Linear Programming 3

Mathematical programming is concerned with the extremization of a function f defined over an n-dimensional design space $\mathbf{R}^n$ and bounded by a set $\mathbf{S}$ in the design space. The set $\mathbf{S}$ may be defined by equality or inequality constraints, and these constraints may assume linear or nonlinear forms. The function f together with the set $\mathbf{S}$ in the domain of f is called a *mathematical program* or a mathematical programming problem. This terminology is in common usage in the context of problems which arise in planning and scheduling which are generally studied under operations research, the branch of mathematics concerned with decision making processes. Mathematical programming problems may be classified into several different categories depending on the nature and form of the design variables, constraint functions, and the objective function. However, only two of these categories are of interest to us, namely *linear* and *nonlinear programming* problems (commonly designated as LP and NLP, respectively).

The term linear programming (LP) describes a particular class of extremization problems in which the objective function and the constraint relations are linear functions of the design variables. Because the necessary condition for an interior minimum is the vanishing of the first derivative of the function with respect to the design variables, linear programming problems have a special feature. That is, the derivatives of the objective function with respect to the variables are constants which are not necessarily zeroes. This implies that the extremum of a linear programming problem cannot be located in the interior of the feasible design space and, therefore, must lie on the boundary of the design space described by the constraint relations. Since the constraint relations are also linear functions of the design variables the optimum design must lie at the intersection of two or more constraint functions, unless the bounding constraint is parallel to the contours of the objective function. This special feature of the linear programming problems makes it possible to devise effective algorithms that are suitable for reaching optimum solutions. Linear programming problems involving large number of design variables and constraints are usually solved by an extremely efficient and reliable method known as the simplex method.

Unfortunately, however, very few physically meaningful problems in structural design, if any, can be formulated directly as LP problems without involving a degree

of simplification. Most structural design problems involve highly nonlinear objective function and constraint relations. Nevertheless, the category of LP is of interest to us because of several reasons. First of all, many nonlinear constrained problems can be approximated by linear ones which can be solved efficiently by using standard LP algorithms. Using such approximations opens up a possibility for solving NLP problems. That is, almost all NLP problems can be solved as a sequence of repetitive approximate LP problems which converge to the exact solution of the original NLP problem provided that the procedure is repeated enough number of times. This powerful procedure is called sequential linear programming (SLP) and is discussed in Chapter 6. Also, methods intended for nonlinear constrained problems often utilize linear programming as an intermediate step. For example, Zoutendijk's method of feasible directions (see Chapter 5) employs a LP to generate a search direction.

Whether a given nonlinearly constrained problem in structural optimization can be replaced by an approximately equivalent linearly constrained problem depends to a great extent on the intuition of the designer and his knowledge and experience with the given problem. Such approximations must usually be made so as not to alter the overall character of the problem radically. The trade-off between a higher value of the objective function attained because of the approximation and a lower computational cost must be weighted carefully. Fortunately, there are a few classes of problems in structural analysis and design in which such approximations have found to be indeed reasonable. In the following sections some of those problems will be presented as LP problems, and graphical solution of a simple LP problem will be demonstrated. Next, the standard formulation of the mathematical LP problems will be presented, and solution techniques for LP problems will be discussed. Finally, we would discuss a special case of LP problems that require the design variables to assume values from a set of discrete or integer values.

3.1 Limit Analysis and Design of Structures Formulated as LP Problems

In many structural design problems the initiation of yielding somewhere in the structure is considered to be a criterion for failure, but this is not always reasonable. In many cases we are not interested in the initiation of failure but in the maximum load, called the *limit load* or the *collapse load*, that a structure may carry without losing its functionality. The collapse load can be defined as the load required to generate enough local plastic yield points (referred as plastic hinges for bending type members) to cause the structure to become a mechanism and develop excessive deflections. While the exact calculation of the collapse load of a structure requires the solution of a costly nonlinear problem, for ductile materials it is possible to obtain a conservative estimate of that load by making the assumption that the material behaves as an elastic-perfectly plastic material. That is, the material is assumed to follow the stress-strain diagram shown in Fig. 3.1.1, yielding at stress level σ_0 but functioning as a constant stress carrying medium beyond the elastic limit. It is this important assumption that allows the limit analysis and design problems to be formulated as LP problems.

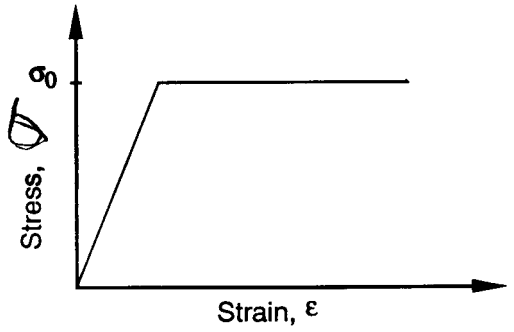

Figure 3.1.1 The stress-strain curve for an elastic-perfectly plastic material.

A simple example of a three bar truss is used in the following example to illustrate the difference between the calculation of the load which initiates yielding and the estimate of the collapse load.

Example 3.1.1

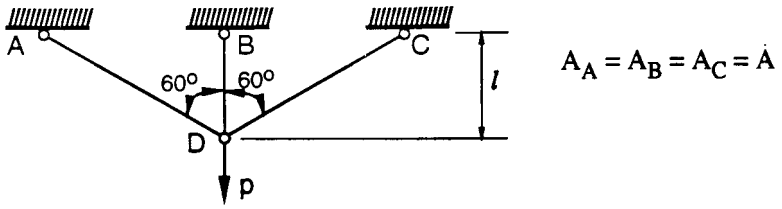

Figure 3.1.2 Collapse of a three bar truss subject to a single load.

We perform the collapse analysis of a three bar pin jointed truss under a vertical load as shown in Fig. 3.1.2. All three bars have the same cross-sectional area A, and are made of material having Young's modulus E and yield stress σ_0. We start by calculating the load p at which the first bar yields. Denoting the vertical displacement at the common joint D by v, we obtain the strains in the three members

$$\epsilon_B = \frac{v}{l}, \qquad \epsilon_A = \epsilon_C = \frac{v}{4l} \ . \tag{3.1.1}$$

The corresponding member forces are

$$n_B = \frac{EA}{l}v, \qquad n_A = n_C = \frac{EA}{4l}v = 0.25n_B \ . \tag{3.1.2}$$

Using the two equations of equilibrium at joint D, we get

$$n_A = n_B, \qquad p = n_B + \frac{1}{2}(n_A + n_C) = 1.25n_B \, , \tag{3.1.3}$$

and the internal forces in the three members are determined as

$$n_A = n_C = 0.2p, \qquad n_B = 0.8p \ . \tag{3.1.4}$$

Clearly, as the load is increased from zero member B yields first, when

$$n_B = \sigma_0 A, \qquad \text{or} \qquad p = 1.25 A\sigma_0 \ . \tag{3.1.5}$$

The structure does not collapse, however, at $p = 1.25A\sigma_0$ since members A and C can still carry the applied load without experiencing excessive deformations. We may increase the load until member A or C yields. Since we have assumed elastic-perfectly plastic material behavior, the stress in member B will remain at σ_0 as we increase the load beyond the initial yield load. Due to the symmetry in this problem, the next yielding takes place simultaneously in members n_A and n_C. Therefore, at collapse all three members will be at the yield point so that

$$n_A = n_B = n_C = A\sigma_0 \, , \tag{3.1.6}$$

and from the equations of equilibrium Eq. (3.1.4) we have

$$p_{\text{collapse}} = 2A\sigma_0 \ . \tag{3.1.7}$$

This is a 60% increase over the load when first yielding starts. • • •

In example 3.1.1 it was easy to identify the sequence of yielding of the members and determine the state of stress in the members at collapse. This fact permitted us to determine the collapse load without difficulty. In general, it is not easy to determine the combination of members that will yield at collapse, and the stress distribution at the collapse is not known. Fortunately, it is possible to cast the problem as an LP problem in order to determine the collapse load [1] based on a general theorem of the theory of plasticity. This theorem is the lower bound theorem, and it is quoted below from Calladine Ref. 2.

The Lower Bound Theorem: If any stress distribution throughout the structure can be found which is everywhere in equilibrium internally and balances the external loads, and at the same time does not violate the yield conditions, these loads will be carried safely by the structure.

The application of this theorem will now be demonstrated for a problem where the choice of stress at collapse is not as trivial as it was in example 3.1.1. We use the same structure used in the previous example, but with an added horizontal load at point D.

Example 3.1.2

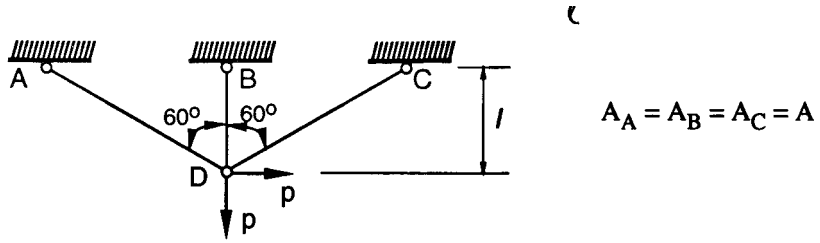

Figure 3.1.3 Limit analysis of a three bar truss subjected to two loads.

Consider the limit analysis of the three bar truss of Figure 3.1.3 under the combined vertical and horizontal loads of equal magnitude, p. The equations of equilibrium in this case are

$$\begin{aligned} n_B + \frac{1}{2}(n_A + n_C) - p = 0\,, \\ \frac{\sqrt{3}}{2}(n_A - n_C) - p = 0\,, \end{aligned} \tag{3.1.8}$$

and we have the constraints

$$-A\sigma_0 \quad \leq \quad n_A, n_B, n_C \quad \leq \quad A\sigma_0\ . \tag{3.1.9}$$

It is no longer easy to know which two of the three bars yield at the collapse. However, we may try different combinations of $n_A, n_B,$ and n_C that satisfy the equations of equilibrium in order to obtain a lower bound to the collapse load. For example, if we try $n_C = 0$, we obtain from the equilibrium relations (3.1.8)

$$n_A = \frac{2}{\sqrt{3}}p = 1.155p, \quad \text{and} \quad n_B = 0.423p\ . \tag{3.1.10}$$

Clearly in this case n_A reaches its yield value of $A\sigma_0$ before n_B so that

$$n_A = A\sigma_0, \quad n_B = 0.366A\sigma_0, \quad \text{and} \quad p = \frac{\sqrt{3}}{2}A\sigma_0 = 0.866A\sigma_0\ . \tag{3.1.11}$$

Having satisfied all the requirements for the lower bound theorem, we thus know that the collapse load is bounded below by $0.866A\sigma_0$. We can now try different combinations of member force distribution until we obtain a higher value of p than the one obtained in Eq. (3.1.11). To get the best estimate, we cast the problem as a maximization problem

$$\begin{aligned} &\textbf{maximize} \quad && p \\ &\textbf{such that} \quad && \text{Eqs. (3.1.8) and Eqs. (3.1.9) are satisfied.} \end{aligned} \tag{3.1.12}$$

This is clearly a LP problem in the variables n_A, n_B, n_C and p , and may be solved using any LP algorithm. It is also simple enough to admit a graphical solution if required (see Exercise 1). • • •

The general formulation of the calculation of the limit load for truss structures is similar to the procedure used in example 3.1.2 . It is assumed that no part of the truss structure fails by buckling before the plastic collapse load is reached. If we have a truss structure with r members loaded by a system of loads $\lambda\mathbf{p}$, where $\mathbf{p}$ is a given load vector and λ is a scalar, the limit load can be determined by finding the largest value of λ that the structure can support. The equations of equilibrium are written as

$$\sum_{j=1}^{r} e_{ij} n_j = \lambda p_i, \qquad i = 1, \ldots, m\,, \tag{3.1.13}$$

where n_j $(j = 1, \ldots, r)$ are the forces in each of the truss members, e_{ij} are direction cosines, and m is the number of equilibrium equations. The yield constraints are written as

$$A_j \sigma_{Cj} \leq n_j \leq A_j \sigma_{Tj}\,, \tag{3.1.14}$$

where $A_j, \sigma_{Cj},$ and σ_{Tj} are the cross-sectional areas, and the yield stresses in compression and tension, respectively. The limit or collapse load is then the solution to the following linear programming problem:

$$\begin{aligned} &\textbf{maximize} && \lambda \\ &\textbf{such that} && \text{Eq. (3.1.13) and Eq. (3.1.14) are satisfied,} \end{aligned} \tag{3.1.15}$$

where λ and the member forces n_j are treated as the design variables.

A related problem is the problem of *limit design* where the collapse load is specified and the optimal cross-sectional areas are sought. Often, the objective is to minimize the total mass of the structure

$$\textbf{minimize} \qquad m = \sum_{j=1}^{r} \rho_j A_j l_j\,, \tag{3.1.16}$$

where ρ_j and l_j are the mass density and the length of member j, respectively. The minimization problem of Eq. (3.1.16) has the same set of constraints, Eqs. (3.1.13) and (3.1.14), that applies to the limit analysis problem, but both n_j and A_j are treated as design variables. This time, however, the load amplitude λ is specified.

Example 3.1.3

Formulate the limit analysis and design of the five bar truss shown in Figure (3.1.4) as linear programs. Assume that all bars are made of the same material and that $\sigma_C = -\sigma_T = \sigma_0$.

The vertical and horizontal equations of equilibrium at the unrestrained nodes of the structure are

$$n_{13} + \frac{\sqrt{2}}{2} n_{23} = 0, \qquad n_{24} + \frac{\sqrt{2}}{2} n_{14} = 0\,, \tag{3.1.17a}$$

$$n_{34} + \frac{\sqrt{2}}{2} n_{23} = 0, \qquad n_{34} + \frac{\sqrt{2}}{2} n_{14} = p\ . \tag{3.1.17b}$$

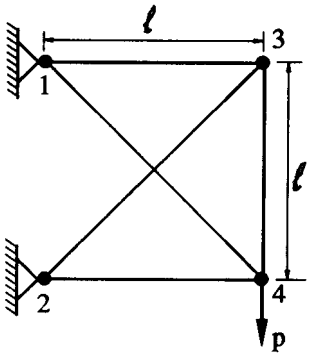

Figure 3.1.4 Limit analysis and design of a five bar truss.

The yield constraints are

$$\begin{aligned} &-A_{13}\sigma_0 \le n_{13} \le A_{13}\sigma_0, && -A_{23}\sigma_0 \le n_{23} \le A_{23}\sigma_0\,, \\ &-A_{14}\sigma_0 \le n_{14} \le A_{14}\sigma_0, && -A_{24}\sigma_0 \le n_{24} \le A_{24}\sigma_0\,, \\ &-A_{34}\sigma_0 \le n_{34} \le A_{34}\sigma_0\ . \end{aligned} \tag{3.1.18}$$

The limit load problem is specified as defined previously: maximize p, by varying the member forces, such that the equations of equilibrium and the yield constraints are satisfied. The limit design problem is

$$\begin{aligned} &\textbf{minimize} && \frac{m}{\rho l} = A_{13} + A_{24} + A_{34} + \sqrt{2}(A_{14} + A_{23}) \\ &\textbf{such that} && \text{Eq. (3.1.17) and Eq. (3.1.18) are satisfied.} \end{aligned} \tag{3.1.19}$$

For the limit design problem both the cross-sectional areas and the member forces are treated as design variables. • • •

The analysis and design of structures that include members under bending may be formulated as LP problems as in Refs. 3-5. Cohn, Ghosh, and Parimi [3] provide an excellent unified approach to both the analysis and design of beams, frames, and arches of given configurations under fixed, alternating, and variable repeated or shakedown loadings. We focus our attention here only on simple examples in this class of problems.

The basic hypothesis regarding the material is that the beam or frame is elastic-perfectly plastic. The fully plastic moment, m_p, of a beam cross-section is defined as the bending moment, m, required to make the entire cross-section yield so as to form a hinge with constant bending resistance.

Example 3.1.4

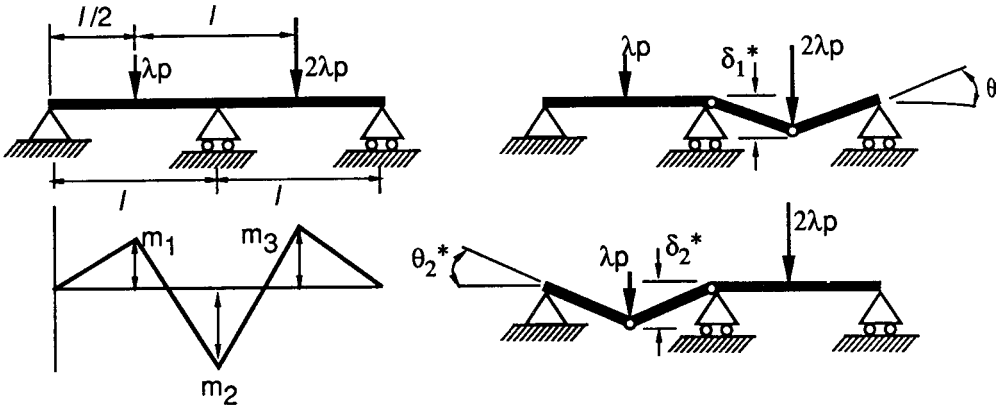

Figure 3.1.5 Limit analysis of a two-span beam.

Limit analysis of bending members is illustrated by using a two-span continuous beam under the loading shown in Figure 3.1.5. Following the general formulation presented earlier, the limit load is the largest value of λ that the structure can support without forming a mechanism. As in the case of Example 3.1.2 the sequence of hinge formation to form a beam mechanism and the distribution of the bending moments along the span of the beam is not obvious. In fact, there are infinitely many statically admissible bending moment distributions that satisfy the equilibrium equations. However, there are only two possible collapse mechanisms. The two elementary mechanisms and the moment distribution for the beam are presented in Figure 3.1.5.

The LP problem for the plastic analysis is

$$\begin{aligned} &\textbf{maximize} && \lambda \\ &\textbf{subject to} && -m_p \le m_i \le m_p, \qquad i = 1, 2, 3\,, \end{aligned} \tag{3.1.20}$$

where $m_1, m_2,$ and $, m_3$ are the magnitudes of the bending moment at those points along the beam which have the potential to form plastic hinges; at these points the bending moments have local maxima. These three moments are also unknowns for the problem and need to be determined. At the onset of either of the collapse mechanisms shown in Figure 3.1.5, we can write down two equations of equilibrium by using the principle of virtual displacements. The basic assumption in writing the virtual displacements is that the hinges in the figure are not plastic hinges, but are introduced to permit the small displacements that are assumed to take place while the members between them remain straight. The resulting equilibrium relations are

$$2m_3\theta_1^* + m_2\theta_1^* = 2\lambda p(l/2)\delta_1^*\,, \tag{3.1.21}$$

$$2m_1\theta_2^* + m_2\theta_2^* = \lambda p(l/2)\delta_2^*\,, \tag{3.1.22}$$

where $\theta_1^*,\ \theta_2^*$ are the virtual rotations of the member at the expected plastic joints and $\delta_1^*,\ \delta_2^*$ the virtual displacements of the beam under the load points. The virtual displacements and the rotations are related to one another through kinematic relations, and can be eliminated from the equations. Furthermore, using the two equilibrium equations, we can eliminate the two variables, m_1 and m_3, to reduce the LP problem of 3.1.20 to finding the λ and m_2 such that

$$\begin{aligned} \textbf{maximize} \quad & \lambda \\ \textbf{subject to} \quad & -m_p \le (\frac{pl}{4}\lambda - \frac{1}{2}m_2) \le m_p\,, \\ & -m_p \le m_2 \le m_p\,, \\ & -m_p \le (\frac{pl}{2}\lambda - \frac{1}{2}m_2) \le m_p\ . \end{aligned} \tag{3.1.23}$$

This is a simple two variable (m_2 and λ) LP problem that can be solved graphically.
• • •

Example 3.1.5

As an illustration of limit design for bending type problems, consider the well-known problem of minimizing the weight of a plane frame to resist a given set of ultimate loads. A single bay, single story portal frame is loaded by a horizontal and a vertical load of magnitude p as shown in Figure 3.1.6. For this design problem the top horizontal member is assumed to be different from the two vertical columns. Accordingly, we assume the beam and the column cross-sections to have associated fully plastic moments m_{pB} and m_{pC}, respectively. These two plastic moments depend on the cross-sectional properties of their respective members and, therefore, are the design variables for the problem.

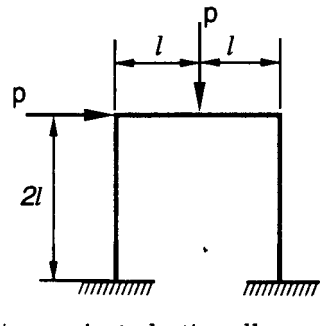

Figure 3.1.6 Portal frame design against plastic collapse.

In order to pose the problem as a weight minimization design problem, we need to relate the design variables and the structural weight. Massonet and Save [6] have shown that for beam sections in bending there is an approximate linear relation

between the weight per running foot, w_l, and the plastic section modulus, m_p/σ_0. Over the relevant range of sections that may be expected to be used for a given frame the error involved in this linearization is of the order of 1%. It is this single assumption which renders the plastic design problem linear.

We will, therefore, assume that the problem of minimizing the weight of a frame for a set of ultimate loads reduces to minimizing a function

$$w = 2m_{p_C} l_C + m_{p_B} l_B = 2m_{p_C}(2l) + m_{p_B}(2l) \ . \tag{3.1.24}$$

In the interest of non-dimensionalization we divide both sides of Eq. (3.1.24) by $2pl^2$ to obtain the weight proportional objective function

$$f(x_1, x_2) = (\frac{w}{2pl^2}) = 2\frac{m_{p_C}}{pl} + \frac{m_{p_B}}{pl} = 2x_1 + x_2 \ . \tag{3.1.25}$$

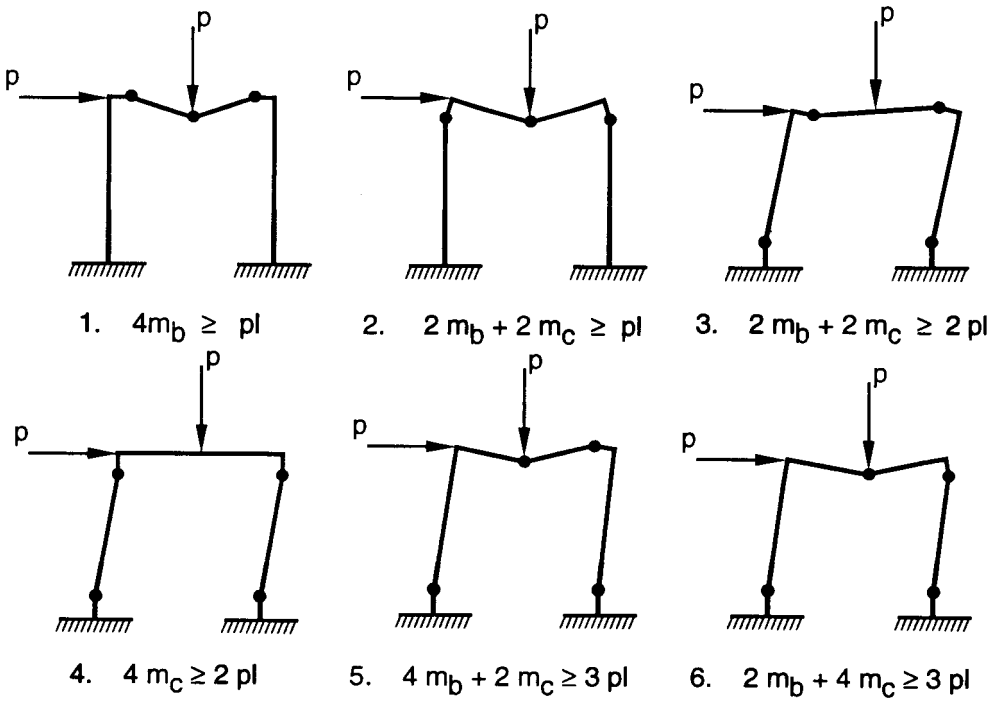

Figure 3.1.7 Collapse mechanisms for the portal frame of Figure 3.1.6.

The equations of equilibrium can be obtained by using the same approach used in the previous example. Figure 3.1.7 shows all possible collapse mechanisms for the frame. The ultimate load carrying capacity of the structure for any given collapse mechanism is obtained by the virtual work equivalence between the external work of the applied loads and the internal work of the fully plastic moments experienced

while undergoing virtual rotations of the plastic hinges. Thus a permissible design is one for which the capacity for internal virtual work is greater than or equal to the external work. It is left as an exercise (see Exercise 4) to verify that behavioral constraints associated with the collapse mechanism of Figure 3.1.7 reduce to

$$4x_2 \geq 1, \qquad \text{beam mechanism 1}, \tag{3.1.26}$$
$$2x_1 + 2x_2 \geq 1, \qquad \text{beam mechanism 2}, \tag{3.1.27}$$
$$x_1 + x_2 \geq 1, \qquad \text{sway mechanism 1}, \tag{3.1.28}$$
$$2x_1 \geq 1, \qquad \text{sway mechanism 2}, \tag{3.1.29}$$
$$2x_1 + 4x_2 \geq 3, \qquad \text{combined mechanism 1}, \tag{3.1.30}$$
$$4x_1 + 2x_2 \geq 3, \qquad \text{combined mechanism 2}. \tag{3.1.31}$$

Furthermore since x_1 and x_2 represent cross-sectional variables it is required that

$$x_1 \geq 0, \qquad \text{and} \qquad x_2 \geq 0\ . \tag{3.1.32}$$

Thus the problem of weight minimization under a set of ultimate load has been reduced to the determination of those non-negative values of $x_1$and x_2 for which f as given by Eq. (3.1.25) is minimized subject to constraints Eqs. (3.1.26 - 3.1.32). The problem is clearly an LP problem. We will defer the analytical solution of this problem until later. • • •

3.2 Prestressed Concrete Design by Linear Programming

Since concrete is weak in tension, prestressing helps to eliminate undesirable tensile stresses in concrete and thereby improve its resistance in bending. A prestressing cable or a tendon exerts an eccentrically applied compressive load to the beam cross-section giving rise to an axial load and possibly a bending moment due to an offset in the cable. In evaluating the total stresses at any given cross-section we must superimpose the stresses due to dead and live loads on the stresses due to the eccentrically applied prestressing forces of the tendons. For a beam of fixed cross-sectional dimensions, the total cost of the beam may be assumed to be approximately proportional to the cost of building in a desired prestressing force. The optimization problem for the design of a prestressed beam thus reduces to minimizing the magnitude of the prestressing force f_0.

Consider the following simple problem of the optimum design of the simply-supported beam shown in Figure 3.2.1 . The initial value of the prestressing force f_0 and the eccentricity f_e is to be determined such that f_0 is a minimum subject to constraints on normal stress, transverse displacement, and upper and lower bound constraints on the design variables. Additionally, in designing a prestressed concrete beam which is expected to remain in service for a number of years, we must allow for the loss of prestressing force through time dependent shrinkage and creep effects in concrete. To simplify design considerations it is frequently assumed that the realizable

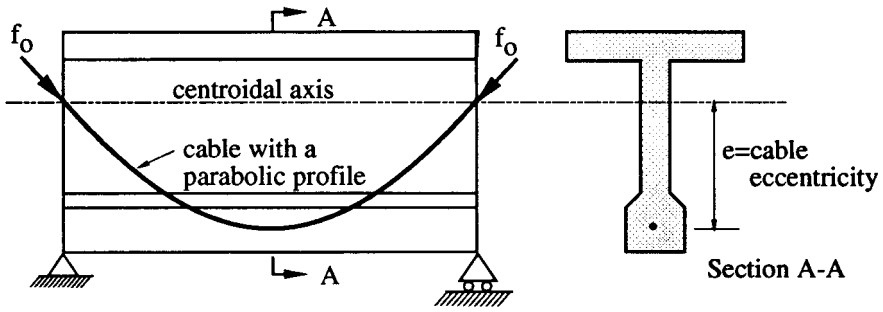

Figure 3.2.1 Simply supported post-tensioned beam.

prestressing force in service is a constant fraction α of the initial prestressing force f_0. In calculating the bending moment distribution or the deflected shape of a prestressed beam, in addition to the usual dead and live loads, we must allow for the equivalent distributed loading (see Exercise 6a) and the end loads resulting from the curved profile of the eccentrically placed tendons. It can be shown [7,8] that for parabolic profiles of the cables (see Figure 3.2.1) the induced moments and deflections are linearly related to the quantity $f_0 e$ with the constant of proportionality k being a function of the known material and cross-sectional properties. With this assumption maximum stresses and the deflections of a simply supported beam occur at the center of the beam. If the maximum positive bending moment and maximum deflection at the center of the simply-supported beam of Figure 3.2.1 due to external loads in the ith loading condition are denoted by m_{ei} and δ_{ei}, respectively, then the beam optimization problem reduces to

$$\textbf{minimize} \qquad f(f_0, e) = f_0 \tag{3.2.1}$$

$$\textbf{subject to} \qquad \sigma^{li} \leq -\frac{\alpha f_0}{a} \pm \frac{m_{ei} - \alpha f_0 e}{z} \leq \sigma^{ui}\,, \tag{3.2.2}$$

$$\delta^{li} \leq \delta_{ei} + \alpha k f_0 e \leq \delta^{ui}\,, \tag{3.2.3}$$

$$e^l \leq e \leq e^u\,, \tag{3.2.4}$$

$$f_0 \geq 0, \qquad i = 1, \ldots, nl\ . \tag{3.2.5}$$

Here nl denotes the number of different loading conditions; $\sigma^l, \sigma^u, \delta^l, \delta^u, e^l,$ and e^u denote lower and upper bounds on stress, deflections and the tendon eccentricity; a and z denote the effective area and the section modulus of the cross-section.

The problem as formulated by Eqs. (3.2.1) through (3.2.5) is not an LP problem because it includes the product $f_0 e$ of the two variables. However, it can be easily cast as one by letting

$$m = f_0 e\,, \tag{3.2.6}$$

and expressing the problems in terms of the new design variables f_0 and m. The transformed problem thus reduces to the following LP problem

$$\textbf{minimize} \qquad f(f_0, m) = f_0 \tag{3.2.7}$$

$$\textbf{subject to} \qquad \sigma^{li} \leq -\frac{\alpha f_0}{a} \pm \frac{m_{ei} - \alpha m}{z} \leq \sigma^{ui}\,, \tag{3.2.8}$$

$$\delta^{li} \leq \delta_{ei} + \alpha k m \leq \delta^{ui}\,, \tag{3.2.9}$$

$$m^l \leq m \leq m^u\,, \tag{3.2.10}$$

$$f_0 \geq 0, \qquad i = 1, \ldots, nl\,, \tag{3.2.11}$$

with m^l and m^u being the upper and lower bounds on $f_0 e$.

Morris [9] has treated a similar problem, but with additional constrains on ultimate moment capacity. He also modified the constraint (3.2.11) to allow the American Concrete Institute's limit on the prestressing force intended to prevent premature failure of the beam by pure crushing of the concrete. Morris linearizes part of the problem by using the reciprocal of the prestressing force as one of the design variables; this transformation however fails to linearize the constraint on the ultimate moment capacity. In the interest of linearization, this nonlinear constraint is replaced by a series of piecewise linear connected chords with true values at chord intersections. Kirsch [10] has shown that appropriate transformations can also be used to reduce the design of continuous prestressed concrete beams to equivalent linear programming problems. These problems involve not only the optimization of the prestressing force and the tendon configuration, but also the optimization of the cross-sectional dimensions of the beam.

3.3 Minimum Weight Design of Statically Determinate Trusses

As another example of the design problems that can be turned into LP problems we consider the minimum weight design of statically determinate trusses under stress and deflection constraints. The difficulty in these problems arises due to the nonlinear nature of the deflections as a function of the design variables which are the cross-sectional areas of the truss members. This type of problem, however, belongs to the class of what is known as separable programming [11] problems. In this class of programming the objective function and the constraints can be expressed as a sum of functions of a single design variable. Each such function can be approximated by a piecewise linear function or a set of connected line segments or chords interpolating the actual function at the chord intersections.

A nonlinear separable function of n design variables,

$$f = f(x_1, \ldots, x_n) = f_1(x_1) + f_2(x_2) + \ldots + f_n(x_n)\,, \tag{3.3.1}$$

can be linearized as

$$f = \sum_{k=0}^{m} \eta_{1k} f_{1k} + \sum_{k=0}^{m} \eta_{2k} f_{2k} + \ldots + \sum_{k=0}^{m} \eta_{nk} f_{nk}\,, \tag{3.3.2}$$

with

$$x_1 = \sum_{k=0}^{m} \eta_{1k} x_{1k}, \quad \ldots \quad , x_n = \sum_{k=0}^{m} \eta_{nk} x_{nk}\,, \tag{3.3.3}$$

$$\sum_{k=0}^{m} \eta_{1k} = \sum_{k=0}^{m} \eta_{2k} = \ldots = \sum_{k=0}^{m} \eta_{nk} = 1\,, \tag{3.3.4}$$

$$\eta_{jk} \geq 0, \qquad j = 0, 1, \ldots, n\,, \quad \text{and} \quad k = 0, 1, \ldots, m\,. \tag{3.3.5}$$

Here f_{jk} and x_{jk} are the values of the functions $f_1, f_2, \ldots, f_n$ and the design variables $x_1, x_2, \ldots, x_n$ at $m+1$ preselected points along each of the design variables, and η_{nk}'s are the interpolation functions for the design variables. Note that the number, m, of points selected for each design variable can, in general, be different ($m_1, m_2, \ldots, m_n,$ etc.), but for the sake of simplicity they are taken to be equal here.

The purpose of using m intervals with $m+1$ values of the design variables is to cover the entire range of the possible design space accurately. The number of segments m decides the degree of approximation to the original problem— the larger the m the closer the solution of the linear problem will be to the true solution. However, at any given design point, a linear approximation to a nonlinear function requires only the value of the function at two values of a design variable. We, therefore, require that for every design variable $j (j = 1, \ldots, n)$, at most two adjacent η_{jk} be positive. This implies that if, for example, η_{pq} and $\eta_{p(q+1)}$ are non-zero with all other η_{pk} zero, then the value of x_p is in the interval between x_{pq} and $x_{p(q+1)}$ and is given by

$$x_p = \eta_{pq} x_{pq} + \eta_{p(q+1)} x_{p(q+1)}, \qquad \text{with} \qquad \eta_{pq} + \eta_{p(q+1)} = 1\,. \tag{3.3.6}$$

The variables, $(x_1, \ldots, x_n)$, of the function have thus been replaced by the interpolation functions, η_{jk}, only two of which are constrained to be non-zero for each of the design variables. Therefore, we have a linear approximation to the function at every design variable.

Example 3.3.1

As an illustration we consider a problem similar to the one solved by Majid [12]. The objective is the minimum weight design of the four bar statically determinate truss shown in Figure 3.3.1 with stress constraints in the members and a displacement constraint at the tip joint of the truss. In order to simplify the problem we assume members 1 through 3 to have the same cross-sectional area A_1, and the member 4 the area A_2. Under the specified loading, the member forces and the vertical displacement at joint 2 can easily verified to be

$$F_1 = 5p, \quad F_2 = -p, \quad F_3 = 4p, \quad \text{and} \quad F_4 = -2\sqrt{3}p\,, \tag{3.3.7}$$

$$\delta_2 = \frac{6pl}{E}\left(\frac{3}{A_1} + \frac{\sqrt{3}}{A_2}\right)\,, \tag{3.3.8}$$

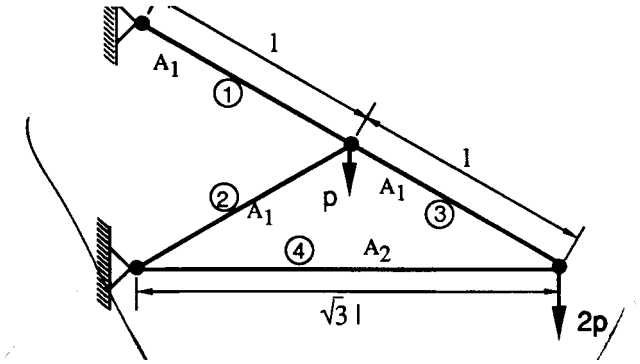

Figure 3.3.1 Four bar statically determinate truss.

where negative values for the forces denote compression. Allowable stresses in tension and compression are assumed to be $7.73 \times 10^{-4}E$ and $4.833 \times 10^{-4}E$, respectively and the vertical tip displacement is constrained to be no greater than $3 \times 10^{-3}l$. The problem of the minimum weight design subject to stress and displacement constraints can be formulated in terms of the non-dimensional variables

$$x_1 = \left(\frac{p}{A_1 E}\right) 10^3, \qquad \text{and} \qquad x_2 = \left(\frac{p}{A_2 E}\right) 10^3\,, \tag{3.3.9}$$

as

$$\textbf{minimize} \qquad f(x_1, x_2) = \frac{3}{x_1} + \frac{\sqrt{3}}{x_2} \tag{3.3.10}$$

$$\textbf{subject to} \qquad 18x_1 + 6\sqrt{3}x_2 \leq 3\,, \tag{3.3.11}$$

$$0.05 \leq x_1 \leq 0.1546\,, \tag{3.3.12}$$

$$0.05 \leq x_2 \leq 0.1395\,, \tag{3.3.13}$$

where lower bound limits on x_1 and x_2 have been assumed to be 0.05. Except for the objective function which is a separable nonlinear function, the rest of the problem is linear. The objective function can be put in a piecewise linear form by using Eqs. (3.3.2) and (3.3.3). For the purpose of demonstration, we divide the design variable intervals of Eqs. (3.3.12) and (3.3.13) into two equal segments ($m = 2$) resulting in

$$x_{10} = 0.05, \quad x_{11} = 0.1023, x_{12} = 0.1546\,,$$

$$\text{and } x_{20} = 0.05, \quad x_{21} = 0.09475, x_{22} = 0.1395\ .$$

Objective function values corresponding to these points are

$$f_{10} = 20, \quad f_{11} = 9.76, f_{12} = 6.47\,,$$

$$\text{and } f_{20} = 34.64, \quad f_{21} = 18.28, f_{22} = 12.42\ .$$

Therefore, the linearized objective function is

$$f(x_1, x_2) = 20\eta_{10} + 9.76\eta_{11} + 6.47\eta_{12} + 34.64\eta_{20} + 18.28\eta_{21} + 12.42\eta_{22}\ .$$

After substituting

$$x_1 = 0.05\eta_{10} + 0.1023\eta_{11} + 0.1546\eta_{12}\,,$$

$$\text{and } x_2 = 0.05\eta_{20} + 0.09475\eta_{21} + 0.1546\eta_{22}\,,$$

into the constraint equations of (3.3.11) through (3.3.13), a standard LP algorithm can be applied with the additional stipulation that only two adjacent η_{ik} for every design variable x_i be positive. • • •

3.4 Graphical Solutions of Simple LP Problems

For simple problems with no more than two design variables a graphical solution technique may be used to find the solution of a LP problem. A graphical method not only gives a solution, but also helps us to understand the nature of LP problems. The following example is included in order to illustrate the nature of the design space and the optimal solution.

Example 3.4.1

Consider the portal frame limit design problem of example 3.1.5. The problem was reduced to minimizing the objective function

$$f(x_1, x_2) = 2x_1 + x_2\,, \tag{3.4.1}$$

subject to inequality constraints Eqs. (3.1.26) through (3.1.32).

Since the problem is an LP problem in two-dimensional space it is possible to obtain a graphical solution. Constraints (3.1.32) imply that we can restrict ourselves to the non-negative quadrant of the $x_1 - x_2$ plane in Figure 3.4.1. We plot all the straight lines corresponding to Eqs. (3.1.26) through (3.1.31) as strict equalities (these lines identify the constraint boundaries). To identify the feasible and the infeasible portions on either side of a given constraint line we choose a point on either side and substitute its coordinates in the inequality. If the inequality is satisfied then the portion on the side of the constraint line which contains this point is the feasible portion, if not it is infeasible. For example, if the coordinates $x_1 = 0$ and $x_2 = 0$ are substituted into the inequality (3.1.27), the inequality is violated, implying that the origin does not belong to the feasible domain. If we continue this process for all the inequality constraints we will soon end up with a feasible region that is a convex polygon; the corners are called extreme points. The feasible region corresponding to the constraints is illustrated in Figure 3.4.1.

Next, we plot the contours of the objective function by setting the function $2x_1 + x_2$ equal to a constant and plotting the lines corresponding to various values of this constant. The optimum point is obtained by finding the contour of the objective function which just barely touches the feasible region. The direction of decreasing f is shown in Figure 3.4.1 with the optimum solution identified as

$$x_1 = x_2 = 1/2\,, \tag{3.4.2}$$

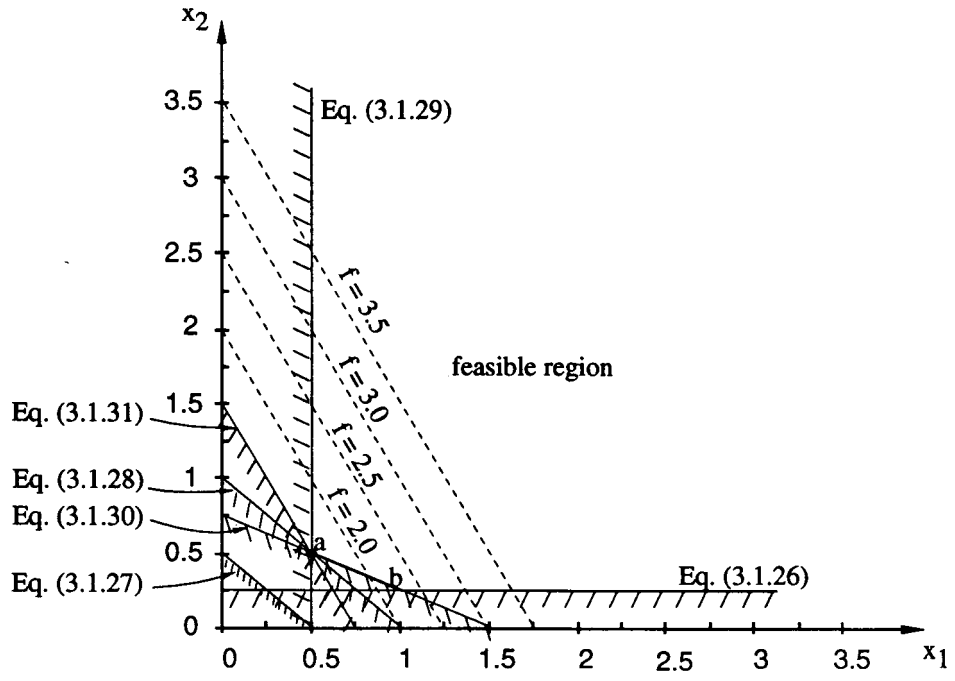

Figure 3.4.1 Graphical solution of the portal frame LP problem.

with $f_{\min} = 1.5$. • • •

Barring degeneracy, the optimum solution in an LP problem will always lie at a corner or an extreme point. The degenerate case may occur when the gradient of the objective function is a constant multiple of the gradient of one of the constraints along which the optimum solution lies. Then, every point along this constraint including the extreme points constitutes an optimum solution. For example if the problem just discussed had an objective function of the type

$$f = c(2x_1 + 4x_2)\,, \tag{3.4.3}$$

with c being a constant, then every point along the line [a,b] in Figure 3.4.1 would constitute an optimum solution.

The concept of a convex polygon with corners or vertices in two dimensions generalizes to a *convex polytope* with *extreme points* in $\mathbf{R}^n$. For example, a convex polytope [11] is defined to be the set which is obtained by the intersection of a finite number of closed half-spaces. Similarly, an extreme point of a set is defined to be a point $\mathbf{x}$ in $\mathbf{R}^n$ which cannot be expressed as a convex combination $\alpha\mathbf{x}_1 + (1-\alpha)\mathbf{x}_2$ $(0 < \alpha < 1)$ of two distinct points $\mathbf{x}_1$ and $\mathbf{x}_2$ belonging to the set. Finally, as in the two-dimensional case of Figure 3.4.1, barring degeneracy, a linear objective function in $\mathbf{R}^n$ achieves its minimum only at an extreme point of a bounded convex polytope.

Interested readers are advised to consult either Ref. 11 or 13 for a comprehensive treatise on this subject.

It is obvious that the above graphical procedure cannot be used for linear programming problems involving more than two variables. We have to look at alternative means of solving such problems. The simplex method first proposed by Dantzig [13] is an efficient method for solving problems with a large number of variables and constraints. We will study the simplex method next and to this end we outline a few definitions and some very important concepts in linear programming.

3.5 A Linear Program in a Standard Form

A linear program is said to be in a standard form if it is posed as

$$\textbf{minimize} \qquad f = \mathbf{c}^T\mathbf{x} \tag{3.5.1}$$

$$\textbf{subject to} \qquad \mathbf{A}\mathbf{x} = \mathbf{b}\,, \tag{3.5.2}$$

$$\mathbf{x} \geq \mathbf{0}\,, \tag{3.5.3}$$

where $\mathbf{c}$ is an $n \times 1$ vector, $\mathbf{A}$ is a $m \times n$ matrix, and $\mathbf{b}$ is a $m \times 1$ vector. Any linear program including inequality constraints can be put into the standard form by the use of what are known as *slack* and *surplus* variables. Consider, for example, the linear program defined by Eqs. (3.1.26) through (3.1.32). We can transform those inequalities into strict equalities as

$$4x_2 - x_3 = 1\,, \tag{3.5.4}$$

$$2x_1 + 2x_2 - x_4 = 1\,, \tag{3.5.5}$$

$$x_1 + x_2 - x_5 = 1\,, \tag{3.5.6}$$

$$2x_1 - x_6 = 1\,, \tag{3.5.7}$$

$$2x_1 + 4x_2 - x_7 = 3\,, \tag{3.5.8}$$

$$4x_1 + 2x_2 - x_8 = 3\,, \tag{3.5.9}$$

by the addition of the surplus variables x_3 through x_8, provided that these variables are restricted to be non-negative, that is

$$x_i \geq 0, \qquad i = 1, \ldots, 8\,. \tag{3.5.10}$$

If the inequalities in Eqs. (3.1.26) through (3.1.31) were of the opposite kind we would add non-negative variables x_3 through x_8 to achieve equality constraints. In this case the variables x_3 through x_8 would be referred to as the slack variables. If the original values of the design variables are not required to be non-negative we can still convert the problem to a standard form of Eqs. (3.5.1) through (3.5.3) by defining either

$$x_1 = u_1 - v_1, \qquad \text{and} \qquad x_2 = u_2 - v_2\,, \tag{3.5.11}$$

where $u_1, u_2, v_1, v_2 \geq 0$, or by adding a large enough positive constant M to the design variable

$$\bar{x}_1 = M + x_1\,, \tag{3.5.12}$$

so that the new variable never becomes negative during the design. Such artificial variables are often used in structural design problems where quantities such as stresses are used as design variables. Stresses can be both positive or negative depending upon the loading condition. It is clear from Eq. (3.5.11) that putting LP program in a standard form may cause an increase in the dimension of the design space. Using Eq. (3.5.12) does not increase the dimension of the problem but it may be difficult to know a priori the value of the constant M that will make the design variable positive (the choice of a very large number may result in numerical ill-conditioning).

Going back to Eq. (3.5.2) we notice that if $m = n$ and all the equations are linearly independent, we have a unique solution to the system of equations, whereas with $m > n$ we have, in general, an inconsistent system of equations. It is only when $m < n$ that we have many possible solutions. Of all these solutions we seek the one which satisfies the non-negativity constraints and minimizes the objective function f.

3.5.1 *Basic Solution*

We assume the rank of the matrix $\mathbf{A}$ to be m and select from the n columns of $\mathbf{A}$ a set of m linearly independent columns. We denote this $m \times m$ matrix by $\mathbf{D}$. Then $\mathbf{D}$ is non-singular and we can obtain the solution

$$\begin{array}{ccccc} \mathbf{x_D} & = & \mathbf{D}^{-1} & \mathbf{b_D}\,, \\ m \times 1 & & m \times m & m \times 1 \end{array} \tag{3.5.13}$$

where $\mathbf{x_D}$ is the vector of independent variables and $\mathbf{b_D}$ is the corresponding right-hand vector. Thus it can easily be verified that

$$\mathbf{x} = \left\{ \begin{array}{c} \mathbf{x_D} \\ \cdots \\ \mathbf{0} \end{array} \right\}\,, \tag{3.5.14}$$

is a solution of the system of Eqs. (3.5.2). Such a solution is known as a basic solution, and $\mathbf{x_D}$ is called the vector of basic variables. A basic solution, however, need not satisfy the non-negativity constraints (3.5.3). Those basic solutions which do indeed satisfy these constraints are known as basic feasible solutions and can be shown to be extreme points. In other words all basic feasible solutions to Eqs. (3.5.2) will correspond to corners or extreme points of the convex polytope [13].

The total number of possible basic solutions to Eqs. (3.5.2) can be estimated by identifying the number of possibilities for selecting m variables arbitrarily from a group of n variables. From the theory of permutations and combinations we know this number to be

$$\binom{n}{m} = \frac{n!}{m!(n-m)!}\,. \tag{3.5.15}$$

Not all of these possibilities will however be feasible.

3.6 The Simplex Method

The idea of the simplex method is to continuously decrease the value of the objective function by going from one basic feasible solution to another until the minimum value of the objective function is achieved. We will postpone the discussion of how to generate a basic feasible solution and assume that we have a basic feasible solution to start the algorithm. Indeed, if we had the following inequality constraints

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n \leq b_i, \qquad i = 1, \ldots, m\,, \tag{3.6.1}$$

$$x_j \geq 0, \qquad j = 1, \ldots, n\,, \tag{3.6.2}$$

where $b_i \geq 0$ for every constraint, then the process of converting the constraint set to the standard form yields the following

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{in}x_n + y_i = b_i, \qquad i = 1, \ldots, m\,, \tag{3.6.3}$$

$$x_j \geq 0, \qquad j = 1, \ldots, n\,, \tag{3.6.4}$$

$$y_i \geq 0, \qquad i = 1, \ldots, m\,, \tag{3.6.5}$$

and we immediately recognize

$$y_i = b_i, \qquad i = 1, \ldots, m, \qquad \text{and} \qquad x_j = 0\,, \qquad j = 1, \ldots, n\,, \tag{3.6.6}$$

as a basic feasible solution. A formal scheme for generating a basic feasible solution will be discussed later in this section. The question of immediate interest at this point is how to go from one basic feasible solution to another basic feasible solution. Without loss of generality let us assume that we have a system of equations in the canonical form shown below (such forms can always be obtained through the well-known Gauss elimination scheme for a matrix $\mathbf{A}$ with rank m).

$$\begin{array}{ccccccccccc}
x_1 & +\,0 & +\ldots & +\,0 & +\ldots & +\,0 & +a_{1,m+1}\,x_{m+1} & +\ldots & +a_{1,n}\,x_n & = & b_1 \\
0 & +x_2 & +\ldots & +\,0 & +\ldots & +\,0 & +a_{2,m+1}\,x_{m+1} & +\ldots & +a_{2,n}\,x_n & = & b_2 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & & \vdots \\
0 & +\,0 & +\ldots & +x_s & +\ldots & +\,0 & +a_{s,m+1}\,x_{m+1} & +\ldots & +a_{s,n}\,x_n & = & b_s \\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & & \vdots \\
0 & +\,0 & +\ldots & +\,0 & +\ldots & +x_m & +a_{m,m+1}\,x_{m+1} & +\ldots & +a_{m,n}\,x_n & = & b_m
\end{array}\,, \tag{3.6.7}$$

with a basic feasible solution

$$x_1 = b_1, \quad x_2 = b_2, \quad \ldots \quad x_s = b_s, \quad \ldots \quad x_m = b_m\,,$$

$$x_{m+1} \quad = \quad x_{m+2} \quad = \quad \ldots \quad = \quad 0\,. \tag{3.6.8}$$

The variables x_1 through x_m are called basic and the x_{m+1} through x_n are called non-basic variables.

3.6.1 *Changing the Basis*

The simplex procedure changes the set of basic variables while improving the objective function at the same time. However, for the purpose of clarity we will first demonstrate the approach for going from one basic feasible solution to another. The objective function improvement will be discussed in the following section.

We wish to make one of the current non-basic variables of Eq. (3.6.7), say x_t ($m < t \leq n$), basic and in the process cause a basic variable, $x_s (1 \leq s \leq m)$, to become non-basic. At this point we assume that we know the variable x_t which we will bring into the basic set. We only need to decide which variable to drop from the basic set. Consider the selected terms shown below for the coefficients of the sth equation and an additional arbitrary ith equation.

$$\begin{array}{cccccccccc} & i & & & s & & t & & & \\ & & & & & & & & & \\ i & 1 & \ldots & & 0 & \ldots & a_{it} & \ldots & = & b_i \\ & \vdots & & & \vdots & & \vdots & & & \vdots \\ s & 0 & \ldots & & 1 & \ldots & a_{st} & \ldots & = & b_s \end{array} \tag{3.6.9}$$

Since we want to make x_t basic, we need to eliminate it from the rest of the equations except the sth one by reducing the coefficients a_{it} ($i = 1, \ldots, n; \quad i \neq s$) to zeroes, and making the coefficient a_{st} unity by dividing the sth equation throughout by a_{st}. We can do this only if a_{st} is non-zero. Also, unless a_{st} is positive, the process of dividing the sth equation by a_{st} will produce a negative term on the right-hand side since b_s is positive because the current solution is a basic feasible solution. To eliminate the new basic variable x_t from the ith equation ($i = 1, \ldots, n; \quad i \neq s$) we have to multiply the sth equation by the factor (a_{it}/a_{st}) and subtract the resulting equation from each of these equations. The resulting coefficients on the right-hand side of the ith equation will be

$$b_i' = b_i - b_s(\frac{a_{it}}{a_{st}}) \, . \tag{3.6.10}$$

To guarantee that the resulting solution is a basic feasible solution we must require that $b_i' \geq 0$, or rearranging Eq. (3.6.10) we have

$$(\frac{b_s}{a_{st}}) \leq (\frac{b_i}{a_{it}}) \, . \tag{3.6.11}$$

Equation (3.6.11) together with the condition

$$a_{st} > 0 \, , \tag{3.6.12}$$

are the two conditions which identify possible sth rows in changing from one basic feasible solution to another. Thus for a given non-basic variable x_t that is to be made basic we check the coefficients of all the terms in the tth column. We eliminate from consideration all elements in the tth column with non-positive coefficients as violating condition (3.6.12). Among those with positive coefficients we compute the ratios b_i/a_{it} ($i = 1, \ldots, n$). We select the row, s, for which the ratio b_i/a_{it} has the smallest value and call it b_s/a_{st}, Eq. (3.6.11). It is the basic variable corresponding to that row which will become non-basic in the process of making x_t basic.

Example 3.6.1

We illustrate the foregoing discussion with an example. Consider the system of equations

$$\begin{aligned} 2x_1 + 2x_2 + x_3 &= 6\,, \\ 3x_1 + 4x_2 + x_4 &= 10\,, \\ x_1 + 2x_2 + x_5 &= 4\ . \end{aligned} \tag{3.6.13}$$

The system is already in the canonical form with a basic feasible solution being

$$x_1 \ = \ x_2 \ = \ 0, \qquad x_3 \ = \ 6, \quad x_4 \ = \ 10, \quad x_5 \ = \ 4\ . \tag{3.6.14}$$

The variables x_1 and x_2 are the non-basic variables, whereas $x_3, x_4,$ and $, x_5$ are the basic variables. Now, let us assume that we want to make x_1 basic. Rewriting Eqs. (3.6.13) in a matrix form we have

$$\begin{bmatrix} 2 & 2 & 1 & 0 & 0 \\ 3 & 4 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{Bmatrix} = \begin{Bmatrix} 6 \\ 10 \\ 4 \end{Bmatrix}\ . \tag{3.6.15}$$

Since x_1 is to made basic we consider the first column. To chose the variable to be made non-basic we form the ratios (b_i/a_{i1}), $i = 1, 2, 3$.

$$\frac{b_1}{a_{11}} = 3, \qquad \frac{b_2}{a_{21}} = 3\frac{1}{3}, \qquad \frac{b_3}{a_{31}} = 4\ .$$

The smallest ratio is b_1/a_{11} and so we pivot on a_{11}. Thus the new system of equations is

$$\begin{bmatrix} 1 & 1 & 0.5 & 0 & 0 \\ 0 & 1 & -1.5 & 1 & 0 \\ 0 & 1 & -0.5 & 0 & 1 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{Bmatrix} = \begin{Bmatrix} 3 \\ 1 \\ 1 \end{Bmatrix}\ , \tag{3.6.16}$$

and the process of making x_1 basic has resulted in the variable x_3 being non-basic. The new feasible solution is

$$x_2 \ = \ x_3 \ = \ 0, \qquad x_1 \ = \ 3, \quad x_4 \ = \ 1, \quad x_5 \ = \ 1\ .$$

It may be verified by the reader that by using a pivot other than a_{11} we would end up with an infeasible basic solution. For example, if a_{13} is a pivot we obtain

$$x_2 \ = \ x_5 \ = \ 0, \qquad x_1 \ = \ 4, \quad x_3 \ = \ -2, \quad x_4 \ = \ -2\,,$$

which is not feasible since $x_3 < 0$ and $x_4 < 0$. • • •

3.6.2 *Improving the Objective Function*

In the preceding section we considered making a particular non-basic variable x_t basic without losing feasibility. We also need to decide the variable that we make basic. We should seek to bring into the basis only that variable which will decrease the objective function while yielding at the same time a basic feasible solution. Notice that the objective function is a linear equation just like the other equations and hence it can be included with the others. The objective function equation may be written as

$$\mathbf{c}^T\mathbf{x} = f \ . \tag{3.6.17}$$

Assume the system of equations (3.5.2) is in the canonical form, and append Eq. (3.6.17) at the end of all other equations. The form of the equations that includes the objective function is often referred as the *simplex tableau*. We now eliminate all the basic variables from this last equation by subtracting c_i times each of the equations in the canonical form. Then the right-hand of Eq. (3.6.17) becomes $(f - c_1b_1 - c_2b_2 - c_3b_3 - \ldots - c_mb_m)$. Thus if we ignore the presence of f, the right-hand side represents the negative of the value of the objective function since $x_{m+1} = x_{m+2} = \ldots = x_n = 0$. The left-hand side of this last equation will contain only non-basic variables. Next, assume that the coefficient of one of the non-basic variables on the left-hand side of the last equation is negative. If we make this variable basic then we will increase the value of this variable from its present value of zero to some positive value. Since the last equation is just one of the equations, when we pivot on one of the equations (sth) and eliminate the corresponding variable (x_s) from the basic set we perform the operations described in the previous section on all the $m + 1$ equations. When the particular variable with the negative coefficient in the last equation is eliminated, the right-hand side of this equation will increase since the variable has increased in value from zero to a positive value. Since the right-hand side represents the negative of the value of the objective function, a function decrease is therefore guaranteed. Thus the criterion for guaranteeing an improvement of the objective function is to bring into the basis a variable that has a negative coefficient in the objective function equation after it has been cleared of all the basic variables. This can be verified by the following example.

Example 3.6.2

$$\begin{aligned}
\textbf{minimize} \quad & f = x_1 + x_2 + x_3 && (3.6.18)\\
\textbf{subject to} \quad & 2x_1 + 2x_2 + x_3 = 6\,, && (3.6.19)\\
& 3x_1 + 4x_2 + x_4 = 10\,, && (3.6.20)\\
& x_1 + 2x_2 + x_5 = 4\ . && (3.6.21)
\end{aligned}$$

As mentioned above we rewrite the constraint equations (3.6.21) in the matrix form together with the objective function appended as the last row of the matrix

$$\left[\begin{array}{ccccc} 2 & 2 & 1 & 0 & 0 \\ 3 & 4 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 1 \\ -- & -- & -- & -- & -- \\ 1 & 1 & 1 & 0 & 0 \end{array}\right] \left\{\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array}\right\} = \left\{\begin{array}{c} 6 \\ 10 \\ 4 \\ -- \\ 0 \end{array}\right\} \ . \tag{3.6.22}$$

A basic solution is

$$x_1 = x_2 = 0, \qquad x_3 = 6, \quad x_4 = 10, \quad x_5 = 4\ . \tag{3.6.23}$$

The variable x_3 is a basic variable that appears in the last equation of Eqs. (3.6.22) and must be eliminated from it so that its right-hand side yields the negative of the current value of the objective function.

$$\left[\begin{array}{ccccc} 2 & 2 & 1 & 0 & 0 \\ 3 & 4 & 0 & 1 & 0 \\ 1 & 2 & 0 & 0 & 1 \\ -- & -- & -- & -- & -- \\ -1 & -1 & 0 & 0 & 0 \end{array}\right] \left\{\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array}\right\} = \left\{\begin{array}{c} 6 \\ 10 \\ 4 \\ -- \\ -6 = -f \end{array}\right\}\ . \tag{3.6.24}$$

We can pivot either on column (1) or column (2). That is to say the objective function will decrease in value by bringing either x_1 or x_2 into the basis. If we pivot on column (1) (bringing x_1 into the basis) the pivot element is a_{11} because it yields the smallest (b_i/a_{i1}) ratio. The new simplex tableau becomes

$$\left[\begin{array}{ccccc} 1 & 1 & 0.5 & 0 & 0 \\ 0 & 1 & -1.5 & 1 & 0 \\ 0 & 1 & -0.5 & 0 & 1 \\ -- & -- & -- & -- & -- \\ 0 & 0 & 0.5 & 0 & 0 \end{array}\right] \left\{\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{array}\right\} = \left\{\begin{array}{c} 3 \\ 1 \\ 1 \\ -- \\ -3 = -f \end{array}\right\}\ . \tag{3.6.25}$$

The value of the objective function has been reduced from 6 to 3. Since the last equation contains no non-basic variable with a negative coefficient, it is no longer possible to decrease the value of the objective function further. Thus the minimum value of the objective function is 3 and corresponds to the basic solution

$$x_2 = x_3 = 0, \qquad x_1 = 3, \quad x_4 = 1, \quad x_5 = 1\ . \tag{3.6.26}$$

If we had decided to bring x_2 into the basis first, we would have reduced the objective function from 6 to 4, and there would have been a negative number in the last equation in the first column indicating the need for another round of pivoting to bring x_1 into the basis. • • •

This would have completed the discussion of the simplex method except for the fact that we need a basic feasible solution to start the simplex method and we may not have one readily available. This is our next topic.

3.6.3 *Generating a Basic Feasible Solution—Use of Artificial Variables*

In the process of converting an LP problem given in the form of Eqs. (3.6.4) and (3.6.5)

$$\mathbf{Ax} \le \mathbf{b}, \qquad \text{where} \qquad \mathbf{b} > \mathbf{0}, \quad \text{and} \quad \mathbf{x} \ge \mathbf{0}\,, \tag{3.6.27}$$

into the standard form by adding slack variables we obtained a basic feasible solution to start the simplex method. However, when we have a linear program which is

already in the standard form of Eqs. (3.5.2) and (3.5.3) we cannot, in general, identify a basic feasible solution. The following technique can be used in such cases.

Consider the following minimization problem

$$\textbf{minimize} \quad \sum_{i=1}^{m} y_i \tag{3.6.28}$$

$$\textbf{subject to} \quad \mathbf{A}\mathbf{x} + \mathbf{y} = \mathbf{b}\,, \tag{3.6.29}$$

$$\mathbf{x} \geq \mathbf{0}, \quad \text{and} \quad \mathbf{y} \geq \mathbf{0}\,, \tag{3.6.30}$$

where $\mathbf{y}$ is a vector of artificial variables. There is no loss of generality in assuming that $\mathbf{b} > \mathbf{0}$ so that the LP problem (3.6.29) has a known basic feasible solution

$$\mathbf{y} = \mathbf{b}, \quad \text{and} \quad \mathbf{x} = \mathbf{0}\,, \tag{3.6.31}$$

so that the simplex method can be easily applied to solve the LP problem of Eqs. (3.6.30). Note that if a basic feasible solution to the original LP problem (3.6.28) exists then the optimum solution to the modified problem (3.6.30) must have $\mathbf{y}_i$'s as non-basic variables ($\mathbf{y}= \mathbf{0}$). However, if no basic feasible solution to the original problem exists then the minimum value of Eq (3.6.29) will be greater than zero.

Example 3.6.3

We illustrate the use of artificial variables with the following example for which we seek a basic feasible solution to the system

$$\begin{aligned} 2x_1 + x_2 + 3x_3 &= 13\,, \\ x_1 + 2x_2 + x_3 &= 7\,, \\ x_i \geq 0, \qquad & i = 1, 2, 3\ . \end{aligned} \tag{3.6.32}$$

Introduce the artificial variables $y_1$and y_2 and pose the following minimization problem.

$$\textbf{minimize} \quad f = y_1 + y_2 \tag{3.6.33}$$

$$\begin{aligned} \textbf{subject to} \quad & 2x_1 + x_2 + 3x_3 + y_1 = 13\,, \\ & x_1 + 2x_2 + x_3 + y_2 = 7\,, \\ & x_i \geq 0, \quad i = 1, 2, 3\,, \text{ and } \qquad y_i \geq 0,\ j = 1, 2\ . \end{aligned} \tag{3.6.34}$$

With the basic feasible solution, $y_1 = 13,\ y_2 = 7,$ and $x_1 = x_2 = x_3 = 0$ known, we append the objective function (3.6.33) and clear the basic design variables y_1 and y_2 from it to obtain the initial simplex tableau

$$\left[\begin{array}{ccccc} 2 & 1 & 3 & 1 & 0 \\ 1 & 2 & 1 & 0 & 1 \\ -- & -- & -- & -- & -- \\ -3 & -3 & -4 & 0 & 0 \end{array}\right] \left\{\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ y_1 \\ y_2 \end{array}\right\} = \left\{\begin{array}{c} 13 \\ 7 \\ -- \\ -20 \end{array}\right\}\ . \tag{3.6.35}$$

Since it has the largest negative number we choose column (3) for pivoting with a_{13} as the pivot element since $13/3 < 7/1$,

$$\left[\begin{array}{ccccc} 2/3 & 1/3 & 1 & 1/3 & 0 \\ 1/3 & 5/3 & 0 & -1/3 & 1 \\ -- & -- & -- & -- & -- \\ -1/3 & -5/3 & 0 & 4/3 & 0 \end{array}\right] \left\{\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ y_1 \\ y_2 \end{array}\right\} = \left\{\begin{array}{c} 13/3 \\ 8/3 \\ -- \\ -8/3 \end{array}\right\} . \tag{3.6.36}$$

Next we choose a_{22} as the pivot element to obtain

$$\left[\begin{array}{ccccc} 9/15 & 0 & 1 & 6/15 & -1/5 \\ 1/5 & 1 & 0 & -1/5 & 3/5 \\ -- & -- & -- & -- & -- \\ 0 & 0 & 0 & 1 & 1 \end{array}\right] \left\{\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ y_1 \\ y_2 \end{array}\right\} = \left\{\begin{array}{c} 19/5 \\ 8/5 \\ -- \\ 0 \end{array}\right\} . \tag{3.6.37}$$

The process has converged to the basic feasible solution

$$x_1 = 0, \quad x_2 = 8/5, \quad \text{and} \quad x_3 = 19/5 \ . \tag{3.6.38}$$

to the original problem. • • •

3.7 Duality in Linear Programming

It was shown by Dantzig [13] that the primal problem of minimization of a linear function over a set of linear constraints is equivalent to the dual problem of the maximization of another linear function over another set of constraints. Both the dual objective function and constraints of the dual problem are obtained from the objective function and constraints of the primal problem. Thus if the primal problem is defined to be

$$\begin{aligned} \textbf{minimize} \qquad & f_p = c_1 x_1 + \ldots\ldots + c_n x_n = \mathbf{c^T x} && (n \text{ variables}) \\ \textbf{subject to} \qquad & \sum_{j=1}^{n} a_{ij} x_j \geq b_i, \quad i = 1, \ldots, m \,, && (m \text{ constraints}) \\ & x_j \geq 0, \quad j = 1, \ldots, n \,, && \end{aligned} \tag{3.7.1}$$

then the dual problem is defined to be

$$\begin{aligned} \textbf{maximize} \qquad & f_d = b_1 \lambda_1 + \ldots\ldots + b_m \lambda_m = \mathbf{b^T \boldsymbol{\lambda}} && (m \text{ variables}) \\ \textbf{subject to} \qquad & \sum_{i=1}^{m} a_{ij} \lambda_i \leq c_j, \quad j = 1, \ldots, n \,, && (n \text{ constraints}) \\ & \lambda_i \geq 0, \quad j = 1, \ldots, m \ . && \end{aligned} \tag{3.7.2}$$

The choice of the primal or dual formulation depends on the number of design variables and the number of constraints. The computational effort in solving an LP

problem increases as the number of constraints increases. Therefore, if the number of constraint relations is large compared to the number of design variables then it may be desirable to solve the dual problem which will require less computational effort. The classification of problems into the primal and dual categories is, however, arbitrary since if the maximization problem is defined as the primal then the minimization problem is its dual. It can be shown [13] that the optimal values of the basic variables of the primal can be obtained from the solution of the dual and that $(f_p)_{\min} = (f_d)_{\max}$. Thus if x_j is a basic variable in the primal problem, then it implies that the jth constraint of the dual problem is active and vice versa.

If the primal problem is stated in its standard form; namely with equality constraints

$$\begin{aligned} \textbf{minimize} \quad & f_p = c_1 x_1 + \ldots\ldots + c_n x_n = \mathbf{c^T x} && (n \text{ variables}) \\ \textbf{subject to} \quad & \sum_{j=1}^{n} a_{ij} x_j = b_i, \quad i = 1, \ldots, m\,, && (m \text{ constraints}) \\ & x_j \geq 0, \quad j = 1, \ldots, n\,, && (3.7.3) \end{aligned}$$

then the corresponding dual problem is

$$\begin{aligned} \textbf{maximize} \quad & f_d = b_1 \lambda_1 + \ldots\ldots + b_m \lambda_m = \mathbf{b^T \boldsymbol{\lambda}} && (m \text{ variables}) \\ \textbf{subject to} \quad & \sum_{i=1}^{m} a_{ij} \lambda_i \leq c_j, \quad j = 1, \ldots, n\,, && (n \text{ constraints}) \\ & && (3.7.4) \end{aligned}$$

with the variables λ_i being unrestricted in sign [11].

It should be noted that, in practice, it is rare for a LP problem to be solved either as a primal or as a dual problem. Most state-of-the-art LP software employ what is known as a primal-dual algorithm. This algorithm begins with a feasible solution to the dual problem that is successively improved by optimizing an associated restricted primal problem. The details of this algorithm are beyond the scope of this book and interested readers should consult Ref. [11].

Example 3.7.1

As an example of the simplex method for solving an LP problem via the dual formulation we use the portal frame problem formulated in Example 3.1.5 with a slightly different loading condition. The new loading condition is assumed to correspond to a 25% increase in the magnitude of the horizontal load while keeping the magnitude of the vertical load the same. The corresponding constraint equations have different right-hand sides than those given in Eqs. (3.5.4) through (3.5.9), namely

$$\begin{aligned} 4x_2 &\geq 1\,, \\ 2x_1 + 2x_2 &\geq 1\,, \\ x_1 + x_2 &\geq 1.25\,, \\ 2x_1 &\geq 1.25\,, \\ 2x_1 + 4x_2 &\geq 3.5\,, \\ 4x_1 + 2x_2 &\geq 3.5\ . \end{aligned} \qquad (3.7.5)$$

However, when put into the standard form, not only does the problem involve a total of 8 variables, but also a basic feasible solution to the problem is not immediately obvious. Because the objective function (3.1.25) involves only two variables x_1 and x_2 the solution of the dual problem may be more efficient. The dual problem is

$$\begin{aligned} \textbf{maximize} \quad & f_d = \lambda_1 + \lambda_2 + 1\frac{1}{4}\lambda_3 + 1\frac{1}{4}\lambda_4 + 3\frac{1}{2}\lambda_5 + 3\frac{1}{2}\lambda_6 && (3.7.7) \\ \textbf{subject to} \quad & 2\lambda_2 + \lambda_3 + 2\lambda_4 + 2\lambda_5 + 4\lambda_6 \le 2\,, \\ & 4\lambda_1 + 2\lambda_2 + \lambda_3 + 4\lambda_5 + 2\lambda_6 \le 1\,, && (3.7.8) \\ & \lambda_i \ge 0, \quad i = 1, \ldots, 6\ . \end{aligned}$$

Maximizing f_d is same as minimizing $-f_d$ and the process of converting the above linear problem to the standard form yields

$$\begin{aligned} \textbf{minimize} \quad & -f_d = -\lambda_1 - \lambda_2 - 1\frac{1}{4}\lambda_3 - 1\frac{1}{4}\lambda_4 - 3\frac{1}{2}\lambda_5 - 3\frac{1}{2}\lambda_6 && (3.7.9) \\ \textbf{subject to} \quad & 2\lambda_2 + \lambda_3 + 2\lambda_4 + 2\lambda_5 + 4\lambda_6 + \lambda_7 = 2\,, \\ & 4\lambda_1 + 2\lambda_2 + \lambda_3 + 4\lambda_5 + 2\lambda_6 + \lambda_8 = 1\,, && (3.7.10) \\ & \lambda_i \ge 0, \quad i = 1, \ldots, 8\,, \end{aligned}$$

with the basic feasible solution

$$\lambda_i = 0, \quad i = 1, \ldots, 6\,, \quad \text{and} \qquad \lambda_7 = 2, \quad \lambda_8 = 1\ .$$

We can begin with the initial simplex tableau with the basic variables cleared from the last equation which represents the objective function.

$$\left[\begin{array}{cccccccc} 0 & 2 & 1 & 2 & 2 & 4 & 1 & 0 \\ 4 & 2 & 1 & 0 & 4 & 2 & 0 & 1 \\ -- & -- & -- & -- & -- & -- & -- & -- \\ -1 & -1 & -5/4 & -5/4 & -7/2 & -7/2 & 0 & 0 \end{array}\right] \left\{\begin{array}{c} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \\ \lambda_7 \\ \lambda_8 \end{array}\right\} = \left\{\begin{array}{c} 2 \\ 1 \\ -- \\ 0 \end{array}\right\}\ . \tag{3.7.11}$$

Although we should perhaps be choosing fifth or sixth column for pivoting, since it has the largest negative value, pivoting on third column produces the same final answer with one less simplex tableau. Pivoting on element a_{23} we have

$$\left[\begin{array}{cccccccc} -4 & 0 & 0 & 2 & -2 & 2 & 1 & -1 \\ 4 & 2 & 1 & 0 & 4 & 2 & 0 & 1 \\ -- & -- & -- & -- & -- & -- & -- & -- \\ 4 & 3/2 & 0 & -5/4 & 3/2 & -1 & 0 & 5/4 \end{array}\right] \left\{\begin{array}{c} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \\ \lambda_7 \\ \lambda_8 \end{array}\right\} = \left\{\begin{array}{c} 1 \\ 1 \\ -- \\ 5/4 \end{array}\right\}\ . \tag{3.7.12}$$

Because of the presence of negative terms in the last equation, it is clear that the objective function can still be decreased further. Pivoting on element a_{14} we obtain

$$\left[\begin{array}{cccccccc} -2 & 0 & 0 & 1 & -1 & 1 & 1/2 & -1/2 \\ 4 & 2 & 1 & 0 & 4 & 2 & 0 & 1 \\ -- & -- & -- & -- & -- & -- & -- & -- \\ 3/2 & 3/2 & 0 & 0 & 1/4 & 1/4 & 5/8 & 5/8 \end{array}\right] \left\{\begin{array}{c} \lambda_1 \\ \lambda_2 \\ \lambda_3 \\ \lambda_4 \\ \lambda_5 \\ \lambda_6 \\ \lambda_7 \\ \lambda_8 \end{array}\right\} = \left\{\begin{array}{c} 1/2 \\ 1 \\ -- \\ 15/8 \end{array}\right\} . \quad (3.7.13)$$

Hence we conclude that $(f_d)_{\min} = -15/8$ or $(f_d)_{\max} = (f_p)_{\min} = 15/8$ with the solution

$$\lambda_1 = \lambda_2 = \lambda_5 = \lambda_6 = \lambda_7 = \lambda_8 = 0, \quad \text{and} \quad \lambda_3 = 1, \quad \lambda_4 = 1/2 \, . \quad (3.7.14)$$

The non-zero λ's indicate that the active constraints in the primal problem are the third and fourth, namely

$$2x_1 = 1.25, \quad \text{and} \quad x_1 + x_2 = 1.25 \, , \quad (3.7.15)$$

Solution of Eqs. (3.7.15) yields $x_1 = x_2 = 5/8$. • • •

In closing this section, it is interesting to point out that the dual variables can be interpreted as the prices of the constraints. For a given variation on the right hand side $\mathbf{b}$ of the constraint relations of Eq. (3.7.5), the change in the optimum value of the objective function can be determined from

$$\Delta f^* = \boldsymbol{\lambda}^T \Delta \mathbf{b} \, . \quad (3.7.16)$$

For Eq. (3.7.16) to hold, however, the changes in the $\mathbf{b}$ vector must be such that it does not result in a change in the active constraint set. The dual problem can also be viewed as one of maximization of a *profit* subject to limitations on availability of *resources*. It is clear then that the non-negative dual variables can be interpreted as increased costs which would ensue from a violation of given constraints on resource availabilities. Similarly a primal problem can be viewed as one of minimization of total *cost* while satisfying *demand*. The full significance of dual variables, however, can be brought out more clearly only in the context of the Kuhn-Tucker conditions and the sensitivity of the optimum solutions to changes in design parameters which will be discussed in Chapter 5. The following example demonstrates the use of dual variables to find the sensitivity of the optimal solution to a change in a problem parameter.

Example 3.7.2

Consider the portal frame design problem solved in Example 3.7.1 using dual variables. We will determine the change in the value of the optimum objective function $f^* = 1.875$ corresponding to a 25% reduction in the value of the horizontal force,

keeping the vertical force at p. These loads correspond to the problem formulated in Example 3.1.5 and solved graphically in Example 3.4.1 .

From Eqs. (3.7.5) and (3.1.26) through (3.1.31) the change in the right-hand side is $\Delta b_3 = \Delta b_4 = -\frac{1}{4}$, and $\Delta b_5 = \Delta b_6 = -\frac{1}{2}$. Using the values of the dual variables from Example 3.7.1 in Eq. (3.7.15) we obtain

$$\Delta f^* = -\left(\frac{1}{4}\right) 1 + -\left(\frac{1}{4}\right)\left(\frac{1}{2}\right) = -0.375 \ .$$

Therefore the optimum value of the objective function under this new loading configuration would be $f^* = 1.5$, of course, assuming that the active constraints (the ones associated with non-zero dual variables) remain active. Fortunately, that assumption is correct for the present example. However, beside the two constraints that are active initially there are two more constraints which become active at the new design point (see Fig. 3.4.1). Any reduction larger than 25% in the value of the horizontal load would have caused a change in the active constraint set and resulted in an incorrect answer.

We, therefore, emphasize the fact that in applying Eq. (3.7.15) one has to be cautious not to perturb the design parameter to an extent that the active constraint set changes. This is generally achieved by limiting the parameter perturbations to be small. However, if we had used the design in Example 3.4.1 as our nominal design, no matter how small the perturbation of the magnitude of the horizontal force, the active constraint set would have changed. This is due to the redundancy of the constraints at the optimal solution of Example 3.4.1. • • •

3.8 An Interior Method — Karmarkar's Algorithm

In using the simplex algorithm discussed in section 3.6, we operate entirely along the boundaries of the polytope in $\mathbf{R}^n$ moving from one extreme point (vertex) to another following the shortest path between them, an edge of the polytope. Of all the possible vertices adjacent to the one at which we start, the selection of the next vertex is based on the maximum reduction in the objective function. With these basic premises, the simplex algorithm is only a systematic approach for identifying and examining candidate solutions to the LP problem. The number of operations needed for convergence grows exponentially with the number of variables. In the worst case, the number of operations for convergence for an n variable problem with a set of s constraints can be $s!/n!(s-n)!$. However, it is possible to choose a move direction different from an edge of the polytope, be consistent with the constraint relations, and attain larger gains in the objective function. Although such a choice can lead to a rapid descent toward the optimal vertex, it will do so through intermediate points which are not vertices.

Interior methods of solving LP problems have drawn serious attention only since the dramatic introduction of Karmarkar's algorithm [14] by AT&T Bell Laboratories. This new algorithm was originally claimed to be 50 times faster than the simplex

method. Since then, much work has been invested in improvements and extensions of Karmarkar's algorithm. Developments include demonstration of how dual solutions can be generated during the course of this algorithm [15], and extension of Karmarkar's algorithm to treat upper and lower bounds more efficiently [16] by eliminating the slack variables which are commonly used for such bounds in the Simplex algorithm.

Because some of the recent developments of the algorithm are mathematically involved and beyond the scope of this book, only a general outline of Karmarkar's algorithm are presented in the following sections. At this point we would like to warn the reader that the tools used in the algorithm were originally introduced for minimization of constrained and unconstrained nonlinear functions which are covered in Chapters 4 and 5. Therefore, the reader is advised to read these chapters before proceeding to the next section.

3.8.1 *Direction of Move*

The direction of maximum reduction in the objective function is the direction of steepest descent, which is the direction of the negative of the gradient of the objective function ∇f (see section 4.2.2). For an LP problem posed in its standard form, see Eq. (3.5.1), the gradient direction is,

$$\nabla f = \mathbf{c} \ . \tag{3.8.1}$$

Although we are not limiting the move direction to be an edge of the polytope formed by the constraint surfaces, for an LP problem the move direction cannot be selected simply as the negative of the gradient direction. The direction must be chosen such that the move leads to a point in the feasible region. This can be achieved by using the projection matrix $\mathbf{P}$

$$\mathbf{P} = \mathbf{I} - \mathbf{N}(\mathbf{N}^T\mathbf{N})^{-1}\mathbf{N}^T \, , \tag{3.8.2}$$

derived in section 5.3, where the columns of the matrix $\mathbf{N}$ correspond to the gradient of the constraint equations. Since the constraints are linear functions of the variables, we have $\mathbf{N} = \mathbf{A}^T$. Operating on the gradient vector $-\mathbf{c}$, $\mathbf{P}$ projects the steepest descent direction onto the nullspace of the matrix $\mathbf{A}$. That is, if we start with an initial design point $\mathbf{x}_0$ which satisfies the constraint equation $\mathbf{A}\mathbf{x}_0 = \mathbf{b}$, and move in a direction $-\mathbf{P}\mathbf{c}$ we will remain in the subspace defined by that constraint equation. Note that in numerical application of this projection the matrix product $\mathbf{A}\mathbf{A}^T$ may not actually be inverted, but rather the linear system $\mathbf{A}\mathbf{A}^T\mathbf{y} = \mathbf{A}\mathbf{c}$ may be solved and then the projected gradient may be calculated by using $\mathbf{P}\mathbf{c} = \mathbf{c} - \mathbf{A}^T\mathbf{y}$. A more efficient and better conditioned procedure based on QR factorization of the matrix $\mathbf{A}$ for the solution of the projection matrix is described in section 5.5 . The following simple example by Strang from reference [17] illustrates graphically the move direction for a three dimensional design space.

Example 3.8.1

Consider the following minimization problem in three design variables,

$$\textbf{minimize} \qquad f = -x_1 - 2x_2 - 3x_3 \tag{3.8.3}$$

$$\textbf{subject to} \qquad x_1 + x_2 + x_3 = 1\,, \tag{3.8.4}$$

$$\mathbf{x} \geq \mathbf{0}\ . \tag{3.8.5}$$

Starting at an initial point $\mathbf{x}^{(0)} = (1/3, 1/3, 1/3)^T$ determine the direction of move.

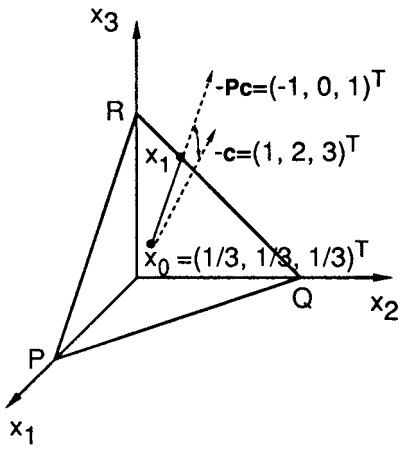

Figure 3.8.1 Design space and move direction.

The design space and the constraint surface for the problem are shown in Figure (3.8.1). The direction corresponding to the negative of the gradient vector is marked as $-\mathbf{c}$. The projection matrix for the problem can be obtained from Eq. (3.8.2) where $\mathbf{A} = [1 \quad 1 \quad 1]$. The system $\mathbf{A}\mathbf{A}^T\mathbf{y} = \mathbf{A}\mathbf{c}$ produces a scalar for $\mathbf{y}$,

$$\{1 \quad 1 \quad 1\} \begin{Bmatrix} 1 \\ 1 \\ 1 \end{Bmatrix} y = \{1 \quad 1 \quad 1\} \begin{Bmatrix} -1 \\ -2 \\ -3 \end{Bmatrix}\,, \tag{3.8.6}$$

$$y = -2\ .$$

The projected direction $\mathbf{Pc}$ is then given by

$$\mathbf{Pc} = \mathbf{c} - y\mathbf{A}^T\,, \tag{3.8.7}$$

$$\mathbf{Pc} = \begin{Bmatrix} -1 \\ -2 \\ -3 \end{Bmatrix} - \begin{Bmatrix} -2 \\ -2 \\ -2 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 0 \\ -1 \end{Bmatrix}\ . \tag{3.8.8}$$

Moving in a direction $-\mathbf{Pc}$ guarantees maximum reduction in the objective function while remaining in the plane PQR formed by the constraint equation. The minimum value of the objective function for this problem is achieved at the vertex R which, clearly, can not be reached in one iteration. Therefore, the move has to be terminated before the non-negativity requirement is violated (which is at $\mathbf{x}^{(1)} = (2/3, 1/3, 0)^T$),

and the procedure has to be repeated until a reasonable convergence to the minimum point is achieved. • • •

In the preceding example no explanation is provided for the selection of the initial design point, and for the distance travelled in the chosen direction. Karmarkar [14] stops the move before hitting the polytope boundary, say at $\mathbf{x}^{(1)} = (19/30, 1/3, 1/30)^T$ in the previous example, so that there will be room left to move in the next iteration. That is, starting either at the polytope or close to it increases the chances of hitting another boundary before making real gains in the objective function. The solution to this difficulty is accomplished by transforming the design space discussed in the next section.

3.8.2 *Transformation of Coordinates*

In order to focus on the ideas which are important for his algorithm, Karmarkar [14] makes several assumptions with respect to the form of the LP problem. In his canonical representation, the LP problem takes the following form,

$$\textbf{minimize} \qquad f = \mathbf{c}^T\hat{\mathbf{x}} \tag{3.8.9}$$

$$\textbf{subject to} \qquad \mathbf{A}\hat{\mathbf{x}} = 0\,, \tag{3.8.10}$$

$$\mathbf{e}^T\hat{\mathbf{x}} = 1\,, \tag{3.8.11}$$

$$\hat{\mathbf{x}} \geq \mathbf{0}\,, \tag{3.8.12}$$

where $\mathbf{e}$ is a $1 \times n$ vector, $\mathbf{e} = (1, \ldots, 1)^T$. The variable $\hat{\mathbf{x}}$ represents the transformed coordinate such that the initial point is the center, $\hat{\mathbf{x}}^{(0)} = \mathbf{e}/n$, of a unit simplex, and is a feasible point, $\mathbf{A}\mathbf{x}^{(0)} = \mathbf{0}$. A simplex is a generalization to n dimensions of a 2-dimensional triangle and 3-dimensional tetrahedron. A unit simplex has edges of unit length along each of the coordinate directions. Karmarkar also assumes that $\mathbf{c}^T\hat{\mathbf{x}} \geq 0$ for every point that belongs to the simplex, and the target minimum value of the objective function is zero. Conversion of the standard form of an LP problem into this new canonical form can be achieved through a series of operations that involve combining the primal and dual forms of the standard formulation, introducing of slack and artificial variables, and transforming coordinates. The combination of the primal and dual formulations is needed to accommodate the assumption that the target minimum value of the objective function be zero. Details of the formation of this new canonical form is provided in Ref. [14]. In this section we will demonstrate the coordinate transformation which is referred as *projective rescaling transformation*. This is the same transformation that helps to create room for move as we proceed from one iteration to another.

Consider an arbitrary initial point $\mathbf{x}^{(a)}$ in the design space, and let

$$\mathbf{D}_x = \text{Diag } (x_1^{(a)}, \ldots, x_n^{(a)})\ . \tag{3.8.13}$$

The transformation, T_x, used by Karmarkar maps each facet of the simplex given by $x_i = 0$ onto the corresponding facet $\hat{x}_i = 0$ in the transformed space, and is given by

$$\hat{\mathbf{x}} = \frac{1}{\mathbf{e}^T\mathbf{D}_x^{-1}\mathbf{x}}\mathbf{D}_x^{-1}\mathbf{x}\ . \tag{3.8.14}$$

While mapping the unit simplex onto itself, this transformation moves the point $\mathbf{x}^{(a)}$ to the center of the simplex, $\hat{\mathbf{x}}^{(0)} = (1/n)\mathbf{e}$. Karmarkar showed that repeated application of this transformation, in the worst case, leads to convergence to the optimal corner in less than $\mathcal{O}(n^{\frac{7}{2}})$ arithmetic operations.

Karmarkar's transformation is nonlinear and a simpler form of this transformation has been suggested. A linear transformation,

$$\hat{\mathbf{x}} = \mathbf{D}_x^{-1}\mathbf{x}\,, \tag{3.8.15}$$

has been shown to perform as well as Karmarkar's algorithm in practice and to converge in theory [18].

3.8.3 *Move Distance*

Following the transformation, Karmarkar optimizes the transformed objective function over an inscribed sphere of radius $r = 1/(\sqrt{n(n-1)}$ centered at $\hat{\mathbf{x}}^{(0)}$. This is the largest radius sphere that is contained inside the simplex. For the three dimensional design space of Example 3.8.1, for example, where there is one constraint surface, the 'sphere' is a circle in the plane of the constraint equation. In practice, the step length along the projected direction used by Karmarkar is a fraction, α, of the radius. Thus, the new point at the end of the move is given by

$$\mathbf{x}^{(k+1)} = \hat{\mathbf{x}}^{(k)} - \alpha r^{(k)}\mathbf{P}\mathbf{c}^{(k)}\,, \tag{3.8.16}$$

where $0 < \alpha < 1$. A typical value of α used by Karmarkar is 1/4.

During the course of the algorithm the optimality of the solution is checked periodically by converting the interior solution to an extreme point solution at the closest vertex. If the extreme point solution is better than the current interior, then, it is tested for optimality.

3.9 Integer Linear Programming

Solution techniques for the LP problems considered so far have been developed under the assumption that the design variables are positive and continuously-valued; they can thus assume any value between their lower and upper bounds. In certain design situations, some or all of the variables of a LP problem are restricted to take discrete values. That is, the standard form of the LP problem of Eq. (3.5.1-3.5.3) takes the form

$$\begin{aligned} \textbf{minimize} \quad & f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} \\ \textbf{such that} \quad & \mathbf{A}\mathbf{x} = \mathbf{b}, \\ & x_i \in X_i = \{d_{i1}, d_{i2}, \ldots, d_{il}\}, \qquad i \in I_d\,, \end{aligned} \tag{3.9.1}$$

where I_d is the set of design variables that can take only discrete values, and X_i is the set of allowable discrete values. Design variables such as cross-sectional areas of

trusses and ply thicknesses of laminated composite plates often fall in this category. Those problems with discrete-valued design variables are called discrete programming problems.

In general, a discrete programming problem can be converted to a form where design variables can assume only integer values. This conversion can be achieved by having the design variable x_i to represent the index j of the $d_{ij}, j = 1, \ldots, l$, Eq. (3.9.1). If the values in the discrete set are uniformly spaced, it is possible to scale the set to form a set of integer values only. The problem is then called an *integer linear programming* (ILP) problem,

$$\begin{array}{ll} \textbf{minimize} & f(\mathbf{x}) = \mathbf{c}_1^T\mathbf{x} + \mathbf{c}_2^T\mathbf{y} \\ \textbf{such that} & \mathbf{A}_1\mathbf{x} + \mathbf{A}_2\mathbf{y} = \mathbf{b}, \\ & x_i \geq 0 \text{ integer}\,, \\ & y_j \geq 0\ . \end{array} \tag{3.9.2}$$

This form, where certain design variables are allowed to be continuous, is referred to as *mixed integer linear programming* (MILP) problem. Problems where all variables are integer are called pure ILP problems or in short ILP problems. It is also common to have problems where design variables are used to indicate a 0/1 type decision making situation. Such problems are referred to as *zero/one* or *binary* ILP problems. For example, a truss design problem where the presence of a particular member or the lack of it is represented by a binary variable falls into this category. Any ILP problem with an upper bound on the design variable x_i of $2^K - 1$ can be posed as binary ILP problem by replacing the variable with K binary variables $x_{i1}, \ldots, x_{iK}$ such that

$$x_i = x_{i1} + 2x_{i2} + \ldots + 2^{K-1}x_{iK}\ . \tag{3.9.3}$$

It is also possible to convert the linear discrete programming problem to a binary ILP by using binary variables ($x_{ij} \in \{0, 1\}, j = 1, \ldots, l$) such that

$$x_i = d_{i1}x_{i1} + d_{i2}x_{i2} + \cdots + d_{il}x_{il}\,, \tag{3.9.4}$$

$$\text{and} \qquad x_{i1} + x_{i2} + \cdots + x_{il} = 1\ . \tag{3.9.5}$$

Most of the following discussion assumes problems to be pure ILP.

A practical approach to solving ILP problems is to round-off the optimum values of the variables, obtained by assuming them to be continuous, to the nearest acceptable integer value. For problems with n design variables there are 2^n possible rounded-off designs, and the problem of choosing the best one is formidable for large n. Furthermore, for some problems the optimum design may not even be one of these rounded-off designs, and for others none of the rounded-off designs may be feasible. A more systematic way of trying possible combinations of variables that will satisfy the requirements of a given problem can be explained by using the *enumeration tree* example of Garfinkel and Nemhauser [19].

Example 3.9.1

Consider the binary ILP problem of choosing a combination of five variables such that the following summation is satisfied

$$f = \sum_{i=1}^{5} ix_i = 5 \ .$$

A decision tree representing the progression of solution of this problem is composed of nodes and branches that represent the solutions and the combinations of variables that lead the those solutions, respectively (Figure 3.9.1). The top node of the tree corresponds to a solution which all the variables are turned off ($x_i = 0, i = 1, \ldots, 5$) with a function value of $f = 0$. Branching off from this solution are two paths corresponding to the two alternatives for the first variable. The branch which has $x_1 = 1$ has a function value of $f = 1$ and tolerates turning additional variables on without running into the risk of exceeding the required function value of 5. Of course the other branch is same as the initial solution, and can be branched further. Next, these two nodes are branched by considering the on and off alternatives for the second variable. The node arrived by taking $x_1 = x_2 = 1$ has $f = 3$ and is terminated as indicated by a vertical line. Such a vertex is said to be *fathomed*, because further branching would mean adding a number that would cause f to exceed its required value of 5. The other three vertices are said to be live, and can be branched further by considering the alternatives for the remaining variables in a sequential manner until either the created nodes are fathomed or the branches arrive at feasible solutions to the problem.

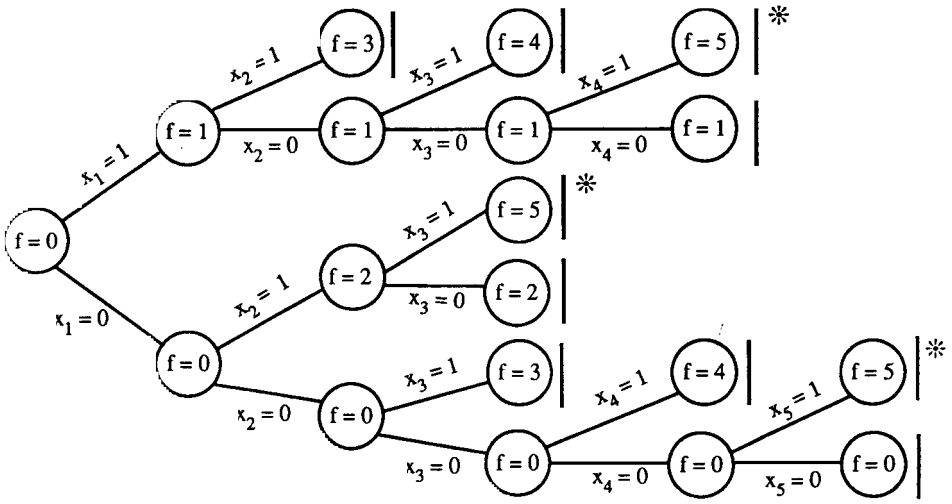

Figure 3.9.1 Enumeration tree for binary ILP problem of $f = \sum_{i=1}^{5} ix_i = 5$.

For the present problem, after considering 19 possible combinations of variables, we identified 3 feasible solutions which are marked by an asterisk. This is a 40% reduction in the total number of possible trials, namely $2^5 = 32$, needed to identify all feasible solutions. For a structural design problem in which trials with different combinations of variables would possibly require expensive analysis an enumeration tree can yield substantial savings. • • •

3.9.1 *Branch-and-Bound Algorithm*

The basic concept behind the enumeration technique forms the basis for this powerful algorithm suitable for MILP problems as well as nonlinear mixed integer problems [20,21]. The original algorithm developed by Land and Doig [22] relies on calculating upper and lower bounds on the objective function so that nodes that result in designs with objective functions outside the bounds can be fathomed and, therefore, the number of analyses required can be cut back. Consider the mixed ILP problem of Eq. (3.9.4). The first step of the algorithm is to solve the LP problem obtained from the MILP problem by assuming the variables to be continuous valued. If all the x variables for the resulting solution have integer values, there is no need to continue, the problem is solved. Suppose several of the variables assume noninteger values and the objective function value is f_1. The f_1 value will form a lower bound $f_L = f_1$ for the MILP since imposing conditions that require any of the noninteger valued variables to take integer values can only cause the objective function to increase. This initial problem is labeled as LP-1 and is placed in the top node of the enumeration tree as shown in Figure (3.9.2). For the purpose of illustration, it is assumed that only two variables x_k and x_{k+1} violate the integer requirement with $x_k = 4.3$ and $x_{k+1} = 2.8$.

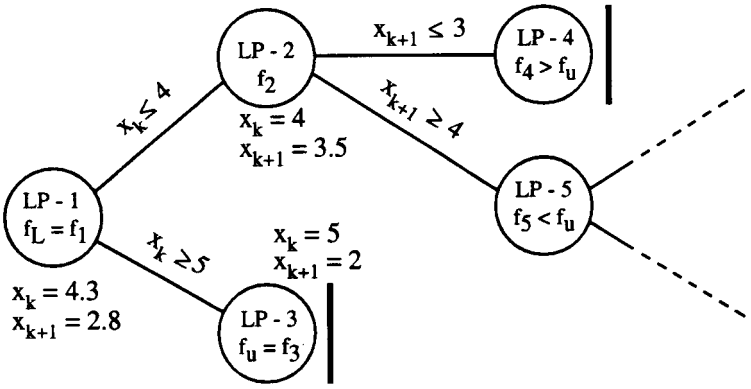

Figure 3.9.2 Branch-and-bound decision tree for ILP problems.

The second step of the algorithm is to branch from the node into two new LP problems by adding a new constraint to the LP-1 that would involve only one of the noninteger variables, say x_k. One of the problems, LP-2, will require the value of the branched variable, x_k to be less than or equal to the largest integer smaller than x_k,

and the other, LP-3, will have a constraint that x_k is larger than the smallest integer larger than x_k. As will be demonstrated later in Example 3.9.2, these two problems actually do branch the feasible design space of the LP-1 into two segments. There are several possibilities for the solution of these two new problems. One of these possibilities is to have no feasible solution for the new problem. In that case the new node will be fathomed. Another possibility is to reach an all integer feasible solution (see LP-3 of Figure 3.9.2) in which case the node will again be fathomed but the value of the objective function will become an upper bound f_U for the MILP problem. That is, beyond this solution point, any node that has an LP solution with a larger value of the objective function will be fathomed, and only those solutions that have the potential of producing an objective function between f_L and f_U will be pursued. If there are no solutions with an objective function smaller than f_U, then the node is an optimum solution. If there are other solutions with an objective function smaller than f_U, they may still include noninteger valued variables (LP-2 of Figure 3.9.2), and are labeled as *live* nodes. Live nodes are then branched again by considering one of the remaining noninteger values and resulting solutions are analyzed until all the nodes are fathomed.

Example 3.9.2

Consider the portal frame problem problem of Example 3.1.5 (see Eqs. (3.1.25) through (3.1.31)) with the requirement that $x_i \in \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}, i = 1, 2$. We rescale the design variables by a factor of 5 to pose the problem as an integer linear programming problem,

$$\begin{aligned} \textbf{minimize} \quad & f = \frac{1}{5}(2x_1 + x_2) \\ \textbf{such that} \quad & x_2 \geq 1.25\,, \\ & x_1 + x_2 \geq 2.5\,, \\ & x_1 + x_2 \geq 5\,, \\ & x_1 \geq 2.5\,, \\ & x_1 + 2x_2 \geq 7.5\,, \\ & 2x_1 + x_2 \geq 7.5\,, \\ & x_i \geq 0 \text{ integer}, \qquad i = 1, 2\ . \end{aligned}$$

Graphical solution of this scaled problem (presented in Example 3.4.1 without the integer design variable requirement before scaling) is

$$x_1 = x_2 = 2.5, \qquad f = 7.5\,,$$

and forms a lower bound for the objective function, $f_L = 7.5$. That is, the optimal integer solution cannot have an objective function smaller than $f_L = 7.5$. Next, we choose x_1 and investigate solutions for which $x_1 \leq 2$ and $x_1 \geq 3$ by forming two new LP's by adding each one of these constraints to the original set of constraints. Since the original set has a constraint that requires $x_1 \geq 2.5$, the first LP problem with $x_1 \leq 2$ has no solution. The solution of the second LP is shown graphically in Figure (3.9.3). The active constraints at the optimum are, $x_1 \geq 3$ and $x_1 + 2x_2 \geq 7.5$, and the solution is,

$$x_1 = 3, \qquad x_2 = 2.25, \qquad f = 8.25\ .$$

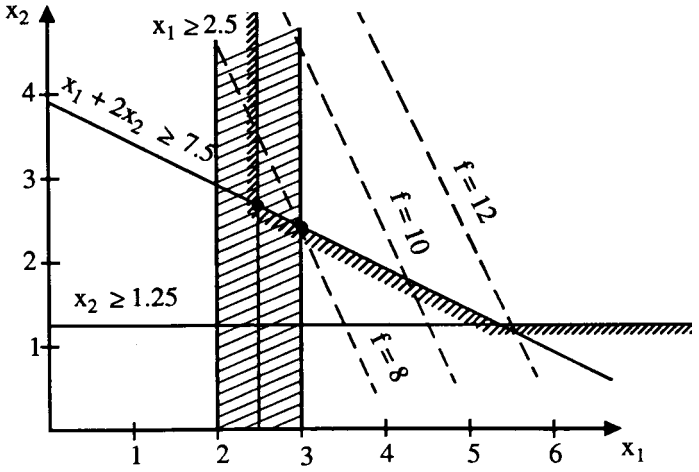

Figure 3.9.3 Branch-and-bound solution for $x_1 \leq 2$ and $x_1 \geq 3$ of Example 3.9.2 .

Since x_2 is still non integer, we create two more LP's, this time by imposing $x_2 \leq 2$ and $x_2 \geq 3$, respectively. Graphical solutions of the new LP's are shown in Figure (3.9.4). The solution for the case $x_2 \geq 3$ is at the vertex $x_1 = 3$ and $x_2 = 3$, and is a feasible solution for the integer problem with an objective function value of $f = 9$. This value of the objective function, therefore, establishes an upper bound, $f_U = 9$ for the problem. The solution for the case $x_2 \leq 2$, on the other hand is at the intersection of $x_2 = 2$ and $x_1 + 2x_2 = 5$ leading to

$$x_1 = 3.5, \qquad x_2 = 2, \quad \text{and} \quad f = 9 \ .$$

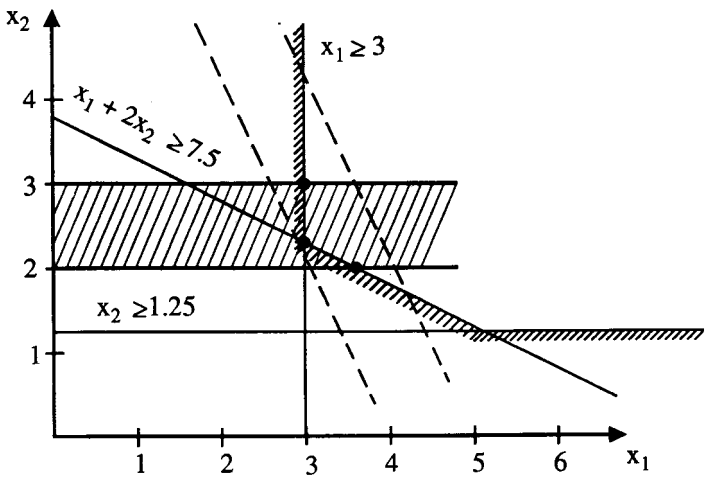

Figure 3.9.4 Branch-and-bound solution for $x_2 \leq 2$ and $x_2 \geq 3$ of Example 3.9.2 .

This solution is not discrete and can be interrogated further by branching on x_1 (that is creating new LP's by adding $x_1 \leq 3$ and $x_1 \geq 4$). However, since its objective function is equal to the upper bound, we cannot improve the objective function any further. To do so would necessitate introducing a further constraint which could only increase the objective function. Therefore, the optimal solution is the one with $x_1 = x_2 = 3$, and $f = 9$. $\bullet\bullet\bullet$

As can be observed from the example, performance of the Branch-and-Bound algorithm relies heavily on the choice of noninteger variable to be used for branching, and the selection of node to be branched. If a selected node and branching variable leads to an upper bound close to the objective function of the LP-1 early in the enumeration scheme, then substantial computational savings can be obtained because of the elimination of branches that would not be capable of generating solutions lower than the upper bound. A rule of thumb for choosing the noninteger variable to be branched is to take the variable with the largest fraction. For the selection of the node to be branched, we choose, among all the live nodes, the LP problem which has the smallest value of the objective function; that node is most likely to generate a feasible design with a tighter upper bound.

Branch-and-Bound is only one of the algorithms for the solution of ILP or MILP problems. However, because of its simplicity it is incorporated into many commercially available computer programs [23, 24]. There are a number of other techniques which are capable of handling general discrete-valued problems (see, for example, Ref. [25]). Some of these algorithms are good not only for ILP problems but also for NLP problems with integer variables. Particularly, methods based on probabilistic search algorithms are emerging for many applications, including structural design applications, that involve linear and nonlinear programming problems. Two of such techniques, namely simulated annealing and genetic algorithms, are discussed in Chapter 4. Another approach, which is based on an extension of the penalty function approach for constrained NLP problems, is presented in Chapter 5. Finally, the use of dual variables (which are presented to be useful as prices of constraints in section 7.3) in ILP problems are discussed in Chapter 9.

One of the interesting design applications of the ILP was introduced by Haftka and Walsh [26] for the stacking sequence design of laminated composite plates for improved buckling response. Since the formulation of this problem involves material introduced in Chapter 11, discussion and demonstration of this application is presented in that chapter.

3.10 Exercises

1. Estimate the limit load for the three bar truss example 3.1.2 using a graphical approach. Verify your solution using the simplex method.

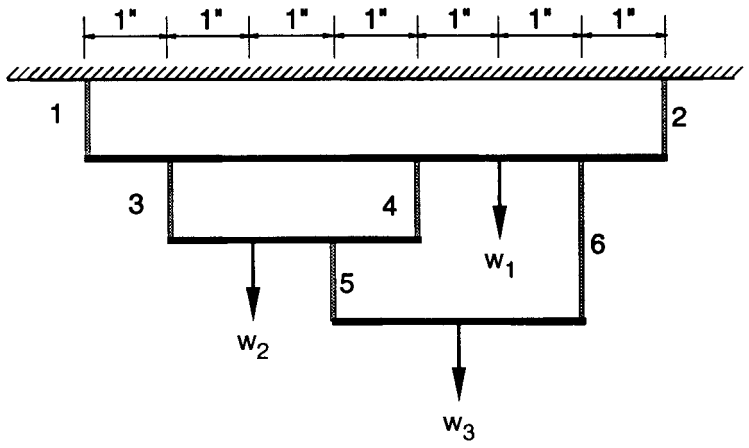

Figure 3.10.1 Platform support system

2. Consider the platform support system shown in Figure 3.10.1 in which cables 1 and 2 can support loads up to 400 lb each; cables 3 and 4 up to 150 lb each and cables 5 and 6 up to 75 lb each. Neglect the weight of the platforms and cables, and assume the weights $w_1, w_2,$ and w_3 at the positions indicated in the figure. Also neglect the bending failure of the platforms. Using linear programming determine the the maximum total load that the system can support.

3. Solve the limit design problem for the truss of Figure 3.1.4 using the simplex algorithm. Assume $A_{13} = A_{24} = A_{34}$, $A_{14} = A_{23}$, and use appropriate nondimensionalization.

4. Using the method of virtual displacements verify that the collapse mechanisms for the portal frame of Figure 3.1.6 lead to Eqs. (3.1.26) through (3.1.31) in terms of the nondimensional variables x_1 and x_2.

5. The single bay, two story portal frame shown in Figure (3.10.2) is subjected to a single loading condition consisting of 4 concentrated loads as shown. Following Example 3.1.5 formulate the LP problem for the minimum weight design of the frame against plastic collapse.

6. Consider the continuous prestressed concrete beam shown in Figure (3.10.3),

a) Verify that the equivalent uniformly distributed upward force exerted on the concrete beam by a prestressing cable with a force f and a parabolic profile defined by eccentricities $y_1, y_2,$ and y_3 at the three points $x = 0, x = l/2,$ and $x = l$ respectively is given by

$$q = \frac{4f}{l^2}(y_3 - 2y_2 + y_1) \ .$$

b) The beam in the figure is subjected to two loading conditions: the first consisting of a dead load of 1 kip/ft together with an equivalent load due to a parabolic

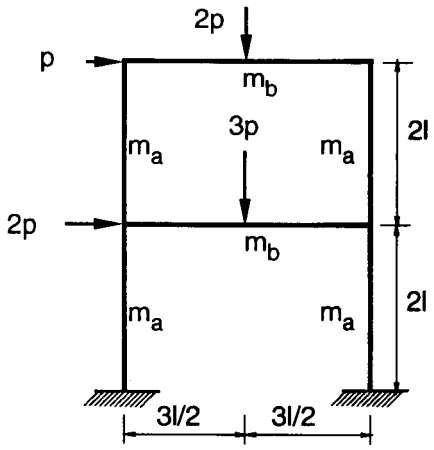

Figure 3.10.2 Two story portal frame

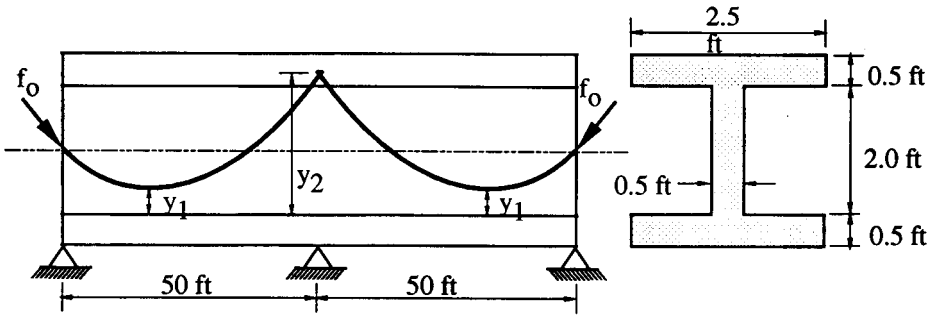

Figure 3.10.3 A continuous prestressed concrete beam

prestressing cable with a force f, and the second due to an additional live load of 2.5 kips/ft in service. It is assumed, however, that in service a 15% loss of prestressing force is to be expected. Formulate the LP problem for the minimum cost design of beam assuming $f, y_1,$ and y_2 as design variables. Assume the allowable stress for the two loading conditions to be $\sigma_1^u = 200$ psi, $\sigma_1^l = -3000$ psi, $\sigma_2^u = 0$ psi, $\sigma_2^l = -2000$ psi and the upper and lower bound limits on the eccentricities y_1 and y_2 to be $0.4ft \leq y_i \leq 2.6ft,\ i = 1, 2.$

c) Solve the LP problem by the simplex algorithm and obtain the solution for the minimum prestressing force and the tendon profile.

7. Consider the statically determinate truss of Figure 3.3.1 and its minimum weight design formulation as described by Eqs. (3.3.9) through (3.3.13). Use the linearization scheme implied by Eqs. (3.3.2) through (3.3.5) to formulate the LP problem for m=3. Solve the LP by the simplex algorithm and compare the approximate solution with

the graphical or an exact solution to the problem.

8. Use Branch-and-Bound algorithm to solve the limit design problem of Exercise 3 by assuming the cross-sections of the members to take values from the following sets

a) $\{0.0, 0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0\}$.

b) $\{0.0, 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1\}$.

3.11 References

[1] Charnes, A. and Greenberg, H. J., "Plastic Collapse and Linear Programming," Bull. Am. Math. Soc., 57, 480, 1951.

[2] Calladine, C.R., Engineering Plasticity. Pergamon Press, 1969.

[3] Cohn, M.Z., Ghosh, S.K. and Parimi, S.R., "Unified Approach to Theory of Plastic Structures," Journal of the EM Division, 98 (EM5), pp. 1133–1158, 1972.

[4] Neal, B. G., The Plastic Methods of Structural Analysis, 3rd edition, Chapman and Hall Ltd., London, 1977.

[5] Zeman, P. and Irvine, H. M., Plastic Design, An Imposed Hinge–Rotation Approach, Allen and Unwin, Boston, 1986.

[6] Massonet, C.E. and Save, M.A., Plastic Analysis and Design, Beams and Frames, Vol. 1. Blaisdell Publishing Co., 1965.

[7] Lin, T.Y. and Burns, N.H., Design of Prestressed Concrete Structures, 3rd ed. John Wiley and Sons, New York, 1981.

[8] Parme, A.L. and Paris, G.H., "Designing for Continuity in Prestressed Concrete Structures," J. Am. Concr. Inst., 23 (1), pp. 45–64, 1951.

[9] Morris, D., "Prestressed Concrete Design by Linear Programming," J. Struct. Div., 104 (ST3), pp. 439–452, 1978.

[10] Kirsch, U., "Optimum Design of Prestressed Beams," Computers and Structures 2, pp. 573–583, 1972.

[11] Luenberger, D. G., Introduction to Linear and Nonlinear Programming, Addison-Wesley, Reading, Mass., 1973.

[12] Majid, K.I., Nonlinear Structures, London, Butterworths, 1972.

[13] Dantzig, G., Linear Programming and Extensions, Princeton University Press, Princeton, NJ, 1963.

[14] Karmarkar, N., "A New Polynomial–Time Algorithm for Linear Programming," Combinatorica, 4 (4), pp. 373–395, 1984.

[15] Todd, M. J. and Burrell, B. P., "An Extension of Karmarkar's Algorithm for Linear Programming Using Dual Variables," Algorithmica, 1, pp. 409–424, 1986.

[16] Rinaldi, G., "A Projective Method for Linear Programming with Box–type Constraints," Algorithmica, 1, pp. 517–527, 1986.

[17] Strang, G., "Karmarkar's Algorithm and its Place in Applied Mathematics," The Mathematical Intelligencer, 9, 2, pp. 4–10, 1987.

[18] Vanderbei, R. F., Meketon, M. S., and Freedman, B. A., "A Modification of Karmarkar's Linear Programming Algorithm," Algorithmica, 1, pp. 395–407, 1986.

[19] Garfinkel, R. S., and Nemhauser, G. L., Integer Programming, John Wiley & Sons, Inc., New York, 1972.

[20] Lawler, E. L., and Wood, D. E., "Branch-and-Bound Methods—A Survey," Operations research, 14, pp. 699–719, 1966.

[21] Tomlin, J. A., "Branch-and-Bound Methods for Integer and Non-convex Programming," in Integer and Nonlinear Programming, J. Abadie (ed.), pp. 437–450, Elsevier Publishing Co., New York, 1970.

[22] Land, A. H., and Doig, A. G., "An Automatic Method for Solving Discrete Programming Problems," Econometrica, 28, pp. 497–520, 1960.

[23] Johnson, E. L., and Powell, S., "Integer Programming Codes," in Design and Implementation of Optimization Software, Greenberg, H. J. (ed.), pp. 225–240, 1978.

[24] Schrage, L., Linear, Integer, and Quadratic Programming with LINDO, 4th Edition, The Scientific Press, Redwood City CA., 1989.

[25] Kovács, L. B., Combinatorial Methods of Discrete Programming, Mathematical Methods of Operations Research Series, Vol. 2, Akadémiai Kiadó, Budapest, 1980.

[26] Haftka, R. T., and Walsh, J. L., "Stacking-sequence Optimization for Buckling of Laminated Plates by Integer Programming," AIAA J. (in press).