-\gamma) f(x_2) \tag{3.1}$$

with γ = a constant having a value between 0 and 1.

A convex objective minimized over a convex region is termed a *convex programming problem*. An important theoretical result in optimization is that a local optimum in a convex programming problem is a global optimum. A point at which all other surrounding (local) points are inferior or worse is a local optimum. For convex programming problems, this is automatically the optimum within the entire region of x defined in the problem!

Therefore, in linear programming a local optimum is a global optimum!

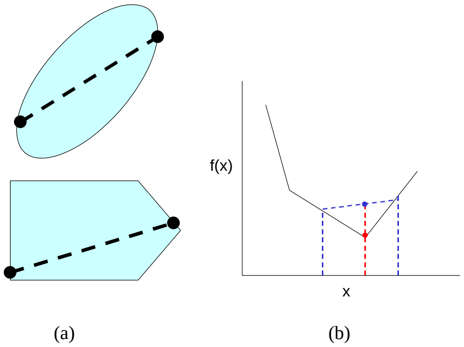


Figure 3.1 (a) Convex sets and convex function

We must be careful to recall that the optimum might *not be unique*. The value for the objective at the local optimum cannot be improved in the feasible region. However, many values for variables **x** in the feasible region might have the same value of the objective function.

3.2 Activity of Inequalities

Here, we introduce a note on terminology regarding status of inequality constraints. Each inequality is described by one of the following terms for any *feasible point*.

Inactive: (Non-binding) When the values of the variables result in the left-hand side not being equal to the constant on the right hand side and conforming to the appropriate (≥ or ≤) inequality constraint.

Active: (Binding) When the values of the variables result in the left-hand side being equal to the constant on the right hand side.

Some references use alternative terminology for the same concept, binding for active and non-binding for inactive. Naturally, if the inequality constraint is violated, the point is infeasible. A few examples are given in the following.

Inequality	Variables values	Status of inequality constraint
$2.5x_1 + 1.5\ x_2 \le 10$	$x_1 = 2\ ,\ x_2 = 3 \Rightarrow 9.5 < 10$	Inactive (feasible)
$2.5x_1 + 1.5\ x_2 \le 10$	$x_1 = 2\ ,\ x_2 = 3.333 \Rightarrow 10 = 10$	Active (feasible)
$2.5x_1 + 1.5\ x_2 \le 10$	$x_1 = 3\ ,\ x_2 = 2 \Rightarrow 10.5 > 10$	Infeasible (violated)

3.3 Location of Optimum

The efficiency and reliability of LP solution techniques depend upon a strong statement about the location of the optimum in an LP problem. The statement involves corner point locations in the feasible region.

> **Corner Point**: A point is a corner point (**p**) if every line segment in the set (feasible region) containing **p** has **p** as an endpoint. When explaining linear programming, various references use the following terms, all having the same meaning: **corner point, extreme point, and vertex**.

From observing Figure 3.2, we can conclude the following (Hillier and Lieberman, 2001).

> Consider a linear programming problem with feasible solutions and a bounded region. The optimal value of the objective function is located at a corner-point solution! Thus, **if the problem has one optimal solution, it must be a corner point (vertex); if it has multiple optimal solutions, at least two must be located at corner points (vertices)**.

In addition to the global optimum the solution method that we will learn determines the following:

- **Bounded or unbounded** – We need to determine whether the optimum values of one or more variables are unbounded (giving an objective value of ± ∞). If this occurs, the problem formulation is in error, because no real system has variables with infinite range.

- **Feasible or infeasible -** We need to determine whether a feasible region exists or does not exist (no solution). This could be due to a formulation error or a very stringent performance requirement. For example, we might require a reactor product yield of over 60%, while the maximum achievable is less than 60% because of side reactions.

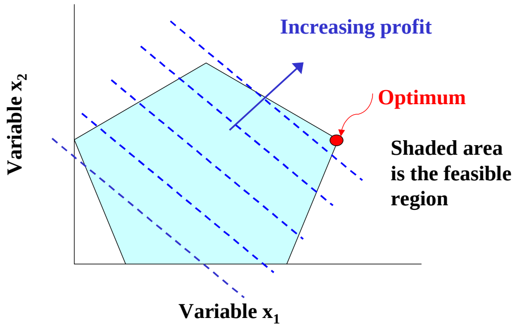


Figure 3.2. A typical LP problem showing the unique optimum at a corner point.

- **Unique or alternative -** The unique optimum value of the objective occurs at a corner point, as indicated above. However, an "edge" intersecting the corner point could have the same value of the objective.
- **Sensitivity to coefficient changes -** We need to determine sensitivity information regarding the effects of changes in some of the parameters.

These are *very complete results*, not generally available in optimization. Therefore, for computational efficiency and excellent results analysis,

We will seek to formulate an optimization problem as an LP, when the method provides adequate accuracy for the problem being solved.

Now, we will learn how we can use these properties to define the principles for locating the optimum of a linear program. Then, we will develop the algorithm in Section 5 that uses these principles to locate the optimum with computational efficiency.

4.0 Principles for Solving a Linear Programming Problem

We have learned that the optimum of a linear program occurs at a corner point of the feasible region. Our task here is to develop equations that define corner points and to establish criteria for identifying the best corner point. These principles will result is set of equations. Therefore, we begin by reviewing the solution to a set on linear equations.

4.1 Solving linear equations

We begin by reviewing the solution of a square set of linear equations with "m" equations and variables. If the equations are linearly independent, a solution exists.

The set of linear equations can be represented in the following matrix equation.

$$\mathbf{A\,x} = \mathbf{b} \tag{4.1}$$

With **A** = coefficient matrix (mxm)
b = "right hand side" vector of constants (mx1)
x = vector of values of the variables x_i (mx1)

The solution of the set of equations can be represented as the following, which requires evaluating the inverse of the coefficient matrix, assuming that **A** is full rank (non-singular), so that the inverse exists.

$$\mathbf{x = A^{-1}\,b} \tag{4.2}$$

The solution can be determined without solving explicitly for the inverse by applying the Gauss-Jordan method, which applies elementary row operations to reduce the coefficient matrix to the identity matrix.

Since similar approaches are used in linear programming, we will briefly review the Gauss-Jordan method for solving a set of linear algebraic equation. This method employs elementary row operations to rearrange the coefficient matrix to the identity matrix. When the same row operations are applied to the right-hand side coefficients, the solution can be obtained by observation, because elementary operations do not change the solution of the equations. (For example, see Edgar et al 2001, Appendix A or Chapra and Canale, 1998).

Example 4.1 Let's solve the following set of linear equations.

$$\begin{aligned} 2x_1 + 1x_2 + 3x_3 &= 3 \\ 5x_1 + 4x_2 + 3x_3 &= 2 \\ \frac{1}{2}x_1 + \frac{2}{3}x_2 + \frac{3}{2}x_3 &= 4 \end{aligned} \tag{4.3}$$

These equations can be restated in matrix form as the following.

$$Ax = b \tag{4.4}$$

with $A = \begin{bmatrix} 2 & 1 & 3 \\ 5 & 4 & 3 \\ 1/2 & 2/3 & 3/2 \end{bmatrix}$ x = $\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$ b = $\begin{bmatrix} 3 \\ 2 \\ 4 \end{bmatrix}$

To prepare for the Gauss-Jordan method, we align the left-hand side "A" coefficient matrix with the rows of the right-hand side "b" values.

$$\begin{array}{ccc:c} 2 & 1 & 3 & 3 \\ 5 & 4 & 3 & 2 \\ 1/2 & 3/2 & 3/2 & 4 \end{array} \tag{4.5}$$

Now, we proceed to perform elementary row operations on the "A" matrix to yield an identity matrix and also perform the same operations on the right-hand side values. We select the (1,1) position for a pivot. First we divide the first row by 2 to achieve a value of 1.0 in the (1,1) element. Then, we multiply the modified first row by -5 and add the values to the second row to achieve a 0.0 in the (2,1) position. After these first two steps, we have the following matrix.

$$\begin{array}{ccc:c} 1 & 1/2 & 3/2 & 3/2 \\ 0 & 3/2 & -9/2 & -13/2 \\ 1/2 & 2/3 & 3/2 & 4 \end{array} \qquad (4.6)$$

At the completion of the procedure, equation (4.5) is transformed in the following result.

$$\begin{array}{ccc:c} 1 & 0 & 0 & -23/6 \\ 0 & 1 & 0 & 7/2 \\ 0 & 0 & 1 & 43/18 \end{array} \qquad (4.5)$$

Clearly, the result has been obtained, with $x_1 = -23/6$, $x_2 = 7/2$ and $x_3 = 43/18$.

The elementary row operations in the Gauss-Jordan do not affect the solution of the linear equations, but they result in coefficients that yield the solution by observation.

These calculations can be quite tedious and time consuming; however, they can be easily performed by a computer program. In addition, excellent interactive tools are available to help students "learn by doing". The educational tools require the student to make key decisions and provide a few numbers, while the computer program performs the extensive calculations and displays updated results. See Appendix B for the location of these tools on the WWW.

4.2 The LP formulation

A general LP problem could be formulated as given in the following.

$$\begin{aligned} &\min_{\mathbf{x}} \mathbf{z} = \mathbf{c}^{\mathbf{T}}\mathbf{x} \\ &\textbf{s.t.} \\ &\mathbf{A_h x = b_h} \\ &\mathbf{A}_1\mathbf{x} \ge \mathbf{b}_1 \\ &\mathbf{A}_2\mathbf{x} \le \mathbf{b}_2 \\ &\mathbf{x} \ge 0 \end{aligned} \qquad (4.6)$$

with **A** = coefficient matrices
c = original cost vector
b = vector of constants on the right hand side (rhs) of equation or inequality constraints
x = vector of problem variables
z = scalar objective function

It is probably worth repeating that this is the formulation used when inputting a problem to a software package. The user does not usually perform the procedures described in this and the next section, such as adding slack and artificial variables, arranging in canonical (standard) form, and performing the tableau calculations. However, the engineer needs to know the concepts behind the method to formulate models and interpret results.

The variables in equation (4.6) are limited to be greater than or equal to zero; we often term this "non-negative". We will generalize the approach later to include variables that can be negative as well as positive and have lower and upper bounds.

The optimization problem is stated as a minimization to be consistent with most books on optimization. We can solve a maximization problem by noting that maximizing (z) is equivalent to minimizing (-z). Also, by convention the values of the right hand sides of the equations and inequalities are positive ($b_i \geq 0$). We can always achieve a positive right hand side by multiplying the equation or inequality by (-1). Recall that multiplying by (-1) changes the sense of an inequality, e.g., less than (<) to greater than (>).

We want to convert to a formulation involving only equalities, since a corner point is defined by equalities (the original equations and a subset of active equalities). Converting to equalities can be achieved by adding a variable to any *inequality*. This variable has the value of the difference between the left-hand side value (depending on the variable values) and the right-hand side value (a constant). These variables are termed **slack variables,** which are limited to be non-negative (≥0). By adding one slack variable to each inequality (but not to equalities), inequalities are converted to equalities. When this addition has been completed, all relationships among variables are equalities. Examples are given in the following.

Original expression	**Slack added to form equality** **(Note that $\mathbf{x_s} \geq 0$)**
$5\mathbf{x}_1 + 7\mathbf{x}_2 - 2.3\mathbf{x}_3 \leq 37$	$5\mathbf{x}_1 + 7\mathbf{x}_2 - 2.3\mathbf{x}_3 + x_{s1} = 37$
$5\mathbf{x}_1 + 7\mathbf{x}_2 - 2.3\mathbf{x}_3 \geq 37$	$5x_1 + 7x_2 - 2.3x_3 - x_{s2} = 37$
$5\mathbf{x}_1 + 7\mathbf{x}_2 - 2.3\mathbf{x}_3 = 37$	No modification needed.

Unfortunately, the terminology is not consistent among references. We will use the term "slack" for a variable added to convert an inequality to an equality, regardless of the sign of its coefficient, plus or minus. Some references use "slack" when the coefficient is +1 and either "surplus" or "excess" variable when the coefficient is –1.

After we have added slack variables where needed, we have the following

<u>Standard Form of the Linear Programming Problem</u>

$$\begin{aligned} &\min_{x} z = c^T x \\ &s.t. \\ &A\ x = b \\ &x \geq 0 \end{aligned} \qquad (4.7)$$

The **A** matrix in the equation above includes coefficients from all equalities and inequalities in the original formulation, equation (4.6) and the coefficients of the slack variables added to the problem to convert inequalities to equalities. The variable **x** vector includes original problem and slack variables.

We will assume that the problem in standard form has more variables than equations.

If the problem had the same number of variables and (independent) equations, a single solution exists, and no degrees of freedom would exist for optimization. If it had fewer variables than equations, no solution would exist. When more variables exist, degrees of freedom exist for improving the objective function while satisfying the equations. In most engineering optimization problems, this assumption is valid. The most common reason for initially violating the assumption is an overly restrictive definition of system performance. In this situation, we usually convert the problem to one with additional variables using the "goal programming" approach covered in Section 8.6.

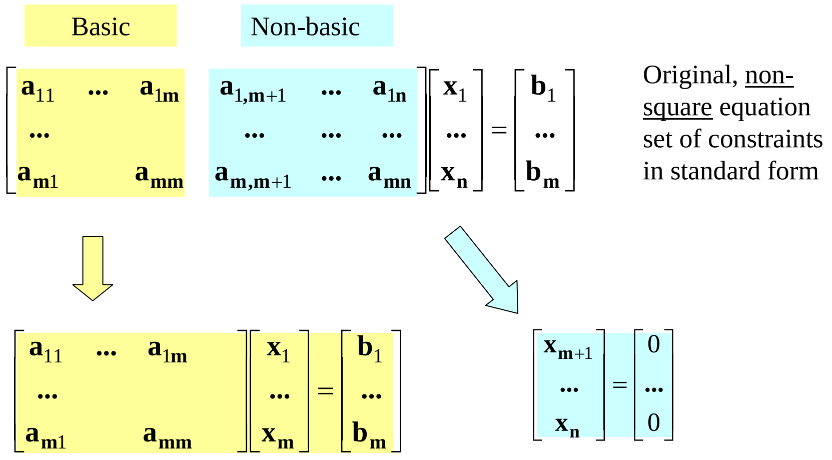
We will use the letter "n" to denote the number of variables and "m" to denote the number of equations, with n ≥ m.

We can think of the problem as "m" variables that are determined by the equations and "n-m" variables that are the degrees of freedom for optimization. Therefore, the solution approach involves finding the correct set of variables for solving the equations and finding the correct values for the remaining variables that minimize the objective function. We will use the following terminology when referring to this selection.

Basic variables: "m" variables determined by the equations
Non-basic variables: "n-m" variables that are set to values that minimize the objective

Recall that the variables include slack variables, so that changing the selection of basic variables has the effect of changing which inequalities are active (slacks = 0) or not active (slacks > 0).

The resulting problem is shown schematically in Figure 4.1. Clearly, we can make many different selections of basic and non-basic variables. How do we determine the best, or optimum selection? One way would be to evaluate all combinations of "m" variables selected from "n". We could solve the equations for each combination, and if the solution were feasible ($\mathbf{x} \geq \mathbf{0}$), we could evaluate the objective function. After all feasible objective values were evaluated, we could select the feasible corner point with the minimum objective as the optimum. While this approach would yield the correct answer, it is extremely inefficient. For example, if m=10 and n=20, the number of possible combinations to evaluate is about 185,000! Thus, we seek a more efficient approach.



Original, non-square equation set of constraints in standard form

Square set of equations that can be solved for the basic variables

Non-basic variables which take the values that optimize the objective. The values will be at an extreme of allowed range.

Figure 4.1. Schematic of the LP solution approach: separating into basic and non-basic variables. When bounded variables are considered, the non-basic variables can have values at either their upper or lower bounds.

4.3 The Best Corner Point

The selection of basic and non-basic variables determines the set of inequalities that are active. Figure 4.2 shows the importance of the active set in finding the optimum and gives insight into the optimum corner point. In the example shown, only one set of active constraints gives the minimum objective. The optimum corner point is determined from the gradient of the objective and constraints.

Cone: A cone is defined by a set of vectors $\mathbf{v_1}$, $\mathbf{v_2}$ and so forth. A vector **P** is contained within the cone if **P** can be expressed as a linear sum of the defining vectors ($\mathbf{v_1}$, $\mathbf{v_2}$, **…**) with *non-negative constants*.

$$\mathbf{P} = \alpha_1 \mathbf{v_1} + \alpha_2 \mathbf{v_2} + \ldots. \quad \text{With } \alpha_1, \alpha_1.. \geq 0 \qquad (4.8)$$

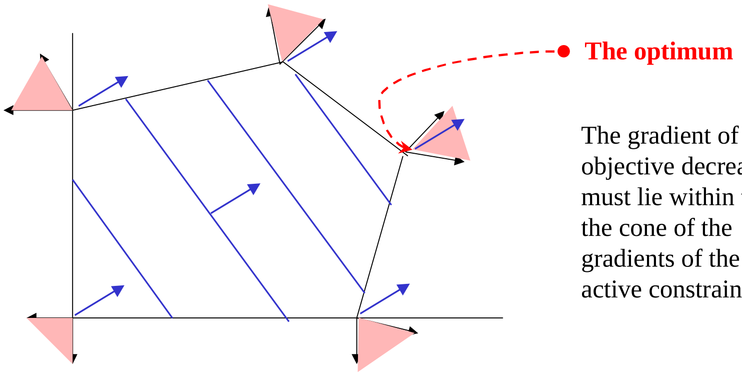
Thus, **P** is a non-negative linear combination of the vectors defining the cone.

Thus, we arrive at the key definition of the corner point at which the optimum is located.

The **optimum corner point** has the gradient of the objective function contained within the cone of the gradients of the active constraints.

$$\begin{array}{cc} \text{Basic} & \text{Non-basic} \\ \left[\begin{array}{ccc} \mathbf{a}_{11} & \cdots & \mathbf{a}_{1m} \\ \cdots & & \\ \mathbf{a}_{m1} & & \mathbf{a}_{mm} \end{array}\right. & \left.\begin{array}{ccc} \mathbf{a}_{1,m+1} & \cdots & \mathbf{a}_{1n} \\ \cdots & \cdots & \cdots \\ \mathbf{a}_{m,m+1} & \cdots & \mathbf{a}_{mn} \end{array}\right] \end{array} \begin{bmatrix} \mathbf{x}_1 \\ \cdots \\ \mathbf{x}_n \end{bmatrix} = \begin{bmatrix} \mathbf{b}_1 \\ \cdots \\ \mathbf{b}_m \end{bmatrix}$$

Original, non-square equation set of constraints in standard form



The gradient of the objective decrease must lie within the the cone of the gradients of the active constraints.

Figure 4.2. Schematic of the optimality conditions for an optimum corner point in LP.

This situation is shown in Figure 4.2. Note that this criterion enables us to determine the optimum from *local information*; we do not have to evaluate all or any other corner points.

When the specified condition is satisfied, no movement along a constraint can improve the objective function. We know for an LP that (1) the optimum must be at a corner point and (2) a local optimum is also global; therefore, the corner point is the global optimum. (Some special cases with alternative optima are discussed later.)

The principles of optimization and the special features of linear programming result in the following concepts for identifying the optimum.

1. Formulate the problem as a general LP optimization problem, equation (4.6)
2. Add slack variables to convert inequalities to equalities, equation (4.7)
3. Separate variables into basic and non-basic, Figure 4.1
4. Choose as the optimum the basic variable selection (from 3 above) that provides a feasible solution with the optimum value of the objective function, Figure 4.2

These principles provide us with excellent insight into the LP method. The student should be sure to understand the concepts shown in Figures 4.1 and 4.2, which give a geometric interpretation to the concepts. While these principles do not define an efficient method of numerical computation, they provide a foundation for the algorithm given in the next section.

5.0 The Linear Programming Simplex Algorithm

Fortunately, the principles presented in the preceding section can be employed through a very efficient algorithm. This algorithm is termed the "simplex" algorithm, which was developed by George Dantzig. He developed the simplex algorithm in the 1940's, and it remains the standard method for numerical solution of linear programs. Currently, low-cost, efficient, robust software is available to solve large systems using this method. Here, we will learn the basic algorithm, which will enable us to formulate problems and interpret results. (Regrettably, other algorithms have been named "simplex"; thus, the student is cautioned that we are here referring to the *linear programming simplex* algorithm.)

The Simplex algorithm has the following excellent features.

- If an optimum exists, the algorithm defines an iterative procedure that concludes with the optimum.
- If an optimum does not exist, the algorithm provides guidance on why not.
- As shown by experience, the method is computationally efficient.

We will need the following definitions for concepts already covered.

Basic solution: A solution to the square, non-singular set of linear equations resulting from the selection of basic variables, after setting the non-basic variables to constant values. (Currently, non-basic $x_i = 0$; later, each non-basic x_i equals its maximum or minimum allowable value.) These are also termed **Corner Points**.
Basic feasible solution: A basic solution for which all variables satisfy their bounds. (Currently, $x_i \geq 0$; later, $x_{imax} \geq x_i \geq x_{imin}$.). This is also called a **Feasible Corner Point**.
Optimal solution: A basic feasible solution for which the objective is at its optimum value.

5.1 Obtaining the initial Corner Point (Basic Feasible Solution)

The algorithm relies on a procedure that iteratively selects basic feasible solutions, but *it must start with a basic feasible solution*. Therefore, we need a method for finding a set of "m" basic vectors for which the resulting set of "m" linear equations has a solution, i.e., is non-singular. The desired approach should work for any LP problem formulation starting with equation (4.7).

The method does this by again adding variables to the problem; these are **artificial variables** that ensure that the system of equations has a solution. The variables are called artificial variables because they are not related to the problem variables; the coefficients associated with the artificial variables form an m-dimensional identity matrix. As we will see, these variables do not affect the final optimum solution, because they are eliminated quickly from the procedure; however, they are essential for finding an initial feasible corner point (basic feasible solution). This initialization procedure is shown in Figure 5.1. The resulting initial problem basis is in canonical form.

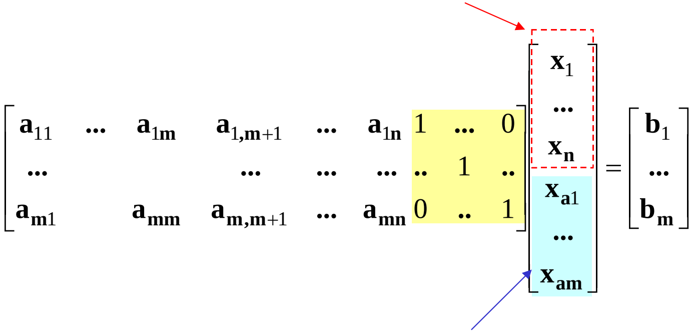
Canonical form: In a canonical form, each equation has one basic variable with a coefficient of 1.0, and all other variables have coefficients of 0.0. Also, each basic variable appears only once with a non-zero coefficient.

We have established an initial basic feasible solution (BFS) or corner point, and the LP algorithm to be described can find any BFS from an initial BFS. However, we have changed the problem by adding the artificial variables. Therefore, we must be sure that the artificial variables are not part of the final solution (if possible). We achieve this by modifying the objective function by placing a very large penalty (M>>0) on each artificial variable, as shown in the following. Recall that the penalty is positive because we are minimizing the objective function.

$z = c_1x_1 + \ldots. + c_nx_n$ the original objective function (5.1)

$z_a = c_1x_1 + \ldots. + c_nx_n + Mx_{a1} + \ldots.+Mx_{am}$ the modified objective function (5.2)

All problem variables initially set to zero



a_{11} … a_{1m} $a_{1,m+1}$ … a_{1n} 1 … 0
… … … … .. 1 ..
a_{m1} a_{mm} $a_{m,m+1}$ … a_{mn} 0 .. 1
x_1
…
x_n
x_{a1}
…
x_{am}
=
b_1
…
b_m

The artificial variables form a basis that ensures a non-singular solution to the equations.

Figure 5.1. A schematic representation of forming the initial basic feasible solution to "m" equations using artificial variables. This gives a non-singular, canonical form problem statement.

Thus, as we proceed with the optimization algorithm, the artificial variables will be eliminated because of their "cost". The value of M must be much larger than other cost coefficients to be sure that it is eliminated. When the artificial variables have been eliminated, the solution will be unaffected by this initial strategy. In fact, after all artificial variables become zero, their contribution to the problem equations and objective function can be eliminated, which reduces computation.

Now that we have an initial feasible corner point (BFS), we follow a procedure that moves along *adjacent corner points* to improve the value of the objective function.

Adjacent corner points: Adjacent corner point solutions in an LP problem with "m" variables share "m-1" of the same active constraints. They are connected by a line segment that is defined by the intersection of the "m-1" common constraint boundaries.

When we say that we “move” among corner points, we are actually performing elementary operations that do not change the solution to the set of equations.

Elementary operations: The following operations do not change the solution to an equation set; these may be performed together.

1. multiplying an equation by a (non-zero) constant.
2. adding two equations

We have encountered this concept already in the Gauss-Jordan method for solving square sets of linear equations. At each elementary operation, we have to make two decisions. First, we choose the variable to enter the basis, i.e., switch from non-basic to basic. Second, we select the variable to leave the basis. We make these choices to improve the objective function. The method is complete and the optimum has been reached when the objective function cannot be improved by moving to any adjacent corner point.

5.2 Adding the cost to the matrix

These elementary operations must be performed on the coefficient matrix and the objective function. We will display these procedures in the “simplex tableau”, which shows the calculations and intermediate results nicely and was used for hand calculations prior to the advent of digital computation. The tableau includes all model (constraint) equations and the objective function. Recall that we represent the objective function value by the variable z, giving

$$\mathbf{z} = \mathbf{c}_1\mathbf{x}_1 + \mathbf{c}_2\mathbf{x}_2 + \mathbf{......} + \mathbf{c}_{\mathbf{s}1}\mathbf{x}_{\mathbf{s}1} + \mathbf{......} + \mathbf{c}_{\mathbf{a}1}\mathbf{x}_{\mathbf{a}1} + \mathbf{.......} \tag{5.3}$$

which can be rearranged to give a constant rhs.

$$-\mathbf{z} + \mathbf{[c}_1\mathbf{x}_1 + \mathbf{c}_2\mathbf{x}_2 + \mathbf{......} + \mathbf{c}_{\mathbf{s}1}\mathbf{x}_{\mathbf{s}1} + \mathbf{......} + \mathbf{c}_{\mathbf{a}1}\mathbf{x}_{\mathbf{a}1} + \mathbf{.......]} = 0 \tag{5.4}$$

with c_i = initial (original) cost coefficient for the problem variables, given in the problem statement

c_{si} = 0 (initial cost of slacks is zero)

c_{ai} = M >> 0 (penalties for the artificial variables)

This cost expression can be included as the first equation to give the starting equation set for the LP algorithm.

$$\begin{bmatrix} -1 & c_1 & c_2 & \textbf{...} & 0 & 0 & M & \textbf{...} & M \\ 0 & a_{11} & a_{12} & \textbf{...} & \textbf{....} & \textbf{...} & 1 & 0 & \textbf{..} \\ 0 & a_{m1} & a_{mm} & a_{m,m+1} & \textbf{...} & a_{mn} & 0 & \textbf{..} & 1 \end{bmatrix} \begin{bmatrix} z \\ x_1 \\ \textbf{...} \\ x_n \\ \textbf{...} \\ x_{am} \end{bmatrix} = \begin{bmatrix} 0 \\ b \end{bmatrix} \tag{5.5}$$

or showing the sub-matrices

$$\begin{bmatrix} -1 & \mathbf{c}^{\mathbf{T}} & 0 & \mathbf{M}^{\mathrm{T}} \\ 0 & \mathbf{A} & \mathbf{I}^{*} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{z} \\ \mathbf{x} \\ \mathbf{x_s} \\ \mathbf{x_a} \end{bmatrix} = \begin{bmatrix} 0 \\ \mathbf{b} \end{bmatrix} \tag{5.6}$$

with

A = original problem coefficient matrix
b = right hand side values for the original problem constraints
c = original problem cost coefficients
z = scalar objective function
x = original problem variables

I* = coefficient matrix for slack variables containing either +1 or –1, depending on type of inequality; not necessarily square
$\mathbf{x_s}$ = slack variables
$\mathbf{x_a}$ = artificial variables

We introduce another useful distinction in terminology. All of entries in equations (5.5) and (5.6) are the "original" values from the initial problem definition. Naturally, after the elementary row operations, the values will be changed. We will refer to the changed values as the reduced values of the matrix entries. When the procedure has been completed, we will refer to the values as the "optimal reduced values". Sometimes, when the reference is obviously to the optimal solution, e.g., when referring to the software output, we will use "reduced" to describe the final values.

Recall that the values for the problem variables and slacks are initially zero, which gives m artificial variables and m equations for this initial equation set. These matrix entries can be represented in a "tableau".

5.3 LP solution algorithm using the tableau

As we move to an adjacent corner point, we must select which two variables are "exchanged", i.e., which variable is removed from the basis and which enters the basis. The following tableau rules determine the appropriate adjacent corner points until a solution is reached.

1. **Entering by the cost test for rapid rate of improvement of the objective**: The non-basic variable chosen to enter the basis has the smallest negative ("most negative") tableau cost coefficient, because increasing the variable value will most rapidly decrease the objective function for a minimization goal. Note that this rule does not consider the change allowed to the next corner point.

   Select the variable (column j=r) to enter the basis from the non-basic variables having the minimum tableau reduced cost, c_j, which must be less than zero.

   Caution: Reference books use slightly different sign conventions, which change the sign used in this test. The convention used here is consistent with Edgar et. al., 2001.

2. **Leaving by the ratio test that ensures a feasible corner point**: Compute b_i/a_{ij} for all rows i for all $a_{ij} > 0$), with j = the variable. Select the row with the minimum value of b_i/a_{ij} as designating the basic variable x_i to leave; note that only one variable is related to each equation (row) in canonical form. Recall that the leaving variable is decreased as the entering variable increases. Because the variables are limited to be non-negative, we select the variable with the smallest ratio, which selects the variable that reaches zero with the smallest change in the entering variable. Thus, the leaving variable will have a value of zero (and become a non-basic variable) after the entering variable has entered. If

we chose a variable with a larger ratio, one of the non-basic variables would have a negative value after the pivot, which would represent an infeasible corner point.

The leaving variable has the row i given by $\min\limits_{a_{is}>0}\left(\dfrac{b_i}{a_{is}}\right)$ **with s the column entering.**

3. **Pivot on the entering-leaving intersection to regain the canonical form**: We now pivot to result in the entering variable having a coefficient of 1.0 in the pivot row and 0.0 in all other rows. Since we began with a canonical formulation, we will continue with a canonical formulation, with each basic variable having only one non-zero coefficient (and that being 1.0). We will use a_{rs} to designate the pivot coefficient and E_r the r^{th} equation.

a. Replace the r^{th} row (equation) E_r with E_r/a_{rs}.
b. For all other rows (equations), replace E_i with $E_i - (E_r)a_{is}/a_{rs}$.

4. **Test for optimality at the new corner point**: If the cost coefficients of all the non-basic variables are positive, increasing any non-basic from zero will increase the objective function. Therefore, no further improvement in the objective is possible. The current result is optimal.

a. If all non-basic $c_j \geq 0$, an optimal solution is found. Stop.
b. If at least one non-basic $c_j < 0$, continue by returning to 1. above.

How well does this simplex algorithm work? Recall that a problem having 20 variables and 10 equations had about 185,000 corner points.

$$\binom{n}{m} = \frac{n!}{(n-m)!m!} \approx 185{,}000 \quad \text{(with n=20 and m=10)} \tag{5.7}$$

If we used exhaustive search, we would have to check every one. The simplex algorithm solves problems of this size in about ***30 iterations*** (Winston, 1994; page 132)! There is no theoretical reason for the excellent performance of the simplex method, since it searches along the boundary of feasible corner points, but experience has shown that it performs well on nearly all real-world problems (Shamir, 1987). (It is possible to formulate a “trick” problem for which the simplex will perform poorly.)

5.4 Sample Tableau for a small LP problem

We will conclude this section with an example showing all tableaus. The reader will likely have to review the algorithm steps above while following the solution tableaus given below.

Example 5.1: Solve the following linear programming problem and show all intermediate tableaus (Winston, 1994; page 164).

1. The mathematical problem is stated in the following.

$$
\begin{aligned}
&\mathbf{min}\ \ \mathbf{z} = 2\mathbf{x}_1 + 3\mathbf{x}_2\\
&\mathbf{s.t.}\\
&0.5\mathbf{x}_1 + 0.25\mathbf{x}_2 \le 4\\
&\ \ \mathbf{x}_1 + \ \ \ 3\mathbf{x}_2 \ge 20\\
&\ \ \mathbf{x}_1 + \ \ \ \ \mathbf{x}_2 = 10\\
&\mathbf{x_i} \ge 0\ \ \mathbf{for}\ \ \mathbf{i} = \mathbf{1,2}
\end{aligned}
$$

2. We convert any inequalities to equalities. In this case, row 1 needs a slack variable with a (+1) coefficient and row 2 a slack with a (-1) coefficient (surplus variable). Row 3 is already an equality.

$$
\begin{aligned}
&\mathbf{min}\ \ \mathbf{z} = 2\mathbf{x}_1 + 3\mathbf{x}_2\\
&\mathbf{s.t.}\\
&0.5\mathbf{x}_1 + 0.25\mathbf{x}_2 + \mathbf{s}_1 = 4\\
&\ \ \mathbf{x}_1 + \ \ \ 3\mathbf{x}_2 - \mathbf{s}_2 = 20\\
&\ \ \mathbf{x}_1 + \ \ \ \ \mathbf{x}_2 \qquad\ = 10\\
&\mathbf{x_i} \ge 0\ \ \mathbf{for}\ \ \mathbf{i} = \mathbf{1,2}
\end{aligned}
$$

3. Now, we modify the formulation to achieve a canonical form, in which an initial feasible corner point (basic feasible solution) is easily achieved. We see that row 1 already has a variable with a coefficient of +1, the slack. Therefore, we need to add artificial variables to only rows 2 and 3.

$$
\begin{aligned}
&\mathbf{min}\ \ \mathbf{z} = 2\mathbf{x}_1 + 3\mathbf{x}_2\\
&\mathbf{s.t.}\\
&0.5\mathbf{x}_1 + 0.25\mathbf{x}_2 + \mathbf{s}_1 \qquad\qquad\quad = 4\\
&\ \ \mathbf{x}_1 + \ \ \ 3\mathbf{x}_2 \qquad - \mathbf{s}_2 + \mathbf{a}_2 \quad\ = 20\\
&\ \ \mathbf{x}_1 + \ \ \ \ \mathbf{x}_2 \qquad\qquad\qquad + \mathbf{a}_3 = 10\\
&\mathbf{x_i} \ge 0\ \ \mathbf{for}\ \ \mathbf{i} = \mathbf{1,2}
\end{aligned}
$$

We see that the initial basis is [s1 a2 a3].

4. We have added the artificial variables, and we want to ensure that they do not appear in the optimal solution, since they were introduced only to find an initial feasible corner point. Therefore, we add large penalties; since the problem is a minimization, the penalties are positive.

$$\begin{array}{l}
\mathbf{min}\ \ \mathbf{z} = 2\mathbf{x}_1 + 3\mathbf{x}_2 + \mathbf{Ma}_2 + \mathbf{Ma}_3 \\
\mathbf{s.t.} \\
0.5\mathbf{x}_1 + 0.25\mathbf{x}_2 + \mathbf{s}_1 \qquad\qquad\quad = 4 \\
\quad \mathbf{x}_1 + \quad 3\mathbf{x}_2 \quad - \mathbf{s}_2 + \mathbf{a}_2 \quad = 20 \\
\quad \mathbf{x}_1 + \quad \mathbf{x}_2 \qquad\qquad\quad + \mathbf{a}_3 = 10 \\
\mathbf{x}_\mathbf{i} \geq 0\ \ \mathbf{for}\ \ \mathbf{i} = \mathbf{1,2}
\end{array}$$

5. We check to see that all rhs values are greater than or equal to zero, which is the case. If any were not, we would multiply the row by (-1).

6. We now place all variables and values into the tableau. Recall that we rearrange row zero to have a constant rhs.

$$-\mathbf{z} + 2\mathbf{x}_1 + 3\mathbf{x}_2 + \mathbf{Ma}_2 + \mathbf{Ma}_3 = 0$$

z	x_1	x_2	s_1	s_2	a_2	a_3	rhs	Basic variable	Ratio
-1	2	3	0	0	M	M	0	z	
0	.5	.25	**1**	0	0	0	4	s_1	---
0	1	3	0	-1	**1**	0	20	a_2	---
0	1	1	0	0	0	**1**	10	a_3	---

7. We are close to beginning the pivoting operation. However, we require all basic variables to have zero elements in the objective (row 0) for a canonical form. Therefore, we must eliminate the "M's" in row zero for the basic variables a_2 and a_3 without changing the solutions to the equations. We see that we can achieve this by adding the following to row 0: (1) row 2 times (-M) and (2) row 3 times (-M). These steps will cancel the penalties on variables a_2 and a_3 and give zero elements for the artificial variables in the initial tableau. (Note that s_1 has a zero value in row 0.)

<u>Initial tableau</u>

This (x_2) is the variable entering the basis (smallest value < 0).

z	x_1	x_2	s_1	s_2	a_2	a_3	rhs	Basic variable	Ratio
-1	-2M+2	-4M+3	0	M	0	0	-30M	z	
0	.5	.25	**1**	0	0	0	4	s_1	16
0	1	3	0	-1	**1**	0	20	a_2	20/3
0	1	1	0	0	0	**1**	10	a_3	10

Pivot element, a_{rs}

This (a_2) is the variable leaving the basis. (Smallest value of b_i/a_{ij} for entering a_{ij} >0)

Now, we apply the pivoting rules to determine the entering and leaving variables, as shown above.

8. We perform the pivoting calculations to yield a value of 1.0 for the pivoting coefficient and zeros in all other coefficients in the column.

Second tableau

This (x_1) is the variable entering the basis (smallest < 0).

z	x_1	x_2	s_1	s_2	a_2	a_3	rhs	Basic variable	Ratio
-1	-2M/3+1	0	0	-M/3+1	+4M/3-1	0	-10M/3-20	z	
0	5/12	0	1	1/12	-1/12	0	7/3	s_1	28/5
0	1/3	1	0	-1/3	1/3	0	20/3	x_2	20
0	2/3	0	0	1/3	-1/3	1	10/3	a_3	5

Pivot element, a_{rs}

This (a_3) is the variable leaving the basis. (Smallest value of b_i/a_{ij} for entering $a_{ij} >0$)

9. Again, we perform the pivoting calculations to yield a value of 1.0 for the pivoting coefficient and zeros in all other coefficients of the column.

Third tableau

All reduced costs are greater than 0.0. The objective cannot be decreased by changing the basis, i.e., moving to an adjacent corner point. We have found the *optimum*!

z	x_1	x_2	s_1	s_2	a_2	a_3	rhs	Basic variable	Ratio
-1	0	0	0	1/2	-1/2+M	-3/2+M	-25	z = 25	
0	0	0	1	-1/8	1/8	-5/8	1/4	$s_1 = 1/4$	---
0	0	1	0	-1/2	1/2	-1/2	5	$x_2 = 5$	---
0	1	0	0	1/2	-1/2	3/2	5	$x_1 = 5$	---

10. In this problem, the optimum was reached after the artificial variables were eliminated. Typically, (many) additional corner points would have to be evaluated using the pivoting procedure.

The solution is $x_1 = 5$, $x_2 = 5$; slack variables values are $s_1 = ¼$ and $s_2 = 0$. The objective function value is z = 25.

The non-basic variables are equal to zero: $s_2 = 0$, $a_2 = 0$, $a_3 = 0$. Since all artificial variables are zero, the solution is feasible. Since the reduced costs of the non-basics are not zero, no alternative solutions exist. Since s_2 is zero, the second inequality constraint is active (or binding); since $s_1 > 0$, the first inequality is inactive (not binding).

Now that the simplex method has been presented, you might be tempted to program the algorithm. Two recommendations are offered. First, programming your own algorithm is an excellent approach for *learning*; however, simple numerical implementations will fall victim to numerical errors. Therefore, the second recommendation is to use commercially available codes for engineering research and practice. These codes have been carefully developed to handle many numerical difficulties caused by ill-conditioning and manipulating large matrices; therefore, the student is advised against developing software for practical use, without considerable further study in optimization mathematics and numerical methods. In addition, Appendix B gives sources for educational, interactive computer programs that allow you to make key decisions, e.g., the pivot element, and then allow the computer to perform the tedious calculations automatically. They also provide some coaching, especially for clearly incorrect user choices.

6.0 Extensions and Special cases (Cautions on LP Weird Events)

We have learned the essential aspects of the simplex algorithm. In this section, we first introduce several extensions to the simplex just described. These extensions provide greater flexibility to the engineer in formulating models and solving problems. These extensions are available in essentially all software solvers, so they are briefly explained without extensive theoretical development.

Then, several important special cases are discussed. They are not just mathematical anomalies!

Since these special cases occur in practice, the engineer must be able to recognize their occurrence, modify the formulation (if possible) and determine good actions based on the optimization study. The naive user can make serious mistakes!

6.1 Simplex Extensions

6.1.1 General variable bounds: We have assumed that the variables must be greater than zero and have no upper bound. These are not serious limitations, as we could always reformulate a model to abide by these restrictions. For example, an upper bound could be achieved for any variable by adding a constraint to the problem, e.g., $x_{10} \leq 45$. However, the reformulation would be larger, require longer computing times and be prone to errors by the analyst. Therefore, software systems allow the user to define variables with any values for lower and upper bounds, so long as the upper is greater than or equal to the lower bound.

Recall that the non-basic variables had values of zero in the “simple” simplex method just described. When variables have lower and upper bounds, the non-basic variables are assigned the bound value that improves the objective function the most, either their lower or upper bounds. No user input is required to active these features.

Lower bounds: Not all variables have lower bounds of zero; for example, the lower bound for a temperature could -20 °C. One approach to this situation would be to define two new variables x’ and x’’, then, define

$$x = x' - x'' \qquad \text{with } x' \geq 0 \text{ and } x'' \geq 0 \tag{6.1}$$

Thus, the substitution of x’ and x’’ for x would leave an LP problem with all non-negative variables (Winston, 1994; page 175). The software system automatically makes all substitutions; the user simply defines the appropriate values for the variable bounds.

Upper bounds: The simplex method includes variable upper bounds in the method for determining how much the entering variable can change. Limits to the size of the change of the entering variable are now (Winston, 1994; page 587; Hillier and Lieberman, 2001; page 317):

1. One of the current basic variables becomes zero (or its lower bound), as always
2. The entering variable cannot exceed its upper bound.
3. One of the current basic variables increases to its upper bound.

6.1.2 Efficient Simplex: The simplex algorithm and tableau previously described requires that all tableau elements be calculated at every iteration. With large problems, these calculations can be very time-consuming. Improved approaches involve much reduced calculations at each iteration that does not change the basic concepts of the simplex. These require a "revised simplex" and "product form of the inverse" techniques (Winston, 1994; page 554; Hillier and Lieberman, 2001; page 202). In addition, advanced matrix methods can be employed for "sparse" problems, because only a small number of elements of the coefficient **A** matrix are non-zero in large problems.

6.1.3 Restart Strategies: For large problems, starting from the initial canonical form with numerous slack and artificial variables is very inefficient. Sometimes we want to solve a related problem based on the results of the initial problem formulation. Therefore, software systems provide the ability to restart with the information about the last optimal solution. It is possible to change right hand side coefficients, objective coefficients, constraint coefficients, or add/subtract some constraints (Rao, 1996; Chapter 4). We will not cover the details of these approaches, but they can greatly speed the solution of similar large LPs solved sequentially.

6.2 Special cases (Cautions on LP Weird Events)

6.2.1 No Feasible Region: As shown in Figure 6.1, an LP problem could be formulated in a manner that results in no feasible region. This could occur for two reasons. First, the engineer has made a mistake in the formulation. Second, very stringent requirements are placed on the performance of the problem system, so that no solution actually exists. We need to be able to recognize when no feasible solution exists and to change the formulation, if appropriate.

Diagnosis: We can recognize when the problem has no feasible solution when the final optimal tableau has one or more *artificial variables* in the basis. Since these variables have very large penalties for being non-zero (in the basis), the only reason for them remaining in the basis would be feasibility. Therefore, the problem has no feasible solution.

Remedial action: In many cases, we would like to learn how we could achieve a feasible solution. One good way to do this is to add additional variables to inequality constraints that have a substantial penalty (but much less than "M"). The solution will be able to find a least costly "feasible" solution, even though it could include violations of the original inequality constraints.
The approach is explained under "goal programming" in Section 9.6.

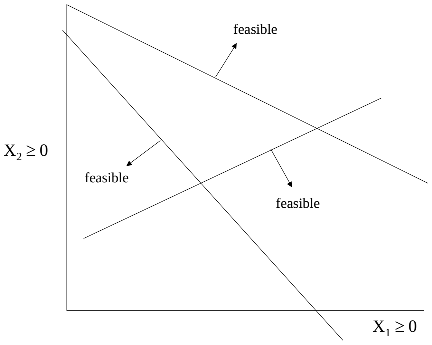


Figure 6.1. A set of LP constraints that yield no feasible region.

6.2.2 Unbounded Solution: As shown in Figure 6.2, the solution to an LP problem could be unbounded, so that one or more variables could increase to infinity and the objective function decrease to minus infinity. This is always a result of a formulation error, because no variable in a real problem can increase without limit.

Diagnosis: The symptom occurs when selecting the variable to leave the basis, which is determined using the following equation.

$$\mathbf{x_i} = \overline{\mathbf{b_i}} - \overline{\mathbf{a_{is}}}\mathbf{x_s} \tag{6.2}$$

Note that if a_{is} is less than zero, the variable x_i can increase without limit, without causing x_i to decrease to its lower bound of zero. If the reduced cost, c_j, is less than zero for this variable, its increase will be beneficial, because it will decrease the objective function. Thus if for a column with $c_j<0$, all $a_{is}<0$, an unbounded solution will occur.

Remedial action: We should seek a modification to the problem definition that limits the feasible region. For example, in a production problem, we might have inadvertently forgotten the market (sales) limitation, or in a personnel allocation, we might have forgotten the limitation to the availability of workers.

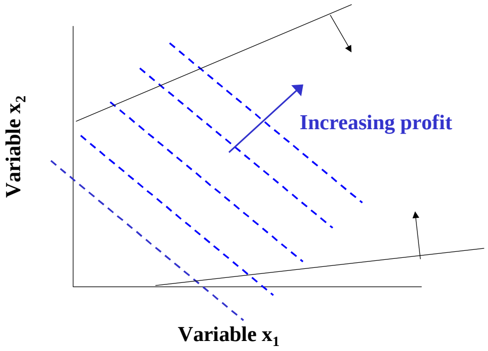


Figure 6.2. A set of LP constraints and objective function that yield an unbounded LP.

6.2.3 Tie on entering or leaving criteria: It is possible that a numerical tie could occur in the criteria for variables entering or leaving the basis. In both cases, the tie can be broken arbitrarily. Actually, the tie in variables leaving the basis could theoretically result in an unending cycle; however, this does not occur in real problems, so that most software does not include special logic (Winston, 1994; page 160).

6.2.4 Multiple (Alternative) Optimal Solutions: The solution of an LP can have multiple optima. The situation is shown graphically in two dimensions in Figure 6.3. We see that the objective function is parallel to one of the active constraints. Thus, either of two corner points – or any point on the line connecting them – has the same value of the objective function. Thus, many (an infinite number) of values of **x** yield the same objective value.

Diagnosis: There are two ways to identify multiple optimal corner points.

1) This situation can be diagnosed by evaluating the reduced costs of all non-basic variables in the final optimum tableau. If any non-basic variable has a reduced cost of 0.0, it can enter the basis (and another can leave as a consequence) with no effect on the objective function. Thus, the diagnosis looks at the optimal solution and determines if any non-basic reduced cost is = 0. If yes, then alternative solutions exist.
2) An additional symptom can be recognized by observing Figure 6.3. We see that the right hand side (RHS) of one or more of the active constraints at (either) optimal solution has a zero marginal value and the marginal value has a non-zero range in both directions.

Remedial action: Alternative solutions might not be a concern, since we will find at least one policy that achieves the best value of the objective. However, we should find all solutions and select the optimum that is truly best, because the objective function might not represent all goals, i.e., alternative solutions are not really equivalent. Common situations concerning alternative solutions are summarized on the following.

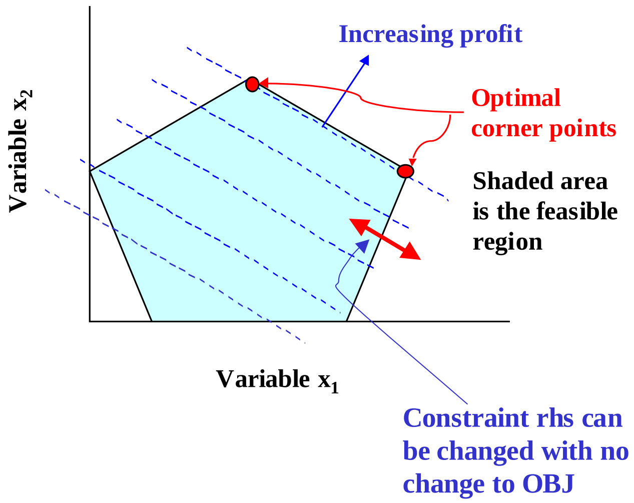

Increasing profit
Optimal corner points
Shaded area is the feasible region
Variable x_2
Variable x_1
Constraint rhs can be changed with no change to OBJ

Figure 6.3 Schematic of LP with alternative optima corner points (anywhere on the connecting line is also optimal). The user must employ additional criteria to select the best solution.

- Perhaps, one solution is close and one far from our current variable values. In some situations there are "hidden costs" for changing conditions that might not be represented in the model. Examples are changing plant operations, which could lead to poor quality products during transitions, and challenges in communicating changes to a schedule that has been adopted in an organization.

  Dynamic optimization occurs in many fields, such as scheduling personnel for airlines, deciding when to produce various products in a flexible manufacturing plant, and in automatic process control where we optimize a trajectory to the set point. When solving dynamic optimization, we often resolve the problem periodically. We solve for a "long" time in the future, but we implement only the solution variables related to the current time. As we resolve the problem, alternative optima could cause large changes in the variable values from solution to solution. This is termed "nervousness" and is to be avoided.

  To select an alternative optimum that is "close", we could add a term to the objective that (slightly) penalizes changes from the current policy being implemented; this would "break ties" and select the least disruptive solution.

- Perhaps, we have had good experience with one solution while we have had either poor experience or no experience with other solutions. Naturally, we will select the solution close to values where we believe that model is accurate and which has given good results in the past when implemented.

6.2.5 Redundancy of Constraints: We have considered "normal" cases in which the active constraints specify the solution, i.e., the values of the basic variables. Constraint redundancy involves a case in which one or more "extra" constraints are active at the solution, with these "extra" constraints not influencing the solution. This situation is shown in Figure 6.4. In this case, the optimum would be defined completely by either constraints 1 and 2 or 2 and 3. The solution found is correct, but standard sensitivity analysis can be misleading. To see why, consider the following two situations.

a. An ***increase*** in the rhs of constraint 3: In this situation, the objective function value does not change. Thus, the sensitivity is 0.0.

b. A ***decrease*** in the rhs of constraint 3: In this situation, the feasible region is reduced, and the objective function is increased (assuming this is a minimization problem). Since the optimum values of the variables change, the sensitivity is non-zero.

We use linear programming to make decisions about how to improve the solution via post-optimal analysis, and clearly, redundant constraints are difficult to analyze. This has received considerable attention and guidance is available for the user (Rubin and Wagoner, 1990) and the system developer (Koltai and Terlaky, 2000).

Diagnosis: We can determine when this situation occurs by analyzing the sensitivity of the optimum (see Section 7). Sensitivity analysis gives the range of the rhs of every constraint over which the value can be changed without changing the basis. If this range has zero as either its maximum or minimum allowable change, the solution has redundant constraints, and the standard sensitivity output from software should not be used.

Remedial Action: The values of the objective function and variables are reliable for the solution. However, the sensitivity depends upon the direction of the change; therefore, we must be aware of the following caution in analyzing the results.

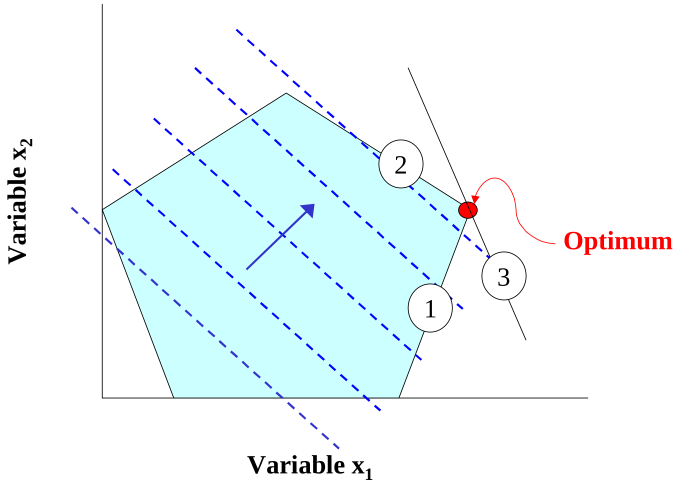


Figure 6.4. Schematic of an LP problem with a redundant constraint at the solution. Care must be taken when using the sensitivity analysis results.

> The engineer should not rely upon standard sensitivity analysis results when the optimum experiences constraint redundancy. The engineer should evaluate sensitivity by executing "delta" cases, each being a reoptimization of the problem with the appropriate parameter(s) changed.

We close with a comment about the likelihood of redundancy. As we work with systems over time and invest to improve their performance, we increase capacities where (1) the objective is limited and (2) we receive a large improvement with a small investment. This process leads to a situation in which many inequality constraints are nearly active, and large investments are required for further improvement. In this situation, constraint redundancy often occurs. This is not just a mathematical peculiarity; it is a challenge for practitioners.

6.2.6 Degeneracy of Constraints: Constraint degeneracy involves a case in which more inequality constraints are active at the solution than the dimension of the problem. For example, a problem could have two degrees of freedom (number of variables - number of equality constraints = 2) and three inequality constraints active at the optimum. Thus, constraint redundancy (subsection 6.5.5) involves degeneracy, but other situations also occur. As another example, the pyramid feasible region in Figure 6.5 has an optimum at the top peak. In this situation, the dimension of the problem is three, but four constraints are active at the optimum. Note that *none* of the inequality constraints are redundant, removing any will change the optimal solution and the objective function.

Diagnosis: The simplex algorithm will select three of the four constraints in Figure 6.5. Degeneracy can be diagnosed by observing that changing the basis by making the fourth constraint active and one of the original three inactive does not change the solution.

Remedial Action: The value of the objective function and variables is reliable for the solution. However, the sensitivity depends upon the direction of the change; therefore, we must again be aware of the following caution in analyzing the results.

> The engineer should not rely upon standard sensitivity analysis results when the optimum experiences constraint degeneracy. The engineer should evaluate sensitivity by executing "delta" cases, each being a reoptimization of the problem with the appropriate parameter(s) changed.

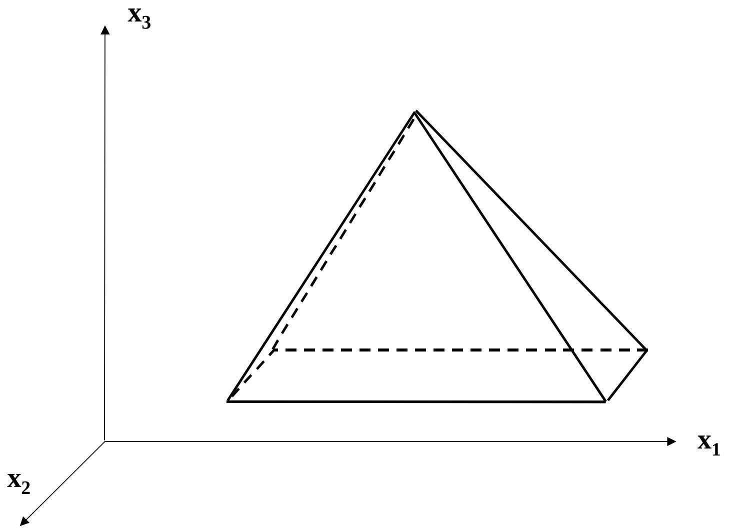


Figure 6.5. Schematic of an LP problem with the optimum at the top corner point, which has degenerate constraints (without redundancy). Care must be taken when using the sensitivity analysis results.

7.0 Sensitivity and Range Analysis of LP Solutions

We have seen that the simplex LP algorithm provides solutions to very complex linear problems with inequality constraints. The good news does not stop there; the method also provides valuable sensitivity information about the optimal solution. Sometimes, we term this "post-optimal" analysis.

7.1 Importance of sensitivity analysis

We seek a full understanding of the solution that extends beyond the values of the objective and variables at the optimum point. We want to understand how sensitive the result is to changes in input data and how the values change with these data changes. A few typical uses of sensitivity analysis are given in the following, and we should note that these results are available with the solution at essentially no cost in computation!

- **Sensitivity to the model**: We want to determine how sensitive the results are to the system model. When we find that a small change in a parameter could change the decision, we will have to carefully investigate the uncertainty in that parameter.
- **Sensitivity to the scenario**: When given an opportunity to change decisions, we can use the sensitivity values to see if the new opportunity could be attractive. For example, if we choose Feed A over feed B, we will also learn how much the price of feed B must decrease to make it an attractive choice. This information would be important in negotiating with the supplier.
- **Range of validity**: The sensitivity values are accurate over a limited range of values of the specific parameter. We will determine the range over which our solution is valid.

The sensitivity results discussed in this section could be determined by changing a parameter by a small amount and solving another optimization problem. However, we will initially concentrate on information that is available with the base case solution, because (1) this information is useful in engineering practice, (2) it reinforces our understanding of linear programming, (3) it provides insight that helps us diagnose potential problems that were discussed in Section 6, and (4) the results are easily determined.

Again, sensitivity results not provided by the methods in this section can be evaluated with numerical differentiation of multiple optimization cases. For example, the reoptimization procedure is required for sensitivity to changes

- in the left-hand side coefficients (the "A" matrix),
- parameter changes that require a basis change to find the optimal solution
- in a solution that involves constraint degeneracies (including redundancies)

We will concentrate on results that are reported with the LP solution in many available software packages. We will begin with changes in only one parameter and extend the results to multiple parameters. We conclude with approximations to sensitivity results when a basis change is required.

7.2 Sensitivity analysis of the optimum without a basis change

Sensitivity tells us the effect of a small change in one or more parameters that were assumed constant when finding the optimum. Thus, the sensitivity is $\Delta z/\Delta \alpha$, with α = a parameter like a cost or rhs constant. We could evaluate the sensitivity in several ways.

1. Sensitivity = $\Delta z/\Delta \alpha$ with all $\mathbf{x}$ = constant
2. ➡ Sensitivity = $\Delta z/\Delta \alpha$ with all basic variables, $\mathbf{x_B}$, allowed to change so the results represent an optimal solution at the same corner point for the modified problem.

We select the second option (2), since we want to learn the effect of a change on the *optimum*. Also, we recognize that using the approach in (1) above would lead to infeasibility for many cases, since the optimal solution is located at a corner point.

The sensitivity is reported using the optimal basis and evaluates the range and effects of parameter changes at the same corner point, i.e., with the basis - without requiring a basis change.

We should be careful when explaining sensitivities. The sensitivities that we are evaluating are the derivatives of the objective (or variable) with respect to a parameter, for example dz/db$_4$. This is often explained as the effect on the optimal objective of a change in b$_4$ of "1 unit". In linear programming, the derivative is constant until a basis change; therefore, this imprecise explanation is acceptable if the *basis does not change for a 1-unit change in the parameter*. Remember that a value of 1.0 is not small when the units are millions of dollars or thousands of tons of production!

When the parameter changes are small enough (as defined later), the basis does not change and the sensitivity information is available with very limited calculation. This becomes clear when we consider the equation below for the optimal solution of the LP.

$$\left\lfloor \mathbf{A}^*_{\mathbf{NB}} \right\rfloor \left\lfloor \mathbf{x}^*_{\mathbf{NB}} \right\rfloor + \left\lfloor \mathbf{A}^*_{\mathbf{B}} \right\rfloor \left\lfloor \mathbf{x}^*_{\mathbf{B}} \right\rfloor = \left\lfloor \mathbf{b}^* \right\rfloor \tag{7.1}$$

with $\mathbf{A}^*_{\mathrm{NB}}$ = The coefficient matrix for non-basic variables at the solution
$\mathbf{A}^*_{\mathrm{B}}$= The coefficient matrix for basic variables at the solution
$\mathbf{x}^*_{\mathrm{NB}}$ = The vector of non-basic variables at the solution (at a bound)
$\mathbf{x}^*_{\mathrm{B}}$ = The vector of basic variables at the solution (between their bounds)
$\mathbf{b}^*$ = The values of the RHS at the solution

Note that “at the solution” indicates the values after all elementary operations; these are the “reduced” values, not the original values in the initial problem definition. All of these values are available in the *final tableau*.

Also, we assume that the system is *not degenerate*. If it is degenerate, the sensitivity results here are suspect, as discussed in Section 6.2.5 where a method for testing for degeneracy is provided and recommendations for sensitivity analysis are provided.

We recall that the non-basic variables are constant at either their lower or upper bound; thus, only the values of the basic variables change for changes in parameters (small enough so that the basis does not change). Also, the reduced costs of the basic variables are zero.

Finally, we emphasize that the sensitivities have *engineering units*, so that we cannot compare magnitudes of numbers directly. The engineer must determine the units and carefully interpret the meaning of the sensitivities.

7.3 One-At-A-Time Parameter Changes

In this subsection, we consider the effects of a change to the value of a *single parameter*. This helps us analyze the solution and understand effects of changes.

7.3.1 Active inequality constraint RHS change: We would like to understand the effects of changes in the rhs values of the active inequality constraints. We can identify the active inequalities because they have non-zero marginal values and zero-valued slack variables. First, we determine how large a change can occur (in either direction) without requiring a basis change. This situation is shown schematically in Figure 7.1. The value of constraint 1 can be increased to 1b or decreased to 1a without a basis change, i.e., the same corner point being optimal. Certainly, the objective function and variable values change, and these can be easily determined from the simple calculations shown below.

Effect of the basic variables

$$\left[\mathbf{A}^*_{\mathbf{NB}}\right]\left[\cancelto{0}{\Delta\mathbf{X}^*_{\mathbf{NB}}}\right]+\left[\mathbf{A}^*_{\mathbf{B}}\right]\left[\Delta\mathbf{X}^*_{\mathbf{B}}\right]=\left[\Delta\mathbf{b}^*\right] \tag{7.2}$$

$$\left[\mathbf{A}^*_{\mathbf{B}}\right]\left[\Delta\mathbf{X}^*_{\mathbf{B}}\right]=\begin{bmatrix}0\\ \Delta\mathbf{b}_{\mathbf{i}}\\ 0\\ 0\end{bmatrix} \tag{7.3}$$

$$\left[\Delta X^*_B\right]=\left[A^*_B\right]^{-1}\begin{bmatrix}0\\ \Delta b_i\\ 0\\ 0\end{bmatrix}=\begin{bmatrix}\Delta x_{B1}\\ \Delta x_{B2}\\ \cdot\cdot\\ \Delta x_{Bm}\end{bmatrix} \tag{7.4}$$

{This is a “marginal mechanism” required to stay at the optimum when $\Delta\mathbf{b}_i$ occurs.}

Effect on the non-basic variables: $\left[\Delta\mathbf{X}^*_{NB}\right]=\mathbf{0}$ (7.5)

Effect on the objective function: $\Delta z=(\Delta b_i)(c_{Ri})$ (7.6)

with c_{Ri} the sensitivity of the inequality i

Therefore, we can determine the change in variables ($\mathbf{x_B}$) and objective (z) for a change in the rhs value of any single inequality constraint. The change in the variables is quite useful because it provides the changes in variables in a *coordinated manner* that maintains optimal results. Consider the situation in which we are optimizing the operation of a plant and we are not sure of the value of a constraint, e.g., the maximum reflux in a distillation tower. The result in equation (7.4) gives how all basic variables (not at bounds) should be changed to retain feasibility and optimality.

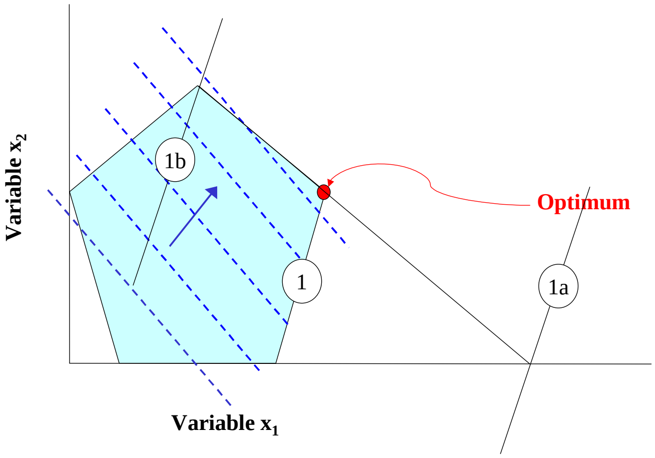


Figure 7.1. A schematic showing the changes that can occur to constraint 1 that do not require a basis change.

7.3.2 Inactive inequality constraint RHS change: We would also like to learn the maximum changes in the rhs of inactive inequality constraints. Within this range, they would not affect the solution, i.e., the values of the variables and the objective function. This is easily determined as the value of the slack variable associated with the constraint, because when the slack is zero, the constraint is active.

Effect on the basic variables: $$\left[\Delta \mathbf{X}_B^*\right] = \mathbf{0} \tag{7.7}$$

Effect on the non-basic variables: $$\left[\Delta \mathbf{X}_{NB}^*\right] = \mathbf{0} \tag{7.8}$$

Effect on the objective function: $$\Delta z = (\Delta b_i)(c_{Ri}) = 0 \tag{7.9}$$

with c_{Ri} the sensitivity of the inequality i = 0

Standard LP software reports the constraint sensitivity information in tabular form. For every constraint, the following are reported. Naturally, the format depends upon the product.

Constraint ID	**Status** **(Active/inactive)**	**slack**	**Shadow price (sensitivity of rhs)**	**Maximum allowable increase (AI)**	**Maximum allowable decrease (AD)**
Max. Reflux flow	Active	0	3.74	47	123
Max. Pump 7	Inactive	321	0	1.0E30	321

7.3.3 Equality constraint RHS change: Equality constraints are always satisfied, so that they are always "active". Thus, changing the rhs of an equation always affects the basic variables and potentially, the objective. The sensitivity results are the same as given in equations (7.4) to (7.6).

7.3.4 Cost change for non-basic variable: For a minimization problem, the reduced cost for a non-basic variable at the optimum is positive. For this variable to enter the basis, the reduced cost must be negative. Therefore, the reduced cost of the non-basic variable must change from its optimal value to 0.0 to effect a basis change, i.e., to have an effect on the solution. For a cost change greater than –(margin variable value), no change occurs to the problem variables or to the objective function. If the cost change is less than (– variable margin value), we must return to the tableau and calculate the pivots to the new optimum.

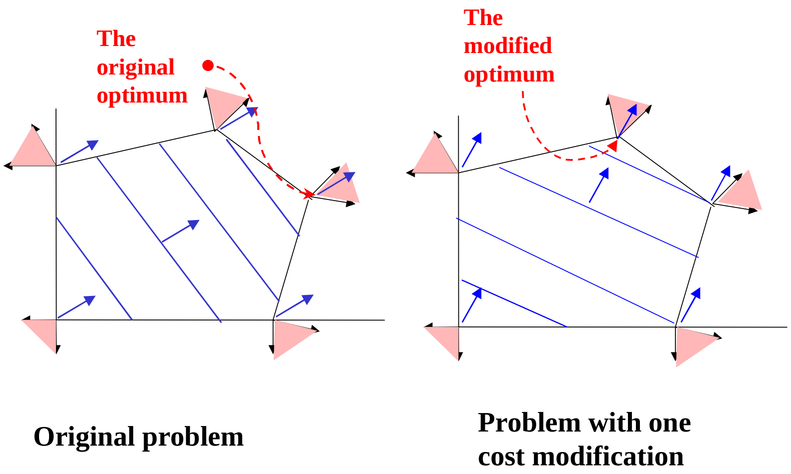
7.3.5 Cost change for a basic variable: The costs of the basic variables affect the slope of the constant profit (objective) lines and the gradient of the profit. To retain the same basis, the gradient of the profit can change, as long as the gradient remains within the cone of the corner point constraints. The concept is demonstrated in Figure 7.2. This sets the maximum and minimum allowable changes in the basic cost. Within these changes, the optimum corner point and variable values do not change, but the objective value changes.

Effect on the basic variables: $$\left[\Delta \mathbf{X}_B^*\right] = \mathbf{0} \tag{7.7}$$

Effect on the non-basic variables: $$\left[\Delta \mathbf{X}_{NB}^*\right] = \mathbf{0} \tag{7.8}$$

Effect on the objective function: $$\Delta z = (\Delta c_j)(x_j) \tag{7.9}$$

with Δc_j the change in the original cost of the variable j



Original problem

Problem with one cost modification

Figure 7.2. Schematic of the effect of a cost change to a basic variable. It the case shown, the cost parameter change was large enough to cause a basis change.

7.4 Multiple Parameter Changes - 100% Rules

In a few cases, we can reach strong conclusions about the effects of more than one parameter. Some of these methods are presented here (Bradley, et al, 1977; Winston, 1994; page 262). This collection of approaches is generally referred to as the **100% Rules**, because they determine the combined (100%) effect of multiple changes. If the "worst case" combined effect does not change the basis, the sensitivity analysis can be performed using the base case optimization results. If the "worst case" combined effect could involve a basis change, a re-optimization with all parameter changes is required.

7.4.1 Objective function original costs: Case 1. All non-basic variables (reduced costs ≠ 0)

We can calculate the following metrics, which measure the fraction of the maximum allowable change that has occurred.

If $\Delta c_j > 0$ $r_j = \Delta c_j / AI_j$ with AI the maximum allowable increase w/o a basis change

(7.10)

If $\Delta c_j < 0$ $r_j = -\Delta c_j / AD_j$ with AD the maximum allowable decrease w/o a basis change

If each of the $r_j \leq 1.0$, the effect of all the changes will not cause a basis change. Therefore, the variables will be unchanged, and the modified objective function is easily calculated.

Effect on the basic variables: $$\left[\Delta\mathbf{X}_B^*\right] = \mathbf{0} \tag{7.11}$$

Effect on the non-basic variables: $$\left[\Delta\mathbf{X}_{NB}^*\right] = \mathbf{0} \tag{7.12}$$

Effect on the objective function: $$\Delta z = \sum_j \left(\Delta c_j\right)\!\left(x_j\right) \tag{7.13}$$

with Δc_j the change in the original cost of the variable j

7.4.2 Objective function original costs: Case 2. One or more basic variables along with non-basic variable costs

It this case, we calculate the same metrics as above and apply the following **100% rule**.

If $\Sigma\ r_j \leq 1.0$, we are sure that the basis has not changed.

Effect on the basic variables: $$\left[\Delta\mathbf{X}_B^*\right] = \mathbf{0} \tag{7.14}$$

Effect on the non-basic variables: $$\left[\Delta\mathbf{X}_{NB}^*\right] = \mathbf{0} \tag{7.15}$$

Effect on the objective function: $$\Delta z = \sum_j \left(\Delta c_j\right)\!\left(x_j\right) \tag{7.16}$$

with Δc_j the change in the original cost of the variable j

If $\Sigma\ r_j > 1.0$, we are not sure whether the basis has or has not changed. We would have to calculate the new optimization case.

7.4.3 Inequality rhs value change: Case 1. All inactive constraints

We can calculate the following metrics, which measure the fraction of the maximum allowable change that has occurred.

If $\Delta b_j > 0$	$r_j = \Delta b_j/AI_j$	with AI the maximum allowable increase w/o a basis change	(7.17)
If $\Delta b_j < 0$	$r_j = -\Delta b_j/AD_j$	with AD the maximum allowable decrease w/o a basis change	

If each of the $r_j \leq 1.0$, the effect of all the changes will not cause a basis change. Therefore, the variables and the objective will be unchanged.

7.4.4 Inequality rhs value change: Case 2. Inactive and active constraints

It this case, we calculate the same metrics as above and apply the following **100% rule**.

If Σ $r_j \le 1.0$, we are sure that the basis has not changed. The final values for **x** can be calculated using equation (7.4) with several Δb_i, and the profit can be calculated using equation (7.5).

Effect on the basic variables:

$$\left[\Delta X_B^*\right] = \left[A_B^*\right]^{-1} \begin{bmatrix} \Delta b_1 \\ \Delta b_i \\ \cdot\cdot \\ \Delta b_m \end{bmatrix} = \begin{bmatrix} \Delta x_{B1} \\ \Delta x_{B2} \\ \cdot\cdot \\ \Delta x_{Bm} \end{bmatrix} \qquad (7.18)$$

{This is a "marginal mechanism" required to stay at the optimum when $\Delta \mathbf{b}_i$ occurs.}

Effect on the non-basic variables: $$\left[\Delta \mathbf{X}_{NB}^*\right] = \mathbf{0} \qquad (7.19)$$

Effect on the objective function: $$\Delta z = \sum_i \left(\Delta b_i\right)\left(c_{Ri}\right) \qquad (7.20)$$

with c_{Ri} the sensitivity of the inequality i

If Σ $r_j > 1.0$, we are not sure whether the basis has or has not changed. We would have to calculate the new optimization case, which might involve pivot operations.

> Recall that we could always resort to making all changes and resubmitting the problem for optimization. Therefore, we can evaluate complex sensitivities not covered by the above methods, although at the cost of increased computation.

7.5 Bounding objective for large changes in the RHS

We would like to determine the sensitivity of the objective function for "large changes" in the inequality constraint bounds. This can be done using the method described above, which requires calculating the results for each corner point as the solution moves from the base case. However, can we determine some sensitivity information without these calculations? The answer is a limited "yes", but since the basis might change, we will have to accept sensitivity information that is not be exact, but provides useful limits.

To understand the concept, we will consider the base case linear programming solution shown in Figure 7.3a. This figure shows the standard sensitivity result for a change in the limit to inequality 1 that increases the range of the feasible region - without a basis change. In contrast, Figure 7.3b shows the same situation, but with a basis change. We can see that the sensitivity of a change in an active inequality that increases the feasible region must be unchanged or decrease from the base case sensitivity. By similar argument, the sensitivity of a change in an active inequality that decreases the feasible region must be unchanged or increase from the base case sensitivity.

For a linear program minimizing the objective, the objective function must always decrease (or remain unchanged) as the size of the feasible region is increased. Also, the objective function must increase (or remain unchanged) as the feasible region is decreased. We can use this principle to develop a useful generalization about the sensitivity analysis of an LP when the basis changes any number of times.

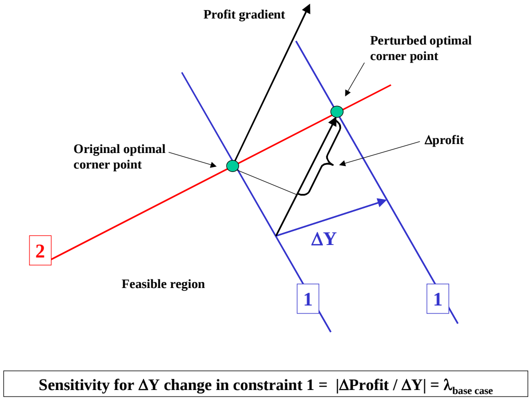

Profit gradient
Perturbed optimal corner point
Original optimal corner point
Δprofit
2
ΔY
Feasible region
1
1
Sensitivity for ΔY change in constraint 1 = |ΔProfit / ΔY| = λ base case

Figure 7.3a. The sensitivity of the objective to a RHS change that increases the feasible region without a basis change.

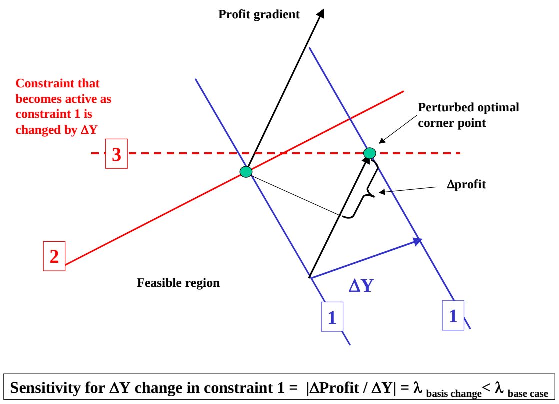

Profit gradient
Constraint that becomes active as constraint 1 is changed by ΔY
3
Perturbed optimal corner point
Δprofit
2
Feasible region
ΔY
1
1
Sensitivity for ΔY change in constraint 1 = |ΔProfit / ΔY| = λ basis change < λ base case

Figure 7.3a. The sensitivity of the objective to a RHS change that increases the feasible region with a basis change.

For a linear program minimizing the objective, the objective function is monotonically decreasing (increasing) as the right hand side of an inequality constraint is changed in the direction of increasing (decreasing) size of the feasible region.

We can use this property to determine whether or not an option being investigated is worth continued evaluation *beyond a change in optimal corner point*, as described in the following.

- **Increasing the feasible region -** When the constraint rhs changes in a direction that increases the size of the feasible region, the objective function must improve or be unchanged. Also, the absolute value of the constraint's marginal value has its largest magnitude at the base case.

Objective is minimized	**Objective is maximized**
OBJ $_{\text{base case+}\Delta\text{rhs}} \geq$ OBJ $_{\text{base case}}$ + (marginal value)(Δrhs)	OBJ $_{\text{base case+}\Delta\text{rhs}} \leq$ OBJ $_{\text{base case}}$ + (marginal value)(Δrhs)

Let's consider an example in which we are maximizing the objective, profit. We can calculate an estimate of the profit using P_{estimate} = Profit $_{\text{base case}}$ + (marginal value)(Δrhs).

- If this profit estimate (P_{estimate}) is less than the acceptable rate of return, we know that the project is not acceptable. We can reject it, because the marginal value would decrease if a basis change occurred within Δrhs. This would lower the profit even further.
- If this profit estimate (P_{estimate}) is above the acceptable rate of return, we are not sure whether the profit with Δrhs is high enough, because the marginal value of the constraint would decrease (perhaps, to zero) if a basis change occurred. Therefore, this problem has to be reoptimized with the right-hand side changed.

- **Decreasing the feasible region -** When the constraint rhs changes in a direction that decreases the feasible region, the objective function must become worse or be unchanged. Also, the absolute value of the constraint's marginal value has its smallest magnitude at the base case.

Objective is minimized	**Objective is maximized**
OBJ $_{\text{base case+}\Delta\text{rhs}} \geq$ OBJ $_{\text{base case}}$ + (marginal value)(Δrhs)	OBJ $_{\text{base case+}\Delta\text{rhs}} \leq$ OBJ $_{\text{base case}}$ + (marginal value)(Δrhs)

Let's consider an example in which we are maximizing the objective, profit. We can calculate an estimate of the profit using $P_{estimate}$ = Profit $_{\text{base case+}\Delta\text{rhs}} \geq$ Profit $_{\text{base case}}$ + (marginal value)(Δrhs).

- If this profit estimate ($P_{estimate}$) is less than the acceptable rate of return, we know that the project is not acceptable. We can reject it, because the magnitude of the marginal value would increase if a basis change occurred within Δrhs. This would lower the profit even further, because Δrhs is negative. In the extreme, the problem could become infeasible, giving an infinite marginal value.
- If this profit estimate ($P_{estimate}$) is above the acceptable rate of return, we are not sure whether the profit with Δrhs is high enough, because the magnitude of the marginal value of the constraint would increase (perhaps, to infinity) if a basis change occurred. Therefore, this problem has to be reoptimized with the right-hand side changed.

8.0 Example Model Formulations for LP Problems

When applying optimization, we are challenged to formulate models that have two properties; sufficient accuracy for meaningful results and sufficient simplicity to be solved within reasonable computing time. This challenge is especially acute when using linear programming, which is limited to very simple models. Not surprisingly, engineers and management scientists have worked for years to formulate models that satisfy both requirements.

No simple recipe exists for linear programming modelling. The engineer must have a toolkit of modelling approaches and use creativity in applying the approaches to each problem. In this section, we will learn a few of the most useful and general model formulations for linear programming. Each approach will be presented with strengths and weaknesses and an example. This will enable the readers to add each formulation to their modelling toolkits.

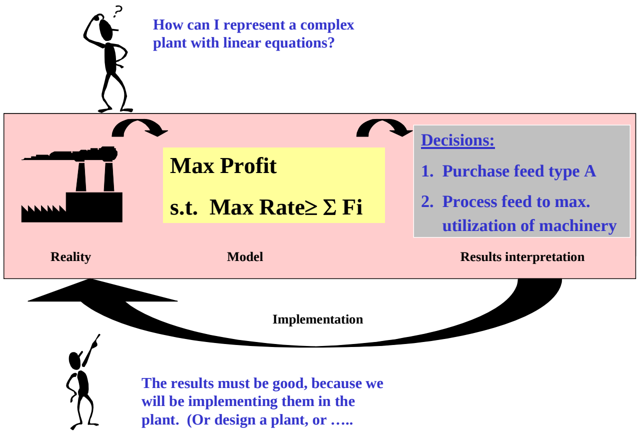

How can I represent a complex plant with linear equations?
Max Profit
s.t. Max Rate≥ Σ Fi
Decisions:
1. Purchase feed type A
2. Process feed to max. utilization of machinery
Reality
Model
Results interpretation
Implementation
The results must be good, because we will be implementing them in the plant. (Or design a plant, or …..

Figure 8.1. Using an LP model to optimize a complex process.

8.1 Straightforward LP Model

The most common LP models represent key balances of conserved properties. While the exact balances are usually nonlinear, the LP formulation simplifies the relationship, so that only the *most important variable* appears. A few examples are given in the following for engineering systems.

Material balance: This is an exact equation, mild assumptions are involved, e.g., no leaking.

$$\mathbf{F_{in}} - \mathbf{F_{out1}} - \mathbf{F_{out2}} = 0 \tag{8.1}$$

Component material balance: This assumes that the separation (separation unit) or the yield (conversion unit) does not change.

$$\alpha_{\mathbf{in}}\mathbf{F_{in}} - \alpha_{\mathbf{out1}}\mathbf{F_{out1}} - \alpha_{\mathbf{out2}}\mathbf{F_{out2}} = 0 \tag{8.2}$$

“Energy” balance: This expresses the energy consumption as a function of the feed rate only. It combines and simplifies several balances.

$$\mathbf{F_{in}} - \gamma\mathbf{F_{steam}} = 0 \tag{8.3}$$

The balances can be on a wide array of entities, for example, ball bearings, workers in a plant, hours available for a piece of equipment. The resulting models are rather crude but can be used to make some important decisions. For example, we will be purchasing feed materials with different properties and prices and with considerable uncertainty in the actual feed material and future market demands. We want to make a good decision, certainly selecting a feed that we can process to make the needed products, but we do not require extreme accuracy because of the uncertainties. Thus, we might select the straightforward model formulation.

8.2 Base-Delta LP Model

The previous model could be thought of as a linearization that is restricted to one (the most important) variable. This approach can be extended to additional variables, which is termed base-delta modelling (Boddington, 1995). The name implies that the model using the most important variable is the "base" model, and the other linear terms provide smaller corrections for “deltas” in other variables. We recognize this as a Taylor’s series retaining only the linear terms.

$$\mathbf{y(x)} = \mathbf{y(x}_0\mathbf{)} + \left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}_1}\right)_{\mathbf{x}_0}\left(\mathbf{x}_1 - \mathbf{x}_{10}\right) + \left(\frac{\partial \mathbf{y}}{\partial \mathbf{x}_2}\right)_{\mathbf{x}_0}\left(\mathbf{x}_2 - \mathbf{x}_{20}\right) + \mathbf{.....} \tag{8.4}$$

We must recognize that the linearization is about a point ($\mathbf{x_0}$), and the accuracy of the solution depends on how well the approximation applies at the solution point, which is not likely the base point ($\mathbf{x_0} \neq \mathbf{x^*}$). To improve the solution accuracy, the deviations in the independent variables ($\mathbf{x}$-$\mathbf{x}_0$) could be limited by lower and upper bounds.

Example 8.1 We will build a straightforward linear programming model for the petroleum reforming reactor shown in Figure 8.2 using the data in Table 8.1 from Boddington (1995). The model is based on the base case operation of reactor severity 850 and feed naphthenes of 15%.

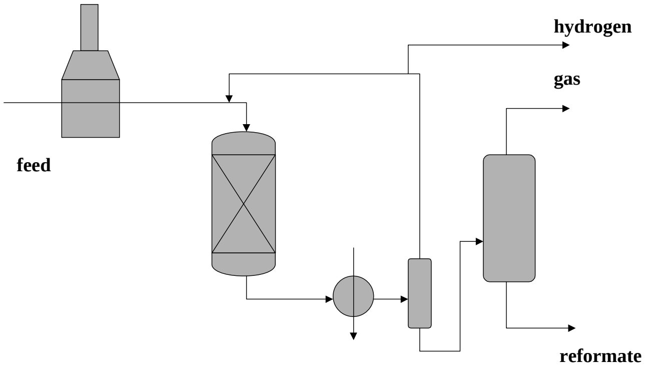


Figure 8.2 Naphtha reformer process

Table 8.1 Data on the Reformer Model Yields Dependence on Severity and Naphthenes

Inputs:		**Base case**			
Reactor severity	800	850	900	800	850
Feed naphthenes (%)	15	15	15	20	20
Product Outputs:					
Reformate (%)	90	86	80	92	88
Gases (%)	9	13	19	7	11
Hydrogen (%)	1	1	1	1	1
Octane (product quality in “octane” units)	80	85	90	81	86

The following model gives the product flow rates (in kg/min) as a function of only the key variable, the mass feed flow rate (in kg/min).

$$F_{reformate} = .86F_{feed}$$

$$F_{gas} = .13F_{feed}$$

$$F_{hydrogen} = .01F_{feed}$$

Example 8.2 We will build a base-delta linear programming model for the petroleum reforming reactor shown in Figure 8.2 using the data in Table 8.1. The model is based on the base case operation of reactor severity 850 and feed naphthenes of 15% and includes linear effects of changes severity and naphthenes.

Based on the data in Table 8.1, the coefficient for the severity-reformate yield is

$$\begin{aligned}\Delta\text{reformate yield}/\Delta\text{severity} &= (80-90)/(900-800) = -0.10\ \%/\text{severity unit}\\ &= -0.0010\ \text{wgt fraction/severity unit}\end{aligned}$$

This delta is applied to the base case feed flow rate, so that the model for the reformate flow rate in kg/min would be the following.

$$F_{reformate} = \lfloor -0.0010(F_{feed})_{ba\sec ase} \rfloor \Delta severity$$

Other delta coefficients are calculated in a similar manner to give the following model for all component flow rates in the product in mass units (kg/min) and for octane in octane units.

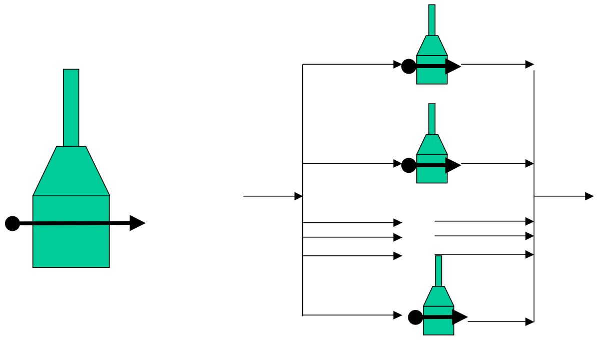
$$\begin{bmatrix} .86 \\ .12 \\ .01 \end{bmatrix} F_{feedba\sec ase} + \begin{bmatrix} .86 \\ .12 \\ .01 \end{bmatrix} \Delta F_{feed} + (F_{feed})_{ba\sec ase} \begin{bmatrix} -0.001 \\ 0.001 \\ 0 \end{bmatrix} \Delta severity + (F_{feed})_{ba\sec ase} \begin{bmatrix} 0.004 \\ -0.004 \\ 0 \end{bmatrix} \Delta napthenes = \begin{bmatrix} F_{reformate} \\ F_{gas} \\ F_{hydrogen} \end{bmatrix}$$

$$86 + (0)F_{feed} + (0.1)\Delta severity + (0.02)\Delta napthenes = octane$$

8.3 Disjunctive Programming in LP Modelling

Linear models can be accurate in a relatively narrow range of conditions. Base-delta modelling extends the models slightly by including additional variables; however, the range remains limited. To achieve a large range, we can prepare separate linear models representing the same sub-system over a range of conditions and apply the appropriate model depending on the conditions at the solution. This approach is termed *disjunctive programming* (Williams, 1999).

The concept of disjunctive programming extends the application of approximate models. We can apply this concept to include substantial changes in the system, for example changes of phase or different piping structures. Here, we will restrict the application to moderate changes in conditions that might not be well represented by base-delta models. Let’s consider the pyrolysis reactor in Figure 8.3 that can operate over a wide range of flows and temperatures. We could model this by pretending that several reactors exist, although only one reactor exists in the plant. The pseudo-reactors are also shown in Figure 8.3. We can model each of the pseudo-reactors as though it operated a different flows and temperatures; the range of conditions to be investigated is spanned by the pseudo-reactors. The total feed flow is distributed among the pseudo-reactors, and the total effluent is the sum of the individual reactor products. The LP optimization allocates the total feed among the pseudo-reactors, which selects the best temperature(s) for the reactor.



Actual plant has one reactor

The disjunctive model has “n” pseudo-reactors at different conditions.

Figure 8.3 Schematic of pyrolysis reactor and a disjunctive representation.

“Straightforward” model	Disjunctive model	
	Reactor operating condition	**Model at this condition**
$F_i = \alpha_i F$ i = 1, 2, ….., m	Reactor 1: T = T_1 …….. Reactor j: T = T_j ……. Reactor n: T = T_n	$F_{i1} = \beta_{i1} F_1$ $F_{ij} = \beta_{ij} F_j$ $F_{in} = \beta_{in} F_n$ $\mathbf{F} = \sum_{\mathbf{j}} \mathbf{F_j}$ feed split $F_{pi} = \sum_j F_{ij}$ effluent mix

with α_i = the “yield” of component “i” at one (nominal) temperature
β_{ij} = the “yield” of component “i” at condition “j”; in this case, T_j
F = the total feed flow rate
F_i = total flow rate of component “i” in the effluent
F_j = flow rate of feed to condition “j”
F_{ij} = flow rate of component “i” in the reactor effluent operated at condition “j”
F_p = product flow
Note: $\sum_i \alpha_i = 1.0$ $\sum_{\mathbf{i}} \beta_{\mathbf{ij}} = 1.0$ and all flows in mass units, kg/min

Clearly, a weakness of this approach is the possibility of allocating the total feed to more than one reactor, because only one actually exists in the plant. The best (and only rigorously correct) approach is to force the solution to use only one of the disjunctive models; this can be achieved using integer programming (Williams, 1999). Often, disjunctive models are solved using only continuous variables, which might result in multiple models being used simultaneously. This approach is reasonable when (1) the implementation can interpolate between models with similar operations and (2) the uncertainties in the problem do not justify further model accuracy.

Example 8.3 We will develop a disjunctive model for the reformer described in Example 8.1 for the levels of severity. The three yield models at three severities (F_{800}, F_{850}, and F_{900}) are determined directly from Table 8.1.

$$\begin{bmatrix} .90 & .86 & .80 \\ .09 & .13 & .19 \\ .01 & .01 & .01 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} F_{800} \\ F_{850} \\ F_{900} \end{bmatrix} = \begin{bmatrix} F_{reformate} \\ F_{gas} \\ F_{hydrogen} \\ F_{feed} \end{bmatrix}$$

We recognize that this model has more flexibility than the real process, which has only one reactor and can operate at only one severity. However, the solution often selects one severity by having only one non-zero severity-flow, which is easily interpreted. Also, if only two contiguous severity-flows are non-zero, the engineer could interpret this result as requiring a severity between the two selected, with the value determined by interpolation.

8.4 Separable Programming

All of the formulations in this section are designed to improve the accuracy by correcting for non-linearities in the standard LP formulation. Separable programming achieves the correction in a manner that is efficient and easily understood. Let's begin with a definition: a separable function can be represented by the sum of individual functions of only one variable each.

$$\mathbf{y}(\mathbf{x}) = \mathbf{y}_1(\mathbf{x}_1) + \mathbf{y}_2(\mathbf{x}_2) + \ldots.. \tag{8.5}$$

with $\mathbf{x}^T = [x_1\ x_2\ \ldots]$

When the model can be separated as shown in equation (8.5), a linear approximation can be built by using piecewise linear approximations for each of the single-variable functions.

An example of piecewise linearization is the efficiency curve for process equipment. For the boiler in Figure 8.4, we seek to model the fuel consumption as a function of the steam production. The fuel consumption depends on other variables, such as the excess oxygen in the flue gas; however, we will consider only the effect of the dominant variable, steam flow rate.

The efficiency is not constant, so that the relationship between the fuel and steam is non-linear.

$$\mathbf{F_{fuel}} = \phi \mathbf{F_{steam}} / \eta \qquad \text{with } \varphi = \text{constant and} \quad \eta = \text{efficiency} \tag{8.6}$$

This efficiency function and the resulting steam-fuel relationship are plotted in Figure 8.4. We can develop an approximate model using *piecewise linearization*, which is also shown in the figure. We can use this to develop multiple, separable models for the fuel flow. The piecewise linear function can be modelled using the following equations.

$$\begin{aligned} F_{fuel} &= \sum_{i=1}^{n} \alpha_i (F_{steam})_i \\ F_{steam} &= \sum_{i=1}^{n} (F_{steam})_i \\ 0 \le (F_{steam})_i &\le (F_i)_{\mathbf{max}} \end{aligned} \tag{8.7}$$

This model separates the steam flow into segments and associates an individual slope between the steam and fuel flows for each segment. This provides the basic model, but another model is required, because if only equations (8.7) were used, the model could use the upper line segment (high steam flow) *first*! Therefore, the model should enforce the order of line segments, as represented in the following.

For any $(F_{steam})_i \neq 0$ and $(F_{steam})_i \neq (F_{steam})_{imax}$, the following must be true

(8.8)

$(F_{steam})_j = (F_{steam})_{jmax}$ for $j < i$ and $(F_{steam})_k = 0$ for $k > i$

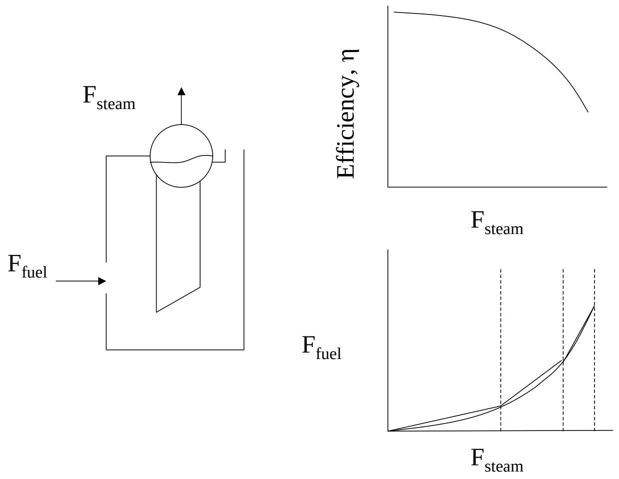


Figure 8.4 Separable programming and piecewise linearization of the boiler efficiency relationship.

Fortunately, equation (8.8) is not needed for an important special case in which the objective function forces the correct selection of variables. As apparent in Figure 8.4, the most efficient segment is the lowest; therefore, the objective of minimizing fuel (or cost) will force the correct selection *in this case*. Again, integer variables would in general be required.

8.5 Linear Transformations in LP Modeling

When developing linearized models, we should always seek linearizing transformations. An important example is blending, which is used in many industries, such as petroleum processing, cement manufacturing, and food processing. Often, properties do not blend linearly.

$$\mathbf{F_B x_B \neq \sum F_i x_i} \qquad (8.9)$$

with the subscript “i” indicating the pure component and “B” indicating the blended material. In some cases, we can develop correlations between pure properties and their contributions to the blended properties.

$$\mathbf{F_B x_B = \sum F_i y_i \quad \text{with } y_i (x_i)} \qquad (8.10)$$

The transformed variable y is often referred to as the “blending index”. It relates the unblended component property to its contribution in the blended material.

8.6 Goal Programming in LP Modelling

We stated in Section 4.2 that an LP always has more variables than equations. This is true for well-posed problems, but it does not occur naturally by formulating fundamental balances. For example, consider the following blending problem, in which two streams are mixed to minimize cost while satisfying four specifications, one total flow F$_B$, and three compositions; w$_B$, x$_B$, and y$_B$.

Conventional linear program with hard constraints

$$\begin{aligned}
&\min_{\mathbf{F_1,F_2}} \quad \mathbf{z = -(F_1 + F_2)} \\
&\mathbf{F_B w_B = F_1 w_1 + F_2 w_2} \\
&\mathbf{F_B x_B = F_1 x_1 + F_2 x_2} \\
&\mathbf{F_B y_B = F_1 y_1 + F_2 y_2} \\
&\mathbf{F_B = F_1 + F_2} \\
&\mathbf{F_1 \geq 0} \\
&\mathbf{F_2 \geq F_{2min}}
\end{aligned} \tag{8.11}$$

with F$_1$ and F$_2$ the only independent variables. We immediately recognize that we cannot satisfy four equalities with only two variables, because we have a problem with fewer variables that equations. The original formulation has no solution; what do we do?

However, we can still use linear programming to find a “good” solution. We define a good solution as one that is either satisfies all constraints, if possible, or is “close” to satisfying all constraints. We can use several definitions of close; here, we will use a definition that conforms to the linear programming assumptions. We add non-negative slack variables to every constraint that we allow to be violated. The new formulation is given in the following, with all expressions allowed violation except the total flow and non-negative flows.

Modified linear program formulation with selected soft constraints using goal programming

$$\begin{aligned}
&\min_{F_1,F_2} \quad z = -(F_1 + F_2) + \boxed{\sum \alpha_i x_{si}} \\
&F_B w_B = F_1 w_1 + F_2 w_2 + \boxed{\begin{matrix} x_{s1} - x_{s2} \\ x_{s3} - x_{s4} \\ x_{s5} - x_{s6} \end{matrix}} \\
&F_B x_B = F_1 x_1 + F_2 x_2 + \\
&F_B y_B = F_1 y_1 + F_2 y_2 + \\
&F_B = F_1 + F_2 \\
&F_1 \geq 0 \quad , \quad F_2 \geq 0 \quad x_{si} \geq 0 \\
&F_2 \geq F_{2\mathbf{min}} - \boxed{x_{s7}}
\end{aligned} \tag{8.12}$$

We see that we need to add one slack variable to an inequality and two slack variables with different signs to an equality constraint, because it could be infeasible in either “direction”. The slack variables must also appear in the objective function with penalties that are large enough so that all slacks will be zero at the solution if a feasible solution is possible. We have included weighting factors, α_i, for every slack variable in the objective function appearing in equation (8.12). This is done for two reasons. First, each slack has different units, so that weighting is required to place similar values on the violations. Second, we can place difference importance on violations of different constraints through a selection of the weightings. This “tuning” is usually necessary because of different effects of violations on economics, product quality and safety. Finally, we note that the solutions to equations (8.11) and (8.12) should be the same when a feasible solution is possible; this can be achieved by selecting large enough values for all weightings.

Goal programming is used widely in mathematical programming. The standard LP formulation, without goal programming, involves *hard constraints*; formulations with goal programming involve *soft constraints*.

Hard Constraints: These are equalities and inequalities that must be strictly observed. Any violation is considered infeasible.
Soft constraints: These are equalities and inequalities that can be violated. The extent of violation is penalized in the objective function, which tends to reduce the violation, to zero if possible.

We note the following considerations when applying goal programming.

- A single model can include both hard and soft constraints.
- The correct penalty function values depend on the user's priorities. Some case studies may be required to determine the correct penalties, α_i.
- The user should always check the values of the slack variables during results interpretation. If one or more is non-zero at the optimum, the user should investigate why and see if the tradeoffs by the LP are appropriate for the current situation.
- The objective function includes the penalties for the infeasibilities, so that it is not easily interpreted when a slack is non-zero. When the objective function value is important, e.g., profit, the value without the penalties should be calculated separately and reported to the user.
- We should never soften a fundamental balance, such as material or energy balances. These must always be strictly observed.
- Softening some constraints helps in debugging models. An infeasible solution for a problem with hard constraints is often difficult to analyze, while a solution with some violations due to goal programming can be more easily interpreted.
- Goal programming should be used when a feasible solution must always be obtained. This is the situation when linear programming is used is used in a closed-loop, real-time application. The goal programming formulation will prevent a disturbance from stopping the LP from finding the "best" solution, even if disturbances occur in the system being controlled that cause infeasibilities in the dependent variables.

8.7 Flow-property blending relations in LP Modelling

Process plants often involve some type of mixing. On first encountering these models, most engineers conclude that the mixing model must be non-linear. However, we will introduce a linear model that can be formulated when certain restrictions are valid. When mixing multiple streams to achieve multiple blended material compositions, we formulate the overall material balance and balances on the properties. The resulting blended composition is usually written in the following form.

$$\mathbf{x_{Bi}} = \frac{\sum_{\mathbf{j}} \mathbf{F_j x_{ij}}}{\sum_{\mathbf{j}} \mathbf{F_j}} \tag{8.13}$$

with i = the i^{th} property
j = the j^{th} flow
F = the flow rates which are variables
x_{ij} = the component properties, which are constants
x_B = the blended property, which is a variable

Equation (8.13) is non-linear and could not be used in a linear program. However, the relationship could be replaced by the following two linear inequalities.

$$\left(\sum_{\mathbf{j}} \mathbf{F_j}\right)\mathbf{x_{B\,max}} \geq \sum_{\mathbf{j}} \mathbf{F_j x_{ij}} \tag{8.14}$$

$$\left(\sum_{\mathbf{j}} \mathbf{F_j}\right)\mathbf{x_{B\,min}} \leq \sum_{\mathbf{j}} \mathbf{F_j x_{ij}} \tag{8.15}$$

We have replaced the variable x_B with its maximum and minimum values in the inequalities, thus making the expressions linear. Using the two inequalities provides the blending relationships that are required. Naturally, if the maximum and minimum bounds are equal, the two expressions enforce the desired blended product property.

We must recognize an important limitation of this LP formulation. The component properties must be known constants. If the mixing model is part of a larger LP model that is used in optimizing the entire plant, the component properties are likely to be variables, because they depend on upstream operations and flows into and output of the component tanks. This situation is called the pooling problem, because the both the flow rates and properties to a component inventory ("pool") are variables. The pooling problem is inherently non-linear, and we must employ a non-linear optimization method (Reklaitis, et. al., 1983).

8.8 Absolute value

Often, the optimization goal is to achieve performance close to a specification. We can use the absolute value of variables from their desired values as a measure of approach to the best performance. However, the absolute value is a non-linear function and cannot be used in a linear program. We can model the system using penalty variables that are non-zero in proportion to the absolute value of the deviation. These penalty variables are given a large positive cost to prevent them from having non-zero values unless needed.

In a chemical reactor operations example, we want to achieve a desired product flow rate, if possible. We could use the absolute value from the desired value as the objective to be minimized.

$$\begin{aligned}
&\min_{x,x_s} \sum_j x_{sj} \\
&s.t. \\
&F_i = \alpha_i F \\
&F_i = F_{desired} + x_{s1} - x_{s2} \\
&F \geq 0 \quad , \quad x_{sj} \geq 0
\end{aligned} \tag{8.16}$$

8.9 Mini-Max Problem

When we consider multiple outcomes of an optimization problem, we have flexibility in formulating the objective function. Two common examples are given for a minimization problem in the following.

- **Min-Min** – In this case we desire the best outcome to be as low as possible.
- **Min-Max** – In this case, we want the worst outcome (maximum of the possible outcomes) to be minimized.

We will consider the min-max problem in this subsection, and we recognize the max-min problem is equivalent, because we can covert between the two by multiplying the objective by (-1). This min-max strategy is often used when dealing with uncertainty. For example, we might require that a plant be able to manufacture a minimum amount of product for a range of possible feed material properties.

The approach provides a model for every possible outcome being considered. This provides a calculation for the performance for every outcome. We require the decision (optimization) variables to have the same value for every outcome, because we do not know in advance which outcome will occur. Then, we "select" the maximum as the objective function. Since a selection would be non-linear, a special formulation using inequality constraints, given in the following, is used to have the same effect.

$$\begin{aligned}
&\min_{x} z \\
&s.t. \\
&z \geq f_1(\alpha_1, x) \\
&z \geq f_2(\alpha_2, x) \\
&\ldots\ldots \\
&z \geq f_k(\alpha_k, x)
\end{aligned} \tag{8.17}$$

with f_i = a set of linear equations and inequalities with the parameters α_i yielding an objective function f_i. This is the "entire problem" for one set of parameters, α_i.

α_i = The parameters associated with outcome i

x = The optimization variables, which are used in every model i. Note that the same values are used for every outcome i, so that we find the values of x that satisfy all constraints for all parameters in the samples α_i.

8.10 Minimum-Proportional variable

Often, a variable is proportional to a decision variable, such as the production of one product is proportional to feed material. Other variables are proportional over a range of the decision variable, but the variable must never decrease below a minimum limit. An example is shown in Figure 8.5, which shows a compressor. Normally, the flow through a compressor is proportional to the feed flow to the compressor. However, the flow through the compressor must never be below a minimum or unstable flow will damage the machine. This is called the surge limit. To protect the machine, a recycle is provided and a flow controller achieves the minimum flow by recycling when required.

We can model this system by including an inequality ensuring that the flow through the compressor is greater than or equal to the minimum. We relate the feed flow to the flow through the compressor using a slack variable to represent the recycle flow rate. The cost of compression ensures that the recycle flow rate is zero unless required to maintain feasibility. A summary of the model is given in the following equations.

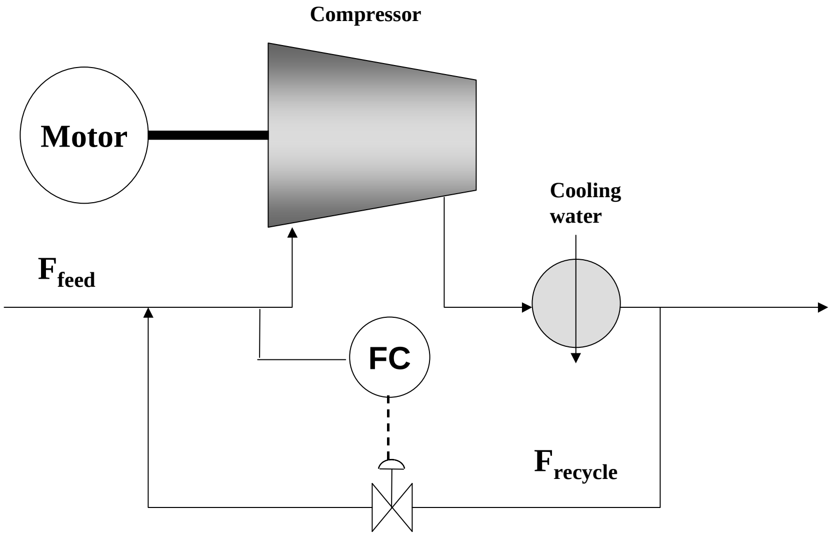


Figure 8.5. Compressor with recycle.

$$
\begin{aligned}
&\min_{F_i} z = -\mathbf{Profit} = (\textit{other terms}) + W(c_{energy})\\
&s.t.\\
&F_{comp} = F_{Feed} + F_{\mathbf{Recycle}}\\
&F_{comp} \geq F_{surge}\\
&W = \alpha \;\; F_{comp}\\
&F_i \geq 0
\end{aligned}
\qquad (8.18)
$$

8.11 General modelling guidelines

A few additional guidelines are presented in this section because they apply to all model structures.

- **Bound variables -** Linear programming models have acceptable accuracy over a limited range of variables. The people who initially build the model usually have the greatest insight into the appropriate range. Therefore, they should bound variables to constraint the results to a meaningful range.

- **Constraint redundancy** - For some problems, a subset of the constraints can be removed without affecting the feasible region of the objective. These constraints are redundant and should be removed. A few examples are presented here.

  - All component material balances and the total material balance for the same stream are included in the model. One of these equations is redundant and should be removed.
  - Some limitation (sales, equipment capacity, etc.) will never affect the solution.

  However, constraints should not be removed if their activity depends on user-input parameters, and these parameters could change. For example, an equipment capacity could change due to lower efficiency; therefore, the constraint must be retained in the model.

- **Variable elimination/retention** - When solving a set of linear equations, we can analytically eliminate some variables without changing the solution. However, the person formulating the model is cautioned that this approach is not generally appropriate for a linear programming model. The key difference is the bounds on variables; if a variable is removed, it cannot be bounded. A variable can only be eliminated analytically from the model if it is never bounded.

- **Use of equalities to replace inequalities** - Often, modelers try to "guide" the linear programming by forcing some inequalities to be active, by changing these to equalities. For example, we might think that the optimum occurs when the production rate is equal to 500 m^3/h, which is the maximum for this variable. It is a poor practice to replace $F_{Production} \leq 500$ with $F_{Production} = 500$. While this inequality constraint might be active for many scenarios, it could be inactive for other situations.

9.0 Presenting Optimization Results

Typically, the analyst who performs the detailed technical work in formulating, solving, and checking the optimization results is not the (sole) person who decides on actions. Therefore, the technical specialist must report the results to others who are competent in their tasks but are not expert in optimization. For these people, a "solution" consisting solely of numerical values for the optimization variables is not adequate; even the complete computer output is inappropriate for people who do not have in-depth knowledge of the model and linear programming methods. Guidance for reporting optimization results is provided in this section.

9.1 Explaining the formulation

Linear programming, even with the clever formulations described in Section 8, usually involves significant model simplification, i.e., the predictions of all dependent variables (flows, temperatures, compositions, and so forth) can deviate significantly from predictions using non-linear models and from real system behavior. However, the results of the linear programming study can be essentially correct when the modelling errors do not significantly influence the key optimization variables.

The report should convey the structure of the linear programming model and the simplifications made to achieve this linear programming model. Depending on the reader's understanding of the technology, terms such as "base-delta" and "disjunctive models" could be used.

- **Fundamental Balances -** The fundamental aspects of the model should be described. Recall that while material balance is fundamental, some approximation is made in the selection of components modelled. Be sure to explain such assumptions and simplifications.
- **Constitutive Models -** These are models whose structures are based on basic physics and chemistry, but they are not exact and have parameters with a limited range of applicability.
- **Correlation models -** The simple model structures in linear programming result in many correlation models that are developed from empirical data or from a more detailed model.

However, a description of the model structure is not adequate; an explanation is required of the quantitative difference between the linear programming model and the expected real behavior, which can be achieved using one of the following methods.

- **Error bounds -** Define maximum errors of predictions in key variables, e.g., the modelled yield of product is within ±3% of the actual reactor behavior. These bounds cannot be used directly to evaluate the optimization results, but the information can be used in the results analysis, as discussed in the next section.
- **Variable bounds -** Define a range of optimization variables over which the optimization will (likely) be reliable.
- **Goal penalties -** If goal programming is used to achieve a specific objective this should be stated along with an indication of how strong the goal penalties are. For example, are the penalties high enough to prevent all occurrences (for example, of mathematical infeasibility) or can the some deviation from the goal occur and still remain feasible?
- **Limits of solution -** Most problems have a range over which the results are "acceptable". Beyond this range of parameters, the solution becomes unacceptable, with the meaning of unacceptable depending on the specific problem. Often, the acceptability depends upon key factors such as safety, product quality, or profit. The boundaries of acceptable performance should be defined and the method for enforcing the limits indicated.

Its is important to recognize that an assurance that the correct optimization result has been obtained in a complex task and cannot generally be evaluated unless the specific problem is presented and solved. Therefore, the description of the effects of model error in the bullet items above will be approximate.

9.2 Explaining sensitivity analysis

The typical audience for sensitivity analysis is not interested in technical issues, such as when the basis (corner point) changes or the meaning of degeneracy. However, the audience is aware of the uncertainty of the model and needs to understand the impact of the uncertainty - *in no uncertain terms*. Therefore, the author of the study has the responsibility of reporting the results using commonly understood terms and in a manner that explains the effects on the key decisions.

There is no simple recipe for deciding what is "important". However, the engineers performing a study will certainly understand the problem; model, key decisions, and parameter uncertainties. Certainly, the typical output from a computer program is inadequate for presentation to a person who does not know the model formulation extremely well. These results should be placed into context of the problem. The following issues should be addressed in the report.

- **Units -** Be sure to include units for all sensitivities. The units are Δ(objective)/Δ(parameter). Thus, a "small" sensitivity value in the computer report could really be very large if the units of the objective function are 10^6 kg of production.

- **Parameter Perturbations -** The types of perturbations must be clearly stated. For example, many of the analysis results are for "one at a time" changes to a parameter. If these are presented without clear guidance, the reader will likely assume that the results from multiple parameter changes can be determined as a linear combination of the changes for each parameter. This is not generally correct and could lead to serious errors.

  The results analysis might consider a change of several parameters because the parameters are correlated in the real problem. For example, the yields of many reactor components could change is a related manner due to a feed composition impurity. Again, this should be explained clearly.

- **Parameter range -** The parameter ranges for every sensitivity should be reported. This should be clearly documented, such as in a table with the sensitivity results.

- **Alternative solutions -** If the system has alternative solutions (or solutions with very close objective values) this situation should be reported. In addition, the reason for selecting the recommended solution should be clearly explained.

- **Degeneracies -** Constraint degeneracies should be carefully explained to avoid an inappropriate decision. For example, investing in capital equipment to expand the bound of a redundant active inequality constraint might not increase the feasible region or improve the objective function. Therefore, an explanation could be provided for lowest cost method for achieving a specific increase in performance, e.g., profit or production rate. Because of the degeneracy, several constraints might have to be changed concurrently.

- **Active set and implementation strategy -** The active set of constraints should be reported and its relevance to the problem explained. Often, we want to achieve this active set in practice, but implementing the values from the linear programming solution will not result in exactly the corner point calculated because of model errors. The implementation can result in violations or in values in the interior of the feasible region. The report should indicate the appropriate strategy for implementing the results, i.e., for adjusting the true system and achieve a "nearly optimal" result. For small adjustments that do not involve a basis change, the basis inverse provides information on how to adjust the optimization (basic) variables in response to changes in the constraint values.

- **Problem specific -** Every problem has its own typical sensitivity questions. Often, these give guidance on how the performance can be improved. One question might be, "How large a decrease in the price of feed material A is required to make its selection attractive?" The reader can decide whether this value could be achieved through aggressive negotiations. A second example is, "How much would it be worth to raise the reactor temperature by 2 °C?" If the potential improvement is substantial, the reader could reevaluate the limit based on long term coke deposition on the catalyst.

- **Infeasibility and unboundness -** The failure to find a basic feasible solution is always reported by linear programming software, but it is not always prominently displayed. As a result, the user could apply the reported variable values without recognizing the solution failure. Thus, reports developed automatically from the computer (without personal intervention and evaluation) should display ""Optimal Solution Found" or an error message in a manner that will be seen by every user.

9.3 Results analysis presentation

The calculations for many of these standard questions should be part of the optimization analysis. Some of the questions might occur only in special circumstances, such as when prices are very volatile or when plant equipment behavior changes in an atypical manner. Naturally, these special questions must be answered as they occur.

Tabular presentation of sensitivity results provides high accuracy (several significant figures) and can accommodate a large number of parameters. Naturally, the ranges for each sensitivity must be included in the report. These will enable the reader to perform sensitivity analysis not defined when the report is written. On the other hand, these values do not "speak for themselves" and do not replace the thoughtful engineering analysis discussed in sub-section 9.2.

An especially clear presentation of sensitivity results plots the effect of a single parameter on the profit and key variables. An example of this plot is given for the blending process in Figure 9.1. The sensitivity plot is given in Figure 9.2, with explanatory comments in Table 9.1. The plot can extend through many changes in corner point (basis), which requires several optimization runs when generating the data for the plot.

For a linear program, the profit and variables are piece-wise linear, with the change in corner point clearly identified when the slope changes. Each change in corner point should be explained in "engineering terms". Some observations from this example include

- Low ranges of the production rate are not feasible. This is due to the minimum flow rate of butane, which has a high vapor pressure.
- Variables can become non-basic (reach their maximum or minimum) and return to the basis as the production rate changes.
- The profit decreases from its maximum when the production rate is increased beyond point 5.

We conclude that this display is extremely easy to understand, and it facilitates the use of optimization results by non-specialists who cannot screen many pages of numerical optimization results in tabular form.

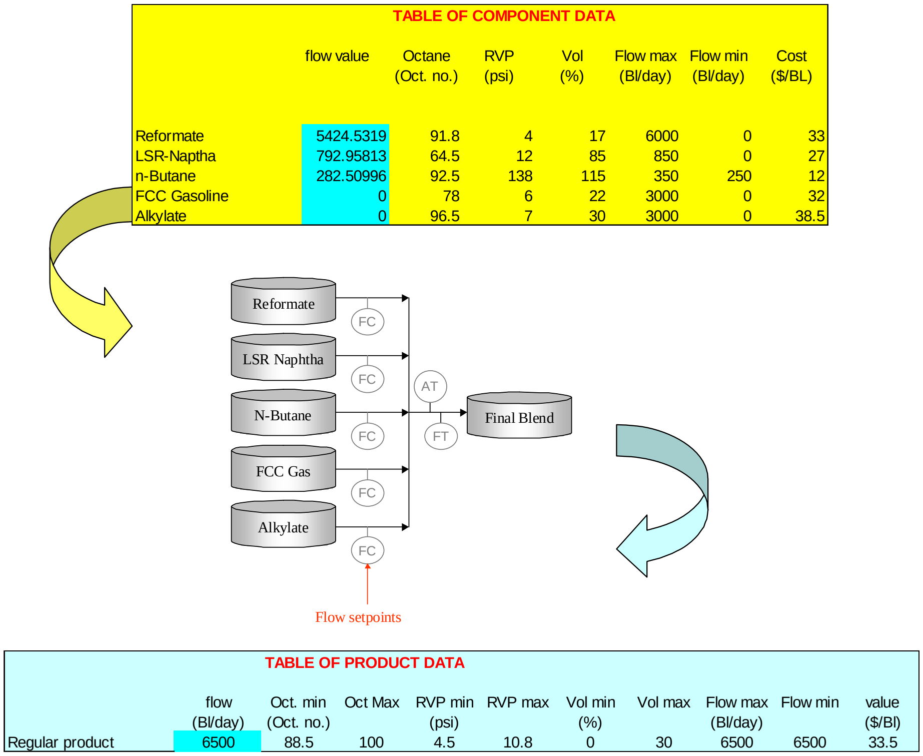


TABLE OF COMPONENT DATA

	flow value	Octane (Oct. no.)	RVP (psi)	Vol (%)	Flow max (Bl/day)	Flow min (Bl/day)	Cost ($/BL)
Reformate	5424.5319	91.8	4	17	6000	0	33
LSR-Naptha	792.95813	64.5	12	85	850	0	27
n-Butane	282.50996	92.5	138	115	350	250	12
FCC Gasoline	0	78	6	22	3000	0	32
Alkylate	0	96.5	7	30	3000	0	38.5

TABLE OF PRODUCT DATA

	flow (Bl/day)	Oct. min (Oct. no.)	Oct Max	RVP min (psi)	RVP max	Vol min (%)	Vol max	Flow max (Bl/day)	Flow min	value ($/Bl)
Regular product	6500	88.5	100	4.5	10.8	0	30	6500	6500	33.5

Figure 9.1 Gasoline Blending problem with base case results

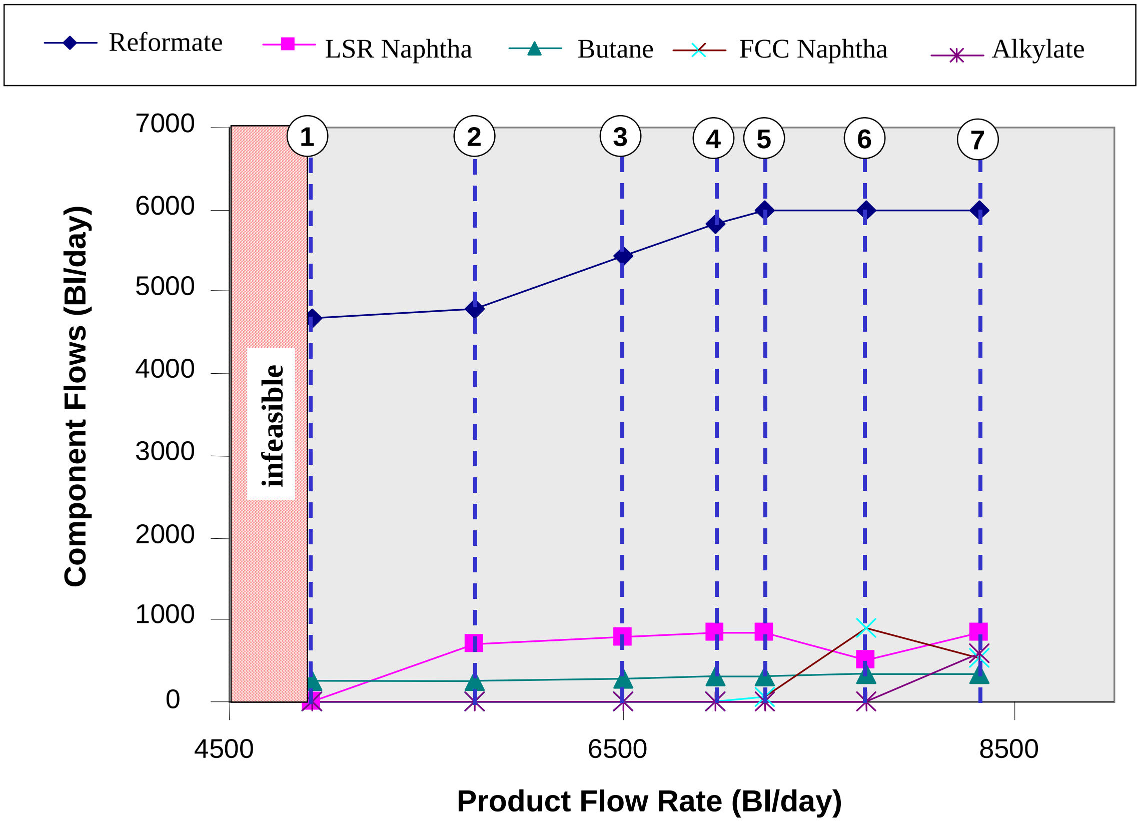


Figure 9.2 Sensitivity plot for a gasoline blending problem.

Table 9.1 Explanation of the corner points designated by circled numbers in Figure 9.2.

Corner point (basis)	Profit ($/day)	Comments
1	7710	The Butane, LSR, FCC naphtha and Alkylate are at their minimum flow rates; only Reformate can be adjusted. Any reduction beyond this point results in infeasibility due to high RVP.
2	12325	Butane flow is reduced to its minimum of 250. The minimum octane bound cannot be achieved, but the qualities remain feasible.
Base Case: 3	13940	Optimum operation and profit for the base case problem defined in Figure 9.1.
4	14924	The LSR flow rate reaches its maximum.
5	15394	Reformate flow reaches its maximum, and FCC gasoline is increased. LSR is reduced.
6	15061	Butane flow reaches its maximum. Alkylate is introduced as a blending component.
7	13901	LSR again reaches its maximum. RVP constraint no longer active.

10. References

Boddington, C.E. (1995) *Planning, Scheduling, and Control Integration in the Process Industries*, McGraw-Hill, New York.

Bradley, S., A. Hax, and T. Magnanti, *Applied Mathematical Programming*, Addison-Wesley, Reading, Massachusetts, 1977.

Chapra, S. and R. Canale (1998) *Numerical Methods for Engineers*, 3rd Edition, McGraw-Hill, New York.

Edgar, T., D. Himmelblau, and L. Lasdon, (2001) *Optimization of Chemical Processes*, 2nd Edition, McGraw-Hill, New York.

Gary, J. and G. Handwerk (1984) Petroleum Refining, Technology and Economics, 2nd Edition, Marcel Dekker, New York.

Geffrion, A. (1976) The Purpose of Mathematical Programming is Insight, Not Numbers, *Interfaces*, 7 (1), 81-92.

Geffrion, A. and Van Roy, T. (1979) Caution: Common Sense Planning Methods can be Hazardous to Your Corporate Health, *Sloan Management Review*, 31-42, Summer, 1979.

Grossmann, I., (1991) *Chemical Engineering Optimization Models with GAMS*, CACHE Design Case Studies Series, No. 6, October 1991.

Hillier, F. and G. Lieberman, (2001) *Introduction to Operations Research*, 7th Edition, McGraw-Hill, New York.

Koltai, T. and T. Terlaky, The Difference Between the Managerial and Mathematical Interpretation of Sensitivity Analysis Results in Linear Programming, *Int. J. Production Economics*, 65, 257-274, 2000.

Rao, S. (1996) *Engineering Optimization, Theory and Practice*, 3rd Edition, Wiley-Interscience, New York.

Reklaitis, G.V., A. Ravindran, and K.M. Ragsdell, (1983) *Engineering Optimization: Methods and Applications*, Wiley, New York.

Rubin, D. and H. Wagner (1990) Shadow Prices: Tips and Traps for Managers and Instructors, *Interfaces*, 20, Juy-August, 1990, pp. 150-157.

Rugarcia, A., R. Felder, D. Woods, and J. Stice, (2000) The Future of Engineering Education: Part 1, *Chemical Engineering Education*, 34, 16-25.

Sen, S. and Hingle, J., (1999) An Introductory Tutorial on Stochastic Linear Programming Models, Interfaces, 29, 2, 33-61

Shamir, R., The Efficiency of the Simplex Method: A Survey, *Man Sci*, 33, 3, 301-333 (1987).

Shu, W.R., L.L. Ross, and K. H. Pang (1979) Naphtha Pyrolysis Model Proves Out over Wide Range of Feedstocks, *Oil & Gas Journal*, Sept 3, pp. 72-79.

Williams, H.P. (1999) *Model Building in Mathematical Programming*, 4th Edition, Wiley, Chichester, UK.

Winston, W., (1994) *Operations Research*, *Applications and Algorithms*, 3rd Edition, Duxbury Press, Belmont, California.

11.0 Study Questions

The following questions are provided to help you review and study linear programming concepts.

Section 1.0 The Importance of Linear Programming

1.1 Discuss when heuristic solution methods are appropriate. Hint: after thinking about this question, read the article by Geffrion and Van Roy (1979).

1.2 Review recent volumes of technical journals such as *Informs, Int. Journal of Production Engineering, Management Science,* and *Interfaces*. Find an article describing an application of linear programming and write a summary. You should discuss the model and its accuracy, the advantages over heuristic approaches, and the benefits described in the article.

Section 2.0 Key Modelling Assumptions and Limitations

2.1 Answer the questions posed in Figure Q2.1.

2.2 Give examples of situations in which additivity is not valid.

2.3 Formulate models containing functions that do and do not satisfy linearity.

2.4 Discuss engineering problems that involve discrete variables. For each, decide whether we can justifiably assume that variables are continuous, and round off the final answer to the nearest integer.

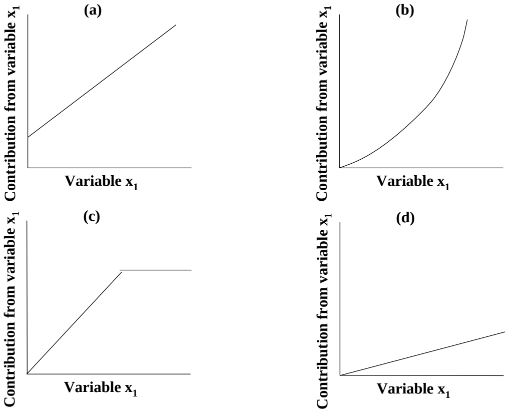


Figure Q2.1. Which of the functions above satisfy the linearity criteria?

2.5 Discuss examples of models that have (a) no uncertainty, (b) negligible uncertainty, and (c) substantial uncertainty.

2.6 Does the sensitivity to parameter errors depend on the problem? Consider these two sets of linear equations.

Problem I: $$\begin{bmatrix} 1 & 2 \\ 3 & \alpha \end{bmatrix} \begin{bmatrix} x_1 \\ x_1 \end{bmatrix} = \begin{bmatrix} 5 \\ 6 \end{bmatrix}$$

Problem II: $$\begin{bmatrix} 1 & 1.999 \\ 2 & \alpha \end{bmatrix} \begin{bmatrix} x_1 \\ x_1 \end{bmatrix} = \begin{bmatrix} 4.9955 \\ 10.000 \end{bmatrix}$$

Solve both equations for a (a) base case with $\alpha = 4.00$ and (b) perturbed case with $\alpha = 4.01$. In which case do the values of the variables x change more? Discuss your results.

Section 3.0 Linear Programming Properties and Advantages

3.1 Discuss some examples of inequality constraints in engineering and business models. Give the variables used to model each and explain why it is a greater than or less than inequality.

3.2 In Figure 3.2, identify the following.

a. Feasible points,
b. infeasible points,
c. feasible corner points,
d. infeasible corner point,
e. active inequality at the optimum, and
f. inactive inequality at the optimum.

3.3 Sketch feasible regions that are convex and others that are convex.

3.4 Can a linear program have the same solution for minimizing and maximizing the same objective function? Explain your answer.

3.5 Can a convex region have a "hole" that is entirely enclosed within the region?

4.0 Principles of Solving a Linear Programming Problem

4.1 Does every set of linear equations have a solution? Does every set have a non-trivial solution?

4.2 Solve the following set of linear equations. To reduce hand calculations, the recommended approach is to use "The Equator" at http://www.ifors.ms.unimelb.edu.au/tutorial/ .

$$\begin{aligned} 0.50x_1 + 0.25x_2 + 0.0x_3 &= 4 \\ 1.0x_1 + 3.0x_2 - 1.0x_3 &= 20 \\ 1.0x_1 + 1.0x_2 + 0.0x_3 &= 10 \end{aligned}$$

This equation set occurs when a specific basis is selected for Example 5.1.

4.3 Define the following terms; point, feasible (infeasible) point, line segment, convex set, and corner point.

4.4 What is a basis? Is any square sub-matrix **A'** in **Ax = b** a basis?

4.5 What are elementary row operations? How do they affect the solution to a set of linear equations?

4.6 Answer the following short questions

a. Summarize the important elements of the LP problem formulation.
b. What is the LP standard form? Why is this the starting point for the solution method?
c. What is canonical form? Why is this an important step?
d. Where is the solution to an LP problem located?
e. What was the purpose of slack variables?

4.7 Answer these questions regarding linear sets of equations.

a. Discuss why a set of equations might not be linearly independent.
b. How can we test whether equations are independent?
c. What is the rank of a matrix?
d. Could we inadvertently formulate a set of equations for a real system that were not independent?

4.8 Describe ill-conditioning of a set of linear equations. Sketch a set of two equations that are (a) well conditioned and (b) ill-conditioned.

4.9 What is the value of a slack variable when the left and right hand sides are equal? What is the coefficient for a slack variables when added to a "greater than" inequality and to a "less than" inequality?

4.10 Computers are very fast. If each evaluation required one second, how long would it take to determine the optimum by the exhaustive search method described in the paragraph above for the problem with m=10 and n=20. Recall that this is a *very small* LP problem, some commercial problems have 100,000 variables or more.

5.0 The Linear Programming Simplex Algorithm

5.1. Locate all basic solutions, basic feasible solutions and optimum solutions for the graphical system in Figure 4.2.

5.2 Which of the following are true for the pivoting operations?

a. A non-basic variable enters the basis.
b. A basic variable enters the basis.
c. The basis changes from square to non-square.
d. After pivoting, all basic variables have values greater than 0.0.

5.3 The LP solution must never contain a non-zero slack variable (T/F).

5.4 What was the purpose of artificial variables?

5.5 Which of the following can be a basic variable in the solution; problem, slack, and artificial variable?

5.6 By referring to a text or reference book, learn the "two-phase" simplex method. Compare Phase I of this method with the "Big-M" method for finding an initial basic solution.

5.7 The simplex algorithm uses local information about adjacent corner points, yet it finds a global optimum. Discuss why this powerful result is achieved, i.e., what about the problem formulation enables this?

5.8 The solution to an LP gives

a. A unique value for the local optimum objective value
b. A unique value for the global optimum objective value
c. Unique values for the problem variables at the global optimum.

5.9 Draw Example 5.1 graphically and confirm the tableau solution by graphical analysis.

5.10 You want to explain linear programming to a high school class. Design a physical system that behaves like a linear program and demonstrates the principles visually. Could you build it with cardboard and tape?

5,11 Formulate and solve the problem in Example 1.2 as a linear program. To reduce hand calculations, the recommended approach is to use the interactive tableau available "The Simplex Place" at http://www.ifors.ms.unimelb.edu.au/tutorial/, the IFORS site. (Hand calculations for the Tableau method are tedious and do not enhance your understanding.)

6.0 Extensions and Special Cases

6.1 How can you detect each of the following; no feasible region, unbounded solution, and multiple optimal solutions, constraint redundancy at the solution, and constraint degeneracy at the solution?

6.2 For the system in Figure 6.4, determine whether

a. the optimum would change if constraint 1 were removed from the problem.
b. the optimum would change if constraint 2 were removed from the problem.
c. the optimum would change if constraint 3 were removed from the problem.

6.3 Discuss the sensitivity for a change in the rhs of constraint 1 in Figure 6.4 for (a) a small increase and (b) a small decrease.

6.4 Formulate and solve the problem in Example 1.2 as a linear program. To reduce hand calculations, the recommended approach is to EXCEL or GAMS. Analyse the solution completely for all possible "weird events".

6.5 Explain the procedure that you would use to restart a linear programming solution after you have changed the values of selected parameters. You will describe how you would use the last tableau in the original solution as your starting point to reduce computations.

6.6 Discuss whether you think that constraint redundancy is likely to occur in manufacturing systems.

7.0 Sensitivity and Range Analysis of LP Solutions

7.2 Perform a graphical sensitivity and range analysis for a different constraint in the system in Figure 7.1.

7.3. Add an inactive inequality constraint to Figure 7.1. Show how much change in its rhs can occur without a change to the optimum.

7.4. The analysis above shows that the reduced cost must *change* to zero for the basis to change. Prove that this equivalent to the original cost *changing* by exactly the same amount.

8.0 Example Model Formulations for LP Problems

8.1 An example application of this formulation is the plant-planning problem from Reklaitis et al. (1983). We are presented with a problem of selecting the quantities of feeds to purchase to maximize profit in a petroleum refinery. A sketch of the system is given in Figure Q8.1, and the data are presented in Table Q8.1.

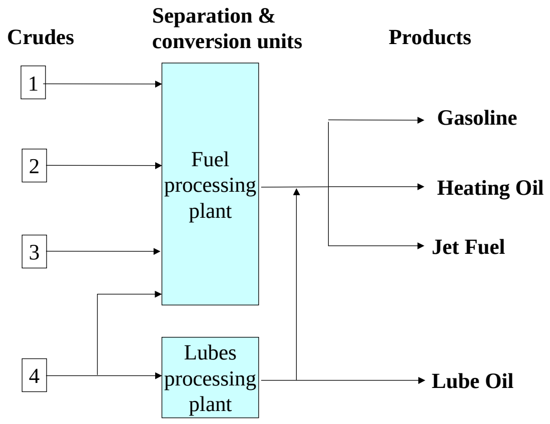


Figure Q8.1 Schematic of a petroleum processing refinery in Question 8.1.

Table Q8.1 Data for Question 8.1 (Bl=barrel)

Product & Crude Names	Product yields (Bl product/Bl crude)					Product Values ($/Bl)	Max. sales (kBl)
	Fuel Processes				Lube		
	1	2	3	4	4		
Gasoline	0.60	0.50	0.30	0.40	0.40	45.00	170
Heating Oil	0.20	0.20	0.30	0.30	0.10	30.00	85
Jet fuel	0.10	0.20	0.30	0.20	0.20	15.00	85
Lube oil	0.0	0.0	0.0	0.0	0.20	60.00	20
Operating losses	0.1	0.10	0.1	0.10	0.10	---	---
Crude cost ($/Bl)	15.00	15.00	15.00	25.00	25.00		
Operating cost ($/Bl)	5.00	8.50	7.50	3.00	2.50		
Maximum availability (kBl)	100	100	100	200			

Operating losses are for the crude and by-products used as fuel in the plant.
Operating cost includes variable costs for fuel, catalysts, etc.

8.2 Formulate the model equations for an LP solution of Question 8.1; your answer should be of the form of equation (4.3). Explain the basis for the model, specifically what is the basis for the balances used?

8.3 Discuss the approximations in the model in Question 8.1.

8.4 Solve the LP problem that you have formulated in Question 8.1. Discuss the solution for validity.

a. Does a feasible solution exist?
b. Is the optimum unique? Is it local or global?

c. How many inequality constraints are active at the optimum including variable upper and lower bounds)?

Answer the following additional questions.

d. If the price of Crude 2 increases to \$15.55, does the active set (corner point) at the optimum change?
e. What is the affect on the active set and optimum of reducing the maximum sales demand of the jet fuel from 85 to 75 kBL?
f. We have found that we have only 75 kBl of Crude 1 available. What is the effect on the optimum?
g. We find that we can sell 179 kBl of gasoline. If we optimize with this new value for the rhs: (i) will the basis change, (ii) will the objective function change, and (iii) will the optima values of the component flows change?
h. One of our competitor's plants has shutdown. As a result, we can sell up to 100 kBl of jet fuel. If we optimize with this new value for the rhs: (i) will the basis change, (ii) will the objective function change, and (iii) will the optima values of the component flows change?

8.5 Answer the following questions about the system in Question 8.1.

a. We are not sure if we are making a profit by producing lube oil. Do you recommend that we continue operating this part of the plant?
b. By minor changes in the operating conditions in the plant, we can change the yields for crude 3 to be [0.3, 0.3±δ, 0.3±δ, 0.0, 0.10], with -.050 ≤ δ ≤ +0.050. (Naturally, the yields must sum to 1.0, so that the changes in heating and jet must be equal in magnitude and opposite in sign.) What is the best value of δ, and what is the potential economic benefit for modifying the operating conditions?

8.6 The problem in Question 8.1 did not consider the time-value of money. Discuss this validity of this assumption.

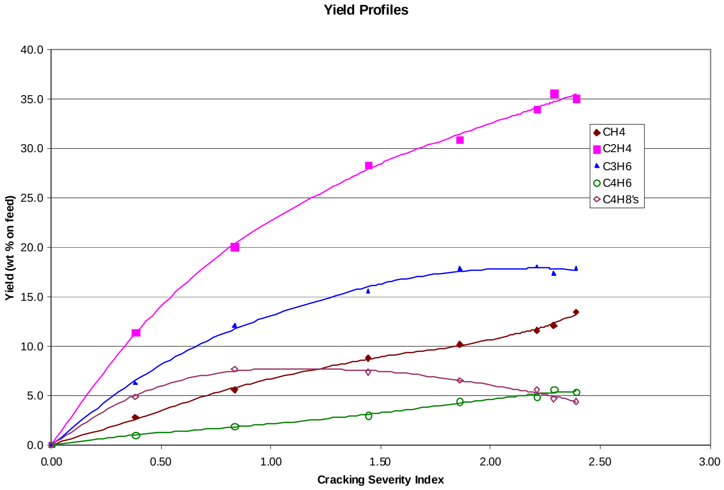


Figure Q8.7 Yields from the pyrolysis of n-heptane. The difference between the sum of the yields and 1.0 is equal to the unconverted n-heptane

8.7 The yields of products from the pyrolysis of n-heptane in a tubular reactor are given by the data in Figure Q8.7 (from data in Shu et al, 1979). (Severity is related to conversion.). Develop a "straightforward" model that predicts the product flow rates of all components from the reactor. The key variable is n-heptane feed flow rate. The nominal severity is 1.75.

8.8 Enhance the model developed in Question 8.7 by adding a delta due to changes in severity. Recommend the allowable range of the ± sizes of the delta in severity, which do not have to be symmetric.

8.9 Repeat the tasks in Questions 8.7 and 8.8 about the nominal severity of 1.0, and discuss your results.

8.10 The heptane pyrolysis reactor in Question 8.7 is to be optimized over a large range of operating conditions. Develop a disjunctive model for the product component flow rates for severities from 0.40 to 2.40.

8.11 Could a base-delta model give a reasonable representation of the component flows for the entire range considered in Question 8.10?

8.12 Boiler efficiency can be modelled according to the following equation.

$$\eta = 0.9127 - 6.6\times 10^{-5}\,\mathbf{L} - 7.1\times 10^{-7}\,\mathbf{L}^2$$

with η = efficiency as a fraction
L = Steam "load" (production) in kton/h (range of 0 to 300)

Develop a piecewise linear function that could be used to optimize the boiler operation to minimize fuel consumption.

8.13 Sketch a boiler efficiency curve that would require discrete variables to ensure that the variable selection in equation (8.8) would be enforced.

8.14 A model contains the equation {$y = ax_1 + bx_2 + x_1x_2$}. Develop a separable representation for this equation.

8.15 By referring to tables in Gary and Handwerk (1984), determine the octane and Reid vapor pressure (RVP) blending indices for the following components.

- Light straight run naphtha
- Reformed naphtha
- n-butane
- i-butane

8.17 A gasoline blending problem is defined in Figure Q8.17. All properties, prices, and constraints are given. (Note that the initial flow values are very small, which are not near the solution. Also, these flows give infeasible product properties.)

a. Formulate the blending problem as a linear program.
b. Program the problem in Excel or GAMS and solve the base case.

1. Does a feasible solution exist?
2. How many constraints are active at the optimum?
3. Do multiple optima exist?

c. Answer the following sensitivity questions.

1. What is the value of the slack variable on the maximum octane constraint? What are its units? How far is octane from its maximum value?
2. We can purchase alkylate from another company at 34 $/Bl. Would we use alkylate at this price in the blend? (Note that it is more costly that the gasoline that we are selling.)

3. We have a customer who will purchase all of the n-butane that we are using in the blend at 20.6 $/Bl. Should we sell or use it in the blend?
4. We have fixed the production rate at 7000 Bl. Is this optimum; if not, should we increase or decrease the blended product quantity to increase profit, assuming that we could sell any amount. What is the effect on profit, and over what range is this effect valid.
5. To reduce the vaporization of hazardous materials, the government wants us to lower the vapor pressure (RVP) of the product. What would be the cost of lowering the maximum RVP to 9?
6. Formulate and solve a meaningful sensitivity problem.

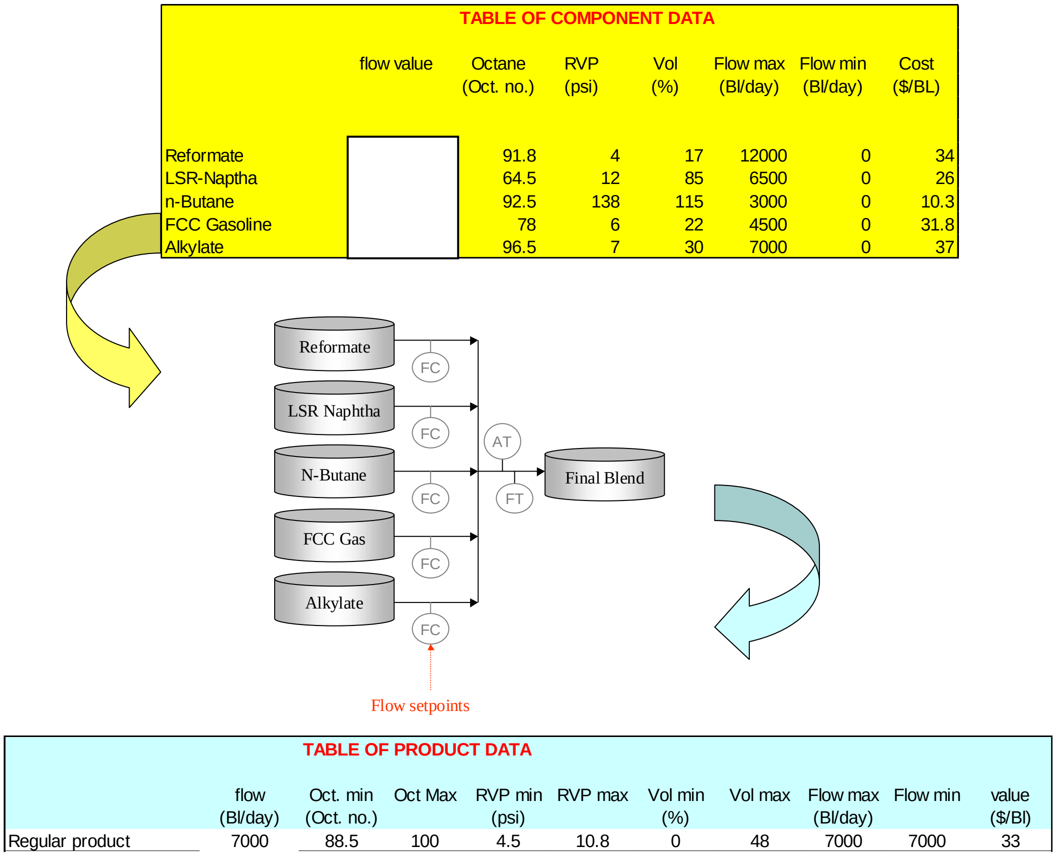


TABLE OF COMPONENT DATA

	flow value	Octane (Oct. no.)	RVP (psi)	Vol (%)	Flow max (Bl/day)	Flow min (Bl/day)	Cost ($/BL)
Reformate		91.8	4	17	12000	0	34
LSR-Naptha		64.5	12	85	6500	0	26
n-Butane		92.5	138	115	3000	0	10.3
FCC Gasoline		78	6	22	4500	0	31.8
Alkylate		96.5	7	30	7000	0	37

TABLE OF PRODUCT DATA

	flow (Bl/day)	Oct. min (Oct. no.)	Oct Max	RVP min (psi)	RVP max	Vol min (%)	Vol max	Flow max (Bl/day)	Flow min	value ($/Bl)
Regular product	7000	88.5	100	4.5	10.8	0	48	7000	7000	33

Figure Q8.17 Gasoline blending problem in Question 8.17.

8.18 We will reconsider the process in Exercise 8.17. The production-planning group has noticed that the inventories of n-butane and LSR-Naphtha are very high. They tell the blending group that they must have 700 barrels of both of these components in the blend.

a. Solve the blending problem for this case.

b. Devise a *goal programming* approach to do the best in this difficult situation.

8.19 Determine which modelling method that was presented in this section is similar to the absolute value model.

8.20 Reconsider the base case blending problem in Questions 8.16 and 8.17. Suppose that we were not sure of the component qualities. For example, the reformate octane could be one of the following values; 92.5, 91.8, 91. Determine the minimum profit for the blend when the blending flows have to be determined with this uncertainty.

8.21 Describe two other process examples of minimum-proportional models. Formulate the modelling equations for linear programming.

8.22 A situation similar to the formulation in Section 8.10 is encountered when a variable is proportional up to a maximum, which it cannot exceed. Describe a process example of this situation, and develop a mathematical model for linear programming.

9.0 Presenting Optimization Results

9.1 Write a report for the results that you obtained in Questions 8.1 and 2.

9.2 Write a report for the results that you obtained in Question 8.17.

Appendix A. Example Linear Programming Problem: Production Planning

The small problem in this appendix demonstrates many of the important aspects of linear programming. The student should solve this problem while reading the chapter

1. Problem statement: Your plant can purchase either of two feed materials in any quantity between their lower and upper bounds. The plant produces three products from these feeds. The yields of each feed to each product and the product bounds are given in the following table.

Table 1. Data for the Classroom LP Example Problem

	Feed flow	Product 1	Product 2	Product 3	min Feed	max Feed	Cost Feed
Feed 1	??	0.7	0.2	0.1	0	1000	5
Feed 2	??	0.2	0.2	0.6	0	1000	6
min product		0	0	0			
max product		100	70	90			
Value product		10	11	12			

Formulate the optimization problem in the general form that can be solved using mathematical programming methods.

a. Define an objective function and the variables
b. Develop equality constraints,
c. Develop inequality constraints.
d. Develop variable bounds
e. Is there anything else that you need?

2. Qualitative analysis: Determine whether this problem involves

a. Operation at an obvious limit.
b. No change to the objective given the limitations on the plant.
c. A worthwhile optimization problem; what are the tradeoffs?.

Can you determine the best operation without calculations?

3. Visualize the problem: Sketch the problem in two dimensions, giving the following.

a. The feasible region
b. Contours of constant objective function value
c. The location of the optimum.

4. Problem solution: Formulate the problem you developed in 1 above in the Excel spreadsheet. Solve the problem and verify the solution you obtained graphically.

a. Does a solution exist? (Does a non-zero feasible region exist?)
b. Is the solution bounded?
c. Does the solution agree with your graphical result?
d. Do alternative (multiple) optima exist?
e. How many constraints are active?
f. Which variables are "basic" or "in the basis"?

Naturally, the answers in questions 3, 4 should be the consistent.

5. Base case sensitivity: Answer the following questions using the results from the base case obtained in question 4.

a. For the constraints that are active, what is the shadow price (marginal value) for each rhs? What is the range for each? What determines the range? Report in a table.
b. For the variable bounds that are active, what is the marginal value for each bound? What is the range for each? What determines the range? Report in a table.
c. If the maximum sales of product 3 decreased to 60, what would be the effect on the solution?
d. If the maximum sales of product 1 increased to 110 and the maximum value of product 3 decreased to 81, would the same constraints be active?
e. What is the meaning of the objective coefficient for the feed flows? Why are they different from Table 1?

6. Larger parameter changes: Answer each of these questions. Some or all will require you to make a change and re-run the solver. Answer each question using the **Base Case** as the starting conditions.

For each case, answer the following questions.

i. Can you determine the result without resolving the LP with modified data?
ii. What is the effect on the optimum values of the variables and on the objective function?
iii. Is there anything of concern in the solution, e.g., weird events?

a. The maximum allowed production of product 2 is reduced to 47.5.
b. The value of product 3 decreases to 3.25 because of a new competitor that is cutting costs to get into the market.
c. We have a contract that requires us to accept at least 150 of feed 1.

7. Qualitative sensitivity: Answer these questions for very large changes to selected parameters from the base case. Here, we investigate the trends when problem parameters change. Answer the following two questions for each change.

i. State whether the variable values will change and whether the profit will increase, decrease or remain the same. In answering this part, give a qualitative result <u>without referring to the sensitivity output values</u>.

ii. Give a bound on the value of the change in the objective function. In answering this part, <u>you may</u> <u>refer to the sensitivity output values for the base case</u>.

a. The maximum production of Product 3 is increased to 100.

b. The feed cost increases from 6 to 7.

8. Reporting results:

a. Design and build an EXCEL spreadsheet that clearly displays the problem input data and solution results.

b. Write a report explaining your optimization study using all results.

c. Graph the effect on the profit and the purchase of the two available feeds of changing the maximum production of product 2 from 0 to 500.

Appendix B. Learning Resources for Linear Programming on the WWW

The WWW is a vast and ever-changing resource that contains, among some less savory contents, learning resources for mathematical programming and specifically, linear programming. The resources described here are selected to match the level of mathematics in the chapter and to be of most interest to the person formulating models and using optimization (not necessarily to the mathematician or software developer). Fortunately, many excellent resources are available in the public domain.

We thank the developers for their generosity in making their work freely available and congratulate them on the quality of their products.

Topic	Resource URL	Author and Comments
Glossary of terms in optimization	http://glossary.computing.society.informs.org	Prepared by Dr. H. Greenberg at University of Colorado at Denver
Linear Algebra Interactive Tool	http://www.ifors.ms.unimelb.edu.au/tutorial/	Interactive tools to practice matrix inversion and solving sets of linear equations. Select "The Equator" or "The Inverter" from the left-hand menu. Prepared by Dr. Sniedovich at the University of Melbourne, Australia.
Linear Programming	http://home.ubalt.edu/ntsbarsh/Business-stat/opre/partVIII.htm	A text presentation on linear programming by Dr. H. Arsham at the University of Baltimore.
Frequently asked questions about LP	http://www-unix.mcs.anl.gov/otc/Guide/faq/	From NEOS by Northwestern University and Argonne National Laboratory, USA.
Linear programming visual solver	http://www.cs.stedwards.edu/%7Ewright/linprog/AnimaLP.html	This site allows you to solve a two-dimensional LP and automatically plot the result.
Linear Programming Interactive Tool	http://www.ifors.ms.unimelb.edu.au/tutorial/	Interactive tools to practice LP Simplex method by tableau. Select "The Simplex Place" from the left-hand menu; then, select "Standard form". Prepared by Dr. Sniedovich at the University of Melbourne, Australia.
Explanation of common misunderstandings and "tricky points"	http://home.ubalt.edu/ntsbarsh/Business-stat/opre/partv.htm	"The dark side of LP" by Dr. H. Arsham at the University of Baltimore.
LP software	http://www.lionhrtpub.com/orms/surveys/LP/LP-surveymain.html	Survey of LP software from 2001