CHAPTER 17
DESIGN OPTIMIZATION—AN OVERVIEW

A. Ravindran
Department of Industrial and Manufacturing Engineering
Pennsylvania State University
University Park, Pennsylvania

G. V. Reklaitis
School of Chemical Engineering
Purdue University
West Lafayette, Indiana

17.1 INTRODUCTION

This chapter presents an overview of optimization theory and its application to problems arising in engineering. In the most general terms, optimization theory is a body of mathematical results and numerical methods for finding and identifying the best candidate from a collection of alternatives without having to enumerate and evaluate explicitly all possible alternatives. The process of optimization lies at the root of engineering, since the classical function of the engineer is to design new, better, more efficient, and less expensive systems, as well as to devise plans and procedures for the improved operation of existing systems. The power of optimization methods to determie the best case without actually testing all possible cases comes through the use of a modest level of mathematics and at the cost of performing iterative numerical calculations using clearly defined logical procedures or algorithms implemented on computing machines. Because of the scope of most engineering applications and the tedium of the numerical calculations involved in optimization algorithms, the techniques of optimization are intended primarily for computer implementation.

Mechanical Engineers' Handbook, 2nd ed., Edited by Myer Kutz.
ISBN 0-471-13007-9 © 1998 John Wiley & Sons, Inc.

17.2 REQUIREMENTS FOR THE APPLICATION OF OPTIMIZATION METHODS

In order to apply the mathematical results and numerical techniques of optimization theory to concrete engineering problems it is necessary to delineate clearly the boundaries of the engineering system to be optimized, to define the quantitative criterion on the basis of which candidates will be ranked to determine the "best," to select the system variables that will be used to characterize or identify candidates, and to define a model that will express the manner in which the variables are related. This composite activity constitutes the process of *formulating* the engineering optimization problem. Good problem formulation is the key to the success of an optimization study and is to a large degree an art. It is learned through practice and the study of successful applications and is based on the knowledge of the strengths, weaknesses, and peculiarities of the techniques provided by optimization theory.

17.2.1 Defining the System Boundaries

Before undertaking any optimization study it is important to define clearly the boundaries of the system under investigation. In this context a system is the restricted portion of the universe under consideration. The system boundaries are simply the limits that separate the system from the remainder of the universe. They serve to isolate the system from its surroundings, because, for purposes of analysis, all interactions between the system and its surroundings are assumed to be frozen at selected, representative levels. Since interactions, nonetheless, always exist, the act of defining the system boundaries is the first step in the process of approximating the real system.

In many situations it may turn out that the initial choice of system boundary is too restrictive. In order to analyze a given engineering system fully it may be necessary to expand the system boundaries to include other subsystems that strongly affect the operation of the system under study. For instance, suppose a manufacturing operation has a point shop in which finished parts are mounted on an assembly line and painted in different colors. In an initial study of the paint shop we may consider it in isolation from the rest of the plant. However, we may find that the optimal batch size and color sequence we deduce for this system are strongly influenced by the operation of the fabrication department that produces the finished parts. A decision thus has to be made whether to expand the system boundaries to include the fabrication department. An expansion of the system boundaries certainly increases the size and complexity of the composite system and thus may make the study much more difficult. Clearly, in order to make our work as engineers more manageable, we would prefer as much as possible to break down large complex systems into smaller subsystems that can be dealt with individually. However, we must recognize that this decomposition is in itself a potentially serious approximation of reality.

17.2.2 The Performance Criterion

Given that we have selected the system of interest and have defined its boundaries, we next need to select a criterion on the basis of which the performance or design of the system can be evaluated so that the "best" design or set of operating conditions can be identified. In many engineering applications, an economic criterion is selected. However, there is a considerable choice in the precise definition of such a criterion: total capital cost, annual cost, annual net profit, return on investment, cost to benefit ratio, or net present worth. In other applications a criterion may involve some technology factors, for instance, minimum production time, maximum production rate, minimum energy utilization, maximum torque, and minimum weight. Regardless of the criterion selected, in the context of optimization the "best" will always mean the candidate system with either the *minimum* or the *maximum* value of the performance index.

It is important to note that within the context of the optimization methods, only *one* critrion or performance measure is used to define the optimum. It is not possible to find a solution that, say, simultaneously minimizes cost and maximizes reliability and minimizes energy utilization. This again is an important simplification of reality, because in many practical situations it would be desirable to achieve a solution that is "best" with respect to a number of different criteria. One way of treating multiple competing objectives is to select one criterion as primary and the remaining criteria as secondary. The primary criterion is then used as an optimization performance measure, while the secondary criteria are assigned acceptable minimum or maximum values and are treated as problem constraints. However, if careful considerations were not given while selecting the acceptable levels, a feasible design that satisfies all the constraints may not exist. This problem is overcome by a technique called *goal programming,* which is fast becoming a practical method for handling multiple criteria. In this method, all the objectives are assigned target levels for achievement and a relative priority on achieving these levels. Goal programming treats these targets as goals to aspire for and not as absolute constraints. It then attempts to find an optimal solution that comes as "close as possible" to the targets in the order of specified priorities. Readers interested in multiple criteria optimizations are directed to recent specialized texts.[1,2]

17.2.3 The Independent Variables

The third key element in formulating a problem for optimization is the selection of the independent variables that are adequate to characterize the possible candidate designs or operating conditions of the system. There are several factors that must be considered in selecting the independent variables.

First, it is necessary to distinguish between variables whose values are amenable to change and variables whose values are fixed by external factors, lying outside the boundaries selected for the system in question. For instance, in the case of the paint shop, the types of parts and the colors to be used are clearly fixed by product specifications or customer orders. These are specified system parameters. On the other hand, the order in which the colors are sequenced is, within constraints imposed by the types of parts available and inventory requirements, an independent variable that can be varied in establishing a production plan.

Furthermore, it is important to differentiate between system parameters that can be treated as fixed and those that are subject to fluctuations which are influenced by external and uncontrollable factors. For instance, in the case of the paint shop, equipment breakdown and worker absenteeism may be sufficiently high to influence the shop operations seriously. Clearly, variations in these key system parameters must be taken into account in the production planning problem formulation if the resulting optimal plan is to be realistic and operable.

Second, it is important to include in the formulation all of the important variables that influence the operation of the system or affect the design definition. For instance, if in the design of a gas storage system we include the height, diameter, and wall thickness of a cylindrical tank as independent variables, but exclude the possibility of using a compressor to raise the storage pressure, we may well obtain a very poor design. For the selected fixed pressure we would certainly find the least cost tank dimensions. However, by including the storage pressure as an independent variable and adding the compressor cost to our performance criterion, we could obtain a design that has a lower overall cost because of a reduction in the required tank volume. Thus, the independent variables must be selected so that all important alternatives are included in the formulation. Exclusion of possible alternatives, in general, will lead to suboptimal solutions.

Finally, a third consideration in the selection of variables is the level of detail to which the system is considered. While it is important to treat all of the key independent variables, it is equally important not to obscure the problem by the inclusion of a large number of fine details of subordinate importance. For instance, in the preliminary design of a process involving a number of different pieces of equipment—pressure vessels, towers, pumps, compressors, and heat exchangers—one would normally not explicitly consider all of the fine details of the design of each individual unit. A heat exchanger may well be characterized by a heat-transfer surface area as well as shell-side and tube-side pressure drops. Detailed design variables such as number and size of tubes, number of tube and shell passes, baffle spacing, header type, and shell dimensions would normally be considered in a separate design study involving that unit by itself. In selecting the independent variables a good rule to follow is to include only those variables that have a significant impact on the composite system performance criterion.

17.2.4 The System Model

Once the performance criterion and the independent variables have been selected, then the next step in problem formulation is the assembly of the model that describes the manner in which the problem variables are related and the performance criterion is influenced by the independent variables. In principle, optimization studies may be performed by experimenting directly with the system. Thus, the independent variables of the system or process may be set to selected values, the system operated under those conditions, and the system performance index evaluated using the observed performance. The optimization methodology would then be used to predict improved choices of the independent variable values and the experiments continued in this fashion. In practice most optimization studies are carried out with the help of a *model*, a simplified mathematical representation of the real system. Models are used because it is too expensive or time consuming or risky to use the real system to carry out the study. Models are typically used in engineering design because they offer the cheapest and fastest way of studying the effects of changes in key design variables on system performance.

In general, the model will be composed of the basic material and energy balance equations, engineering design relations, and physical property equations that describe the physical phenomena taking place in the system. These equations will normally be supplemented by inequalities that define allowable operating ranges, specify minimum or maximum performance requirements, or set bounds on resource availabilities. In sum, the model consists of all of the elements that normally must be considered in calculating a design or in predicting the performance of an engineering system. Quite clearly the assembly of a model is a very time-consuming activity, and it is one that requires a thorough understanding of the system being considered. In simple terms, a model is a collection of equations and inequalities that define how the system variables are related and that constrain the variables to take on acceptable values.

From the preceding discussion, we observe that a problem suitable for the application of optimization methodology consists of a performance measure, a set of independent variables, and a model relating the variables. Given these rather general and abstract requirements, it is evident that the methods of optimization can be applied to a very wide variety of applications. We shall illustrate next a few engineering design applications and their model formulations.

17.3 APPLICATIONS OF OPTIMIZATION IN ENGINEERING

Optimization theory finds ready application in all branches of engineering in four primary areas:

1. Design of components of entire systems.
2. Planning and analysis of existing operations.
3. Engineering analysis and data reduction.
4. Control of dynamic systems.

In this section we briefly consider representative applications from the first three areas.

In considering the application of optimization methods in design and operations, the reader should keep in mind that the optimization step is but one step in the overall process of arriving at an optimal design or an efficient operation. Generally, that overall process will, as shown in Fig. 17.1, consist of an iterative cycle involving synthesis or definition of the structure of the system, model formulation, model parameter optimization, and analysis of the resulting solution. The final optimal design or new operating plan will be obtained only after solving a series of optimization problems, the solution to each of which will have served to generate new ideas for further system structures. In the interest of brevity, the examples in this section show only one pass of this iterative cycle and focus mainly on preparations for the optimization step. This focus should not be interpreted as an indication of the

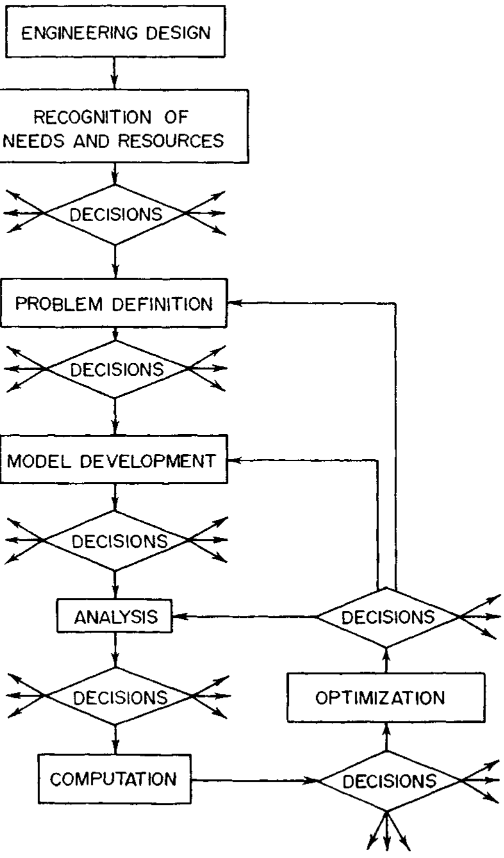


Fig. 17.1 Optimal design process.

dominant role of optimization methods in the engineering design and systems analysis process. Optimization theory is but a very powerful tool that, to be effective, must be used skillfully and intelligently by an engineer who thoroughly understands the system under study. The primary objective of the following example is simply to illustrate the wide variety but common form of the optimization problems that arise in the design and analysis process.

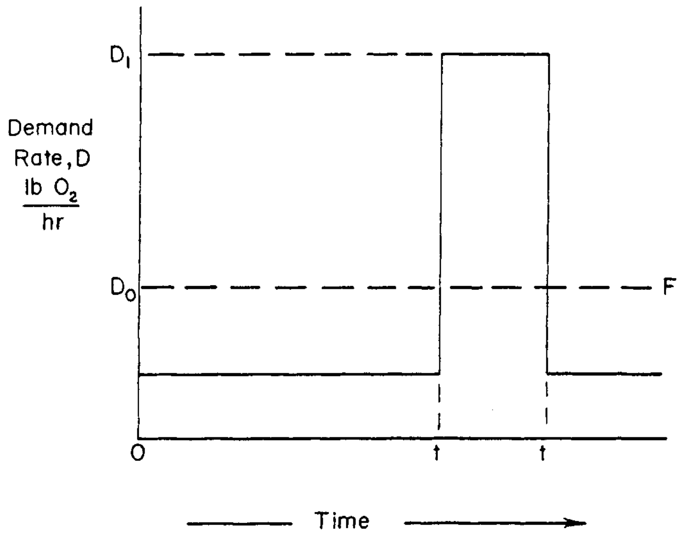
17.3.1 Design Applications

Applications in engineering design range from the design of individual structural members to the design of separate pieces of equipment to the preliminary design of entire production facilities. For purposes of optimization the shape or structure of the system is assumed known and optimization problem reduces to the selection of values of the unit dimensions and operating variables that will yield the best value of the selected performance criterion.

Example 17.1 Design of an Oxygen Supply System

Description. The basic oxygen furnace (BOF) used in the production of steel is a large fed-batch chemical reactor that employs pure oxygen. The furnace is operated in a cyclic fashion: ore and flux are charged to the unit, are treated for a specified time period, and then are discharged. This cyclic operation gives rise to a cyclically varying demand rate for oxygen. As shown in Fig. 17.2, over each cycle there is a time interval of length t_1 of low demand rate, D_0, and a time interval $(t_2 - t_1)$ of high demand rate, D_1. The oxygen used in the BOF is produced in an oxygen plant. Oxygen plants are standard process plants in which oxygen is separated from air using a combination of refrigeration and distillation. These are highly automated plants, which are designed to deliver a fixed oxygen rate. In order to mesh the continuous oxygen plant with the cyclically operating BOF, a simple inventory system shown in Fig. 17.3 and consisting of a compressor and a storage tank must be designed. A number of design possibilities can be considered. In the simplest case, one could select the oxygen plant capacity to be equal to D_1, the high demand rate. During the low-demand interval the excess oxygen could just be vented to the air. At the other extreme, one could also select the oxygen plant capacity to be just enough to produce the amount of oxygen required by the BOF over a cycle. During the low-demand interval, the excess oxygen production would then be compressed and stored for use during the high-demand interval of the cycle. Intermediate designs could involve some combination of venting and storage of oxygen. The problem is to select the optimal design.

Formulation. The system of concern will consist of the O_2 plant, the compressor, and the storage tank. The BOF and its demand cycle are assumed fixed by external factors. A reasonable performance index for the design is the total annual cost, which consists of the oxygen production cost (fixed and variable), the compressor operating cost, and the fixed costs of the compressor and of the storage



or, $D = \begin{cases} D_0 & \text{for } 0 \le t \ge t_1 \\ D_1 & \text{for } t_1 \le t \le t_2 \end{cases}$

Fig. 17.2 Oxygen demand cycle.

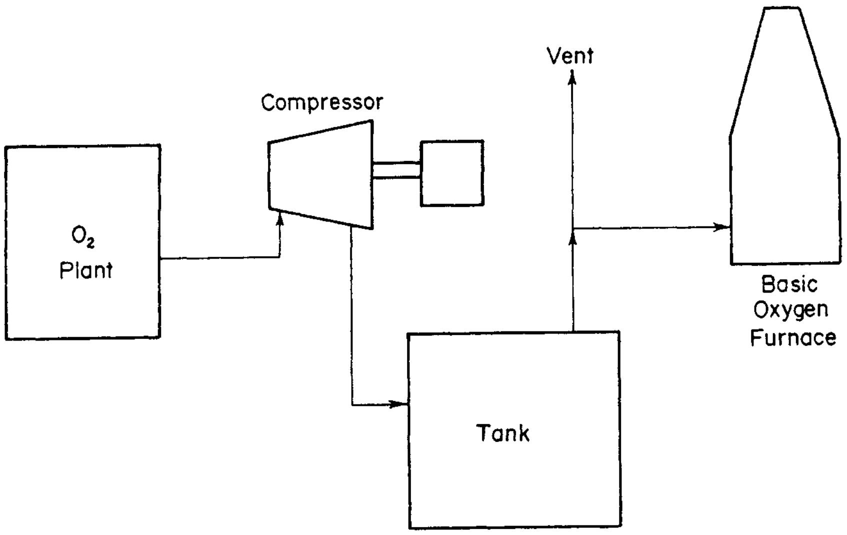


Fig. 17.3 Design of oxygen production system.

vessel. The key independent variables are the oxygen plant production rate F (lb O_2/hr), the compressor and storage tank design capacities, H (hp) and V (ft^3), respectively, and the maximum tank pressure, p (psia). Presumably the oxygen plant design is standard, so that the production rate fully characterizes the plant. Similarly, we assume that the storage tank will be of a standard design approved for O_2 service.

The model will consist of the basic design equations that relate the key independent variables.

If $I_{\max}$ is the maximum amount of oxygen that must be stored, then using the corrected gas law we have

$$V = \frac{I_{\max}}{M}\frac{RT}{p}z \tag{17.1}$$

where R = the gas constant
T = the gas temperature (assume fixed)
z = the compressibility factor
M = the molecular weight of O_2

From Fig. 17.1, the maximum amount of oxygen that must be stored is equal to the area under the demand curve between t_1 and t_2 and D_1 and F. Thus,

$$I_{\max} = (D_1 - F)(t_2 - t_1) \tag{17.2}$$

Substituting (17.2) into (17.1), we obtain

$$V = \frac{(D_1 - F)(t_2 - t_1)}{M}\frac{RT}{p}z \tag{17.3}$$

The compressor must be designed to handle a gas flow rate of $(D_1 - F)(t_2 - t_1)/t_1$ and to compress it to the maximum pressure of p. Assuming isothermal ideal gas compression,[3]

$$H = \frac{(D_1 - F)(t_2 - t_1)}{t_1}\frac{RT}{k_1k_2}\ln\left(\frac{p}{p_0}\right) \tag{17.4}$$

where k_1 = a unit conversion factor
k_2 = the compressor efficiency
p_0 = the O_2 delivery pressure

In addition to (17.3) and (17.4), the O_2 plant rate F must be adequate to supply the total oxygen demand, or

$$F \geq \frac{D_0 t + D_1(t_2 - t_1)}{t_2} \tag{17.5}$$

Moreover, the maximum tank pressure must be greater than the O_2 delivery pressure,

$$p \geq p_0 \tag{17.6}$$

The performance criterion will consist of the oxygen plant annual cost,

$$C_1(\$/\text{yr}) = a_1 + a_2 F \tag{17.7}$$

where a_1 and a_2 are empirical constants for plants of this general type and include fuel, water, and labor costs.

The capital cost of storage vessels is given by a power-law correlation,

$$C_2(\$) = b_1 V^{b_2} \tag{17.8}$$

where b_1 and b_2 are empirical constants appropriate for vessels of a specific construction.

The capital cost of compressors is similarly obtained from a correlation,

$$C_3(\$) = b_3 H^{b_4} \tag{17.9}$$

The compressor power cost will, as an approximation, be given by

$$b_5 t_1 H$$

where b_5 is the cost of power.

The total cost function will thus be of the form,

$$\text{Annual cost} = a_1 + a_2 F + d\{b_1 V^{b_2} + b_3 H^{b_4}\} + N b_5 t_1 H \tag{17.10}$$

where N = the number of cycles per year
d = an appropriate annual cost factor

The complete design optimization problem thus consists of the problem of minimizing (17.10), by the appropriate choice of F, V, H, and p, subject to Eqs. (17.3) and (17.4) as well as inequalities (17.5) and (17.6).

The solution of this problem will clearly be affected by the choice of the cycle parameters (N, D_0, D_1, t_1, and t_2), the cost parameters (a_1, a_2, b_1–b_5, and d), as well as the physical parameters (T, p_0, k_2, z, and M).

In principle, we could solve this problem by eliminating V and H from (17.10) using (17.3) and (17.4), thus obtaining a two-variable problem. We could then plot the contours of the cost function (17.10) in the plane of the two variables F and p, impose the inequalities (17.5) and (17.6), and determine the minimum point from the plot. However, the methods discussed in subsequent chapters allow us to obtain the solution with much less work. For further details and a study of solutions for various parameter values the reader is invited to consult Ref. 4.

The preceding example presented a preliminary design problem formulation for a system consisting of several pieces of equipment. The next example illustrates a detailed design of a single structural element.

Example 17.2 Design of a Welded Beam

Description. A beam A is to be welded to a rigid support member B. The welded beam is to consist of 1010 steel and is to support a force F of 6000 lb. The dimensions of the beam are to be selected so that the system cost is minimized. A schematic of the system is shown in Fig. 17.4.

Formulation. The appropriate system boundaries are quite self-evident. The system consists of the beam A and the weld required to secure it to B. The independent or design variables in this case are the dimensions h, l, t, and b as shown in Fig. 17.4. The length L is assumed to be specified at 14 in. For notational convenience we redefine these four variables in terms of the vector of unknowns $\mathbf{x}$,

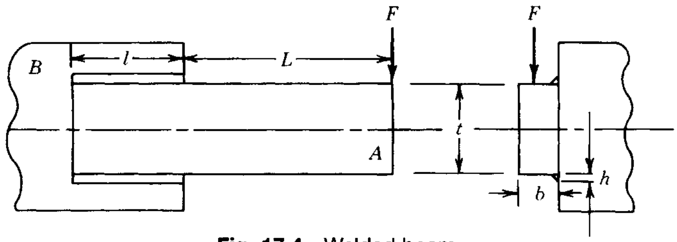


Fig. 17.4 Welded beam.

$$\mathbf{x} = [x_1, x_2, x_3, x_4]^T = [h, l, t, b]^T$$

The performance index appropriate to this design is the cost of a weld assembly. The major cost components of such an assembly are (a) set-up labor cost, (b) welding labor cost, and (c) material cost:

$$F(x) = c_0 + c_1 + c_2 \tag{17.11}$$

where $F(x)$ = cost function
c_0 = set-up cost
c_1 = welding labor cost
c_2 = material cost

Set-Up Cost: c_0. The company has chosen to make this component a weldment, because of the existence of a welding assembly line. Furthermore, assume that fixtures for set-up and holding of the bar during welding are readily available. The cost c_0 can, therefore, be ignored in this particular total cost model.

Welding Labor Cost: c_1. Assume that the welding will be done by machine at a total cost of \$10 per hour (including operating and maintenance expense). Furthermore, suppose that the machine can lay down 1 in.^{3} of weld in 6 min. Therefore, the labor cost is

$$c_1 = \left(10\,\frac{\$}{\text{hr}}\right)\left(\frac{1\text{ hr}}{60\text{ min}}\right)\left(6\,\frac{\text{min}}{\text{in.}^3}\right) V_w = 1\left(\frac{\$}{\text{in.}^3}\right) V_w$$

where V_w = weld volume, in.^{3}

Material Cost: c_2.

$$c_2 = c_3 V_w + c_4 V_B$$

where c_3 = \$/volume of weld material = (0.37)(0.283)(\$/in.^{3})
c_4 = \$/volume of bar stock = (0.17)(0.283)(\$/in.^{3})
V_B = volume of bar A (in.^{3})

From the geometry,

$$V_w = 2(\tfrac{1}{2}h^2 l) - h^2 l \qquad \text{and} \qquad V_B = tb(L + l)$$

so

$$c_2 = c_3 h^2 l + c_4 tb(L + l)$$

Therefore, the cost function becomes

$$F(x) = h^2 l + c_3 h^2 l + c_4 tb(L + l) \tag{17.12}$$

or, in terms of the x variables

$$F(x) = (l + c_3)x_1^2 x_2 + c_4 x_3 x_4 (L + x_2) \tag{17.13}$$

Note all combinations of x_1, x_2, x_3, and x_4 can be allowed if the structure is to support the load required. Several functional relationships between the design variables that delimit the region of feasibility must certainly be defined. These relationships, expressed in the form of inequalities, represent the design model. Let us first define the inequalities and then discuss their interpretation.

The inequities are:

$$g_1(x) = \tau_d - \tau(x) \geq 0 \tag{17.14}$$

$$g_2(x) = \sigma_d - \sigma(x) \geq 0 \tag{17.15}$$

$$g_3(x) = x_4 - x_l \geq 0 \tag{17.16}$$

$$g_4(x) = x_2 \geq 0 \tag{17.17}$$

$$g_5(x) = x_3 \geq 0 \tag{17.18}$$

$$g_6(x) = P_c(x) - F \geq 0 \tag{17.19}$$

$$g_7(x) = x_1 - 0.125 \geq 0 \tag{17.20}$$

$$g_8(x) = 0.25 - DEL(x) \geq 0 \tag{17.21}$$

where τ_d = design shear stress of weld
$\tau(x)$ = maximum shear stress in weld; a function of x
σ_d = design normal stress for beam material
$\sigma(x)$ = maximum normal stress in beam; a function of x
$P_c(x)$ = bar buckling load; a function of x
$DEL(x)$ = bar end deflection; a function of x

In order to complete the model it is necessary to define the important stress states.

Weld stress: $\tau(x)$. After Shigley,[5] the weld shear stress has two components, τ' and τ'', where τ' is the primary stress acting over the weld throat area and τ'' is a secondary torsional stress:

$$\tau' = F/\sqrt{2}x_1x_2 \qquad \text{and} \qquad \tau'' = MR/J$$

with $M = F[L + (x_2/2)]$
$R = \{(x_2^2/4) + [(x_3 + x_1)/2]^2\}^{1/2}$
$J = 2\{0.707x_1x_2[x_2^2/12 + (x_3 + x_1)/2)^2]\}$

where M = moment of F about the center of gravity of the weld group
J = polar moment of inertia of the weld group

Therefore, the weld stress τ becomes

$$\tau(x) = [(\tau')^2 + 2\tau'\tau'' \cos\theta + (\tau'')^2]^{1/2}$$

where $\cos\theta = x_2/2R$.

Bar Bending Stress: $\sigma(x)$. The maximum bending stress can be shown to be equal to

$$\sigma(x) = 6FL/x_4x_3^2$$

Bar Buckling Load: $P_c(x)$. If the ratio $t/b = x_3/x_4$ grows large, there is a tendency for the bar to buckle. Those combinations of x_3 and x_4 that will cause this buckling to occur must be disallowed. It has been shown[6] that for narrow rectangular bars, a good approximation to the buckling load is

$$P_c(x) = \frac{4.013\sqrt{EI\alpha}}{L^2}\left[1 - \frac{x_3}{2L}\sqrt{\frac{EI}{\alpha}}\right]$$

where E = Young's modulus = 30×10^6 psi
$I = \frac{1}{12}x_3x_4^3$
$\alpha = \frac{1}{3}Gx_3x_4^3$
G = shearing modulus = 12×10^6 psi

Bar deflection: $DEL(x)$. To calculate the deflection assume the bar to be a cantilever of length L. Thus,

$$DEL(x) = 4FL^3/Ex_3^3x_4$$

The remaining inequalities are interpreted as follows.

g_3 states that it is not practical to have the weld thickness greater than the bar thickness. g_4 and g_5 are nonnegativity restrictions on x_2 and x_3. Note that the nonnegativity of x_1 and x_4 are implied by g_3 and g_7. Constraint g_6 ensures that the buckling load is not exceeded. Inequality g_7 specifies that it is not physically possible to produce an extremely small weld.

Finally, the two parameters τ_d and σ_d in g_1 and g_2 depend on the material of construction. For 1010 steel $\tau_d = 13{,}600$ psi and $\sigma_d = 30{,}000$ psi are appropriate.

The complete design optimization problem thus consists of the cost function (17.13) and the complex system of inequalities that results when the stress formulas are substituted into (17.14) through (17.21). All of these functions are expressed in terms of four independent variables.

This problem is sufficiently complex that graphical solution is patently infeasible. However, the optimum design can readily be obtained numerically using the methods of subsequent sections. For a further discussion of this problem and its solution the reader is directed to Ref. 7.

17.3.2 Operations and Planning Applications

The second major area of engineering application of optimization is found in the tuning of existing operations. We shall discuss an application of goal programming model for machinability data optimization in metal cutting.[8]

Example 17.3 An Economic Machining Problem with Two Competing Objectives

Consider a single-point, single-pass turning operation in metal cutting wherein an optimum set of cutting speed and feed rate is to be chosen which balances the conflict between metal removal rate and tool life as well as being within the restrictions of horsepower, surface finish, and other cutting conditions. In developing the mathematical model of this problem, the following constraints will be considered for the machining parameters:

Constraint 1: Maximum Permissible Feed.

$$f \leqq f_{\max} \tag{17.22}$$

where f is the feed in inches per revolution. $f_{\max}$ is usually determined by a cutting force restriction or by surface finish requirements.[9]

Constraint 2: Maximum Cutting Speed Possible. If v is the cutting speed in surface feet per minute, then

$$v \leqq v_{\max} \tag{17.23}$$

where

$$v_{\max} = \frac{\pi D N_{\max}}{12}$$

and

$$N_{\max} = \text{maximum spindle speed available on the machine}$$

Constraint 3: Maximum Horsepower Available. If $P_{\max}$ is the maximum horsepower available at the spindle, then

$$vf^{\alpha} \leq \frac{P_{\max}(33{,}000)}{c_t d_c^{\beta}}$$

where α, β, and c_t are constants.[9] d_c is the depth of cut in inches, which is fixed at a given value. For a given $P_{\max}$, c_t, β, and d_c, the right-hand side of the above constraint will be a constant. Hence, the horsepower constraint can be written simply as

$$vf^{\alpha} \leqq \text{constant} \tag{17.24}$$

Constraint 4: Nonnegativity Restrictions on Feed Rate and Speed.

$$v, f \geqq 0 \tag{17.25}$$

In optimizing metal cutting there are a number of optimality criteria that can be used. Suppose we consider the following objectives in our optimization: (i) maximize metal removal rate (MRR), (ii) maximize tool life (TL). The expression for MRR is

$$\text{MRR} = 12vfd_c \text{ in.}^3/\text{min} \tag{17.26}$$

TL for a given depth of cut is given by

$$\text{TL} = \frac{A}{v^{1/n}f^{1/n_1}} \tag{17.27}$$

where A, n, and n_1 are constants. We note that the MRR objective is directly proportional to feed and speed, while the TL objective is inversely proportional to feed and speed. In general, there is no single solution to a problem formulated in this way, since MRR and TL are competing objectives and their respective maxima must include some compromise between the maximum of MRR and the maximum of TL.

A Goal Programming Model

Goal programming is a technique specifically designed to solve problems involving complex, usually conflicting multiple objectives. Goal programming requires the user to select a set of goals (which may or may not be realistic) that ought to be achieved (if possible) for the various objectives. It then uses preemptive weights or priority factors to rank the different goals and tries to obtain an optimal solution satisfying as many goals as possible. For this, it creates a single objective function that minimizes the deviations from the stated goals according to their relative importance.

Before we discuss the goal programming formulation of the machining problem, we should discuss the difference between the terms "real constraint" and "goal constraint" (or simply "goal") as used in goal programming models. The real constraints are absolute restrictions placed on the behavior of the design variables, while the goal constraints are conditions one would like to achieve but are not mandatory. For instance, a real constraint given by

$$x_1 + x_2 = 3$$

requires all possible values of $x_1 + x_2$ to always equal 3. As opposed to this, if we simply had a goal requiring $x_1 + x_2 = 3$, then this is not mandatory and we can choose values of x_1, x_2 such that $x_1 + x_2 \geq 3$ as well as $x_1 + x_2 \leq 3$. In a goal constraint positive and negative deviational variables are introduced as follows:

$$x_1 + x_2 + d_1^- - d_1^+ = 3, \qquad d_1^-, d_1^+ \geq 0$$

Note that if $d_1^- > 0$, then $x_1 + x_2 < 3$, and if $d_1^+ > 0$, then $x_1 + x_2 > 3$. By assigning suitable preemptive weights on d_1^- and d_1^+, the model will try to achieve the sum $x_1 + x_2$ as close as possible to 3.

Returning to the machining problem with competing objectives, suppose that management considers that a given single-point, single-pass turning operation will be operating at an acceptable efficiency level if the following goals are met as closely as possible.

1. The MRR must be greater than or equal to a given rate M_1 (in.^{3}/min).
2. The tool life must equal T_1 (min).

In addition, management requires that a higher priority be given to achieving the first goal than the second.

The goal programming approach may be illustrated by expressing each of the goals as goal constraints as shown below. Taking the MRR goal first,

$$12vfd_c + d_1^- - d_1^+ = M_1$$

where d_1^- represents the amount by which the MRR goal is underachieved, and d_1^+ represents any overachievement of the MRR goal. Similarly, the TL goal can be expressed as

$$\frac{A}{v^{1/n}f^{1/n_1}} + d_2^- - d_2^+ = T_1$$

Since the objective is to have an MRR of at least M_1, the objective function must be set up so that a high penalty will be assigned to the underachievement variable d_1^-. No penalty will be assigned to d_1^+. In order to achieve a tool life of T_1, penalties must be associated with both d_2^- and d_2^+ so that both of these variables are minimized to their fullest extent. The relative magnitudes of these penalties must reflect the fact that the first goal is considered to be more important that the second. Accordingly, the goal programming objective function for this problem is

$$\text{Minimize } z = P_1 d_2^- + P_2(d_2^- + d_2^+)$$

where P_1 and P_2 are nonnumerical preemptive priority factors such that $P_1 >>> P_2$ (i.e., P_1 is infinitely larger than P_2). With this objective function every effort will be made to satisfy completely the first goal before any attempt is made to satisfy the second.

In order to express the problem as a linear goal programming problem, M_1 is replaced by M_2, where

$$M_2 = \frac{M_1}{12d_c}$$

The goal T_1 is replaced by T_2, where

$$T_2 = \frac{A}{T_1}$$

and logarithms are taken of the goals and constraints. The problem can then be stated as follows:

$$\text{Minimize } z = P_1 d_1^- + P_2(d_2^- + d_2^+)$$

Subject to

(MRR goal)	$\log v + \log f + d_1^- - d_1^+ = \log M_2$
(TL goal)	$(1/n)\log v + (1/n_1)\log f + d_2^- - d_2^+ = \log T_2$
($f_{\max}$ constraint)	$\log f \leqq \log f_{\max}$
($V_{\max}$ constraint)	$\log v \leqq \log v_{\max}$
(Horsepower constraint)	$\log v + \alpha \log f \leqq \log \text{constant}$

$$\log v, \log f, d_1^-, d_1^+, d_2^-, d_2^+ \geqq 0$$

We would like to reemphasize here that the last three inequalities are real constraints on feed, speed, and horsepower that must be satisfied at all times, while the equations for MRR and TL are simply goal constraints. For a further discussion of this problem and its solution, see Ref. 8. An efficient algorithm and a computer code for solving linear goal programming problems is given in Ref. 10. Readers interested in other optimization models in metal cutting should see Ref. 11. The textbook by Lee[12] contains a good discussion of goal programming theory and its applications.

17.3.3 Analysis and Data Reduction Applications

A further fertile area for the application of optimization techniques in engineering can be found in nonlinear regression problems as well as in many analysis problems arising in engineering science. A very common problem arising in engineering model development is the need to determine the parameters of some semitheoretical model given a set of experimental data. This data reduction or regression problem inherently transforms to an optimization problem, because the model parameters must be selected so that the model fits the data as closely as possible.

Suppose some variable y is assumed to be dependent on an independent variable x and related to x through a postulated equation $y = f(x, \theta_1, \theta_2)$, which depends on two parameters θ_1 and θ_2. To establish the appropriate values of θ_1 and θ_2, we run a series of experiments in which we adjust the independent variable x and measure the resulting y. As a result of a series of N experiments covering the range of x of interest, a set of y and x values (y_i, x_i), $i = 1, \ldots, N$, is available. Using these data we now try to "fit" our function to the data by adjusting θ_1 and θ_2 until we get a "good fit." The most commonly used measure of a "good fit" is the *least squares criterion,*

$$L(\theta_1, \theta_2) = \sum_{i=1}^{N} [y_i - f(x_i, \theta_1, \theta_2)]^2 \tag{17.28}$$

The difference $y_i - f(x_i, \theta_1, \theta_2)$ between the experimental value y_i and the predicted value $f(x_i, \theta_1, \theta_2)$ measures how close our model prediction is to the data and is called the *residual.* The sum of the squares of the residuals at all the experimental points gives an indication of goodness of fit. Clearly, if $L(\theta_1, \theta_2)$ is equal to zero, then the choice of θ_1, θ_2 has led to a perfect fit; the data points fall exactly on the predicted curve. The data-fitting problem can thus be viewed as an optimization problem in which $L(\theta_1, \theta_2)$ is minimized by appropriate choice of θ_1 and θ_2.

Example 17.4 Nonlinear Curve Fitting

Description. The pressure-molar-volume-temperature relationship of real gases is known to deviate from that predicted by the ideal gas relationship

$$Pv = RT$$

where P = pressure (atm)
v = molar volume (cm^3/g · mol)
T = temperature (K)
R = gas constant (82.06 atm · cm^3/g · mol · K)

The semiempirical Redlich–Kwong equation

$$P = \frac{RT}{v - b} - \frac{a}{T^{1/2}v(v + b)} \tag{17.29}$$

is intended to direct for the departure from ideality but involves two empirical constants a and b whose values are best determined from experimental data. A series of PvT measurements listed in Table 17.1 are made for CO$_2$, from which a and b are to be estimated using nonlinear regression.

Formulation. Parameters a and b will be determined by minimizing the least squares function (17.28). In the present case, the function will take the form

$$\sum_{i=1}^{8} \left[P_i - \frac{RT_i}{v_i - b} + \frac{a}{T^{1/2}v_i(v_i + b)} \right]^2 \tag{17.30}$$

where P_i is the experimental value at experiment i, and the remaining two terms correspond to the value of P predicted from Eq. (17.29) for the conditions of experiment i for some selected value of the parameters a and b. For instance, the term corresponding to the first experimental point will be

$$\left(33 - \frac{82.06(273)}{500 - b} + \frac{a}{(273)^{1/2}(500)(500 + b)} \right)^2$$

Function (17.30) is thus a two-variable function whose value is to be minimized by appropriate choice of the independent variables a and b. If the Redlich–Kwong equation were to precisely match the data, then at the optimum the function (17.30) would be exactly equal to zero. In general, because of experimental error and because the equation is too simple to accurately model the CO$_2$ nonidealities, Eq. (17.30) will not be equal to zero at the optimum. For instance, the optimal values of a = 6.377 × 10^7 and b = 29.7 still yield a squared residual of 9.7 × 10^{-2}.

Table 17.1 PvT Data for CO$_2$

Experiment Number	P (atm)	v (cm^3/g · mol)	T°(K)
1	33	500	273
2	43	500	323
3	45	600	373
4	26	700	273
5	37	600	323
6	39	700	373
7	38	400	273
8	63.6	400	373

17.4 STRUCTURE OF OPTIMIZATION PROBLEMS

Although the application problems discussed in the previous section originate from radically different sources and involve different systems, at root they have a remarkably similar form. All four can be expressed as problems requiring the minimization of a real-valued function $f(x)$ of an N-component vector argument $x = (x_1, x_2, \ldots, x_N)$ whose values are restricted to satisfy a number of real-valued equations $h_k(x) = 0$, a set of inequalities $g_j(x) \geqslant 0$, and the variable bounds $x_i^{(U)} \geqslant x_i \geqslant x_i^{(L)}$. In subsequent discussions we will refer to the function $f(x)$ as the *objective function,* to the equations $h_k(x) = 0$ as the *equality constraints,* and to the inequalities $g_j(x) \geqslant 0$ as the *inequality constraints.* For our purposes, these problem functions will always be assumed to be real valued, and their number will always be finite.

The general problem,

$$\begin{array}{ll} \text{Minimize } f(x) & \\ \text{Subject to } h_k(x) = 0 & k = 1, \ldots, K \\ \qquad g_j(x) \geqslant 0 & j = 1, \ldots, J \\ \qquad x_i^{(L)} \geqslant x_i \geqslant x_i^{(L)} & i = 1, \ldots, N \end{array}$$

is called the *constrained* optimization problem. For instance, Examples 17.1, 17.2, and 17.3 are all constrained problems. The problem in which there are no constraints, that is,

$$J = K = 0$$

and

$$x_i^{(U)} = -x_i^{(L)} = \infty, \qquad i = 1, \ldots, N$$

is called the *unconstrained* optimization problem. Example 17.4 is an unconstrained problem. Optimization problems can be classified further based on the structure of the functions f, h_k, and g_j and on the dimensionality of x. Figure 17.5 illustrates one such classification. The basic subdivision is between unconstrained and constrained problems. There are two important classes of methods for solving the unconstrained problems. The direct search methods require only that the objective function be evaluated at different points, at least through experimentation. Gradient-based methods require the analytical form of the objective function and its derivatives.

An important class of constrained optimization problems is *linear programming,* which requires both the objective function and the constraints to be linear functions. Out of all optimization models, linear programming models are the most widely used and accepted in practiced. Professionally written software programs are available from all major computer manufacturers for solving very large linear programming problems. Unlike the other optimization problems that require special solution methods based on the problem structure, linear programming has just one common algorithm, known as the "simplex method," for solving all types of linear programming problems. This essentially has contributed to the successful applications of linear programming models in practice. In 1984, Narendra Karmarkar,[13] an AT&T researcher, developed an interior point algorithm, which was claimed to be 50 times faster than the simplex method for solving linear programming problems. By 1990, Karmarkar's seminal work had spawned hundreds of research papers and a large class of interior point methods. It has become clear that while the initial claims are somewhat exaggerated, interior point methods do become competitive for very large problems. For a discussion of interior point methods, see Refs. 14 and 15.

Integer programming **(IP)** is another important class of linearly constrained problems where some or all of the design variables are restricted to be integers. But solutions of IP problems are generally difficult, time-consuming, and expensive. Hence, a practical approach is to treat all the integer variables as continuous, solve the associated LP problem, and round off the fractional values to the nearest integers such that the constraints are not violated. This generally produces a good integer solution close to the optimal integer solution, particularly when the values of the variables are large. However, such an approach would fail when the values of the variables are small or binary valued (0 or 1). A good rule of thumb is to treat any integer variable whose value will be at least 20 as continuous and use special purpose IP algorithms for the rest. For a complete discussion of integer programming applications and algorithms, see Refs. 16 and 17.

The next class of optimization problems involves nonlinear objective functions and linear constraints. Under this class we have the following:

1. Quadratic programming, whose objective is a quadratic function.
2. Convex programming, whose objective is a special nonlinear function satisfying an important mathematical property called "convexity."

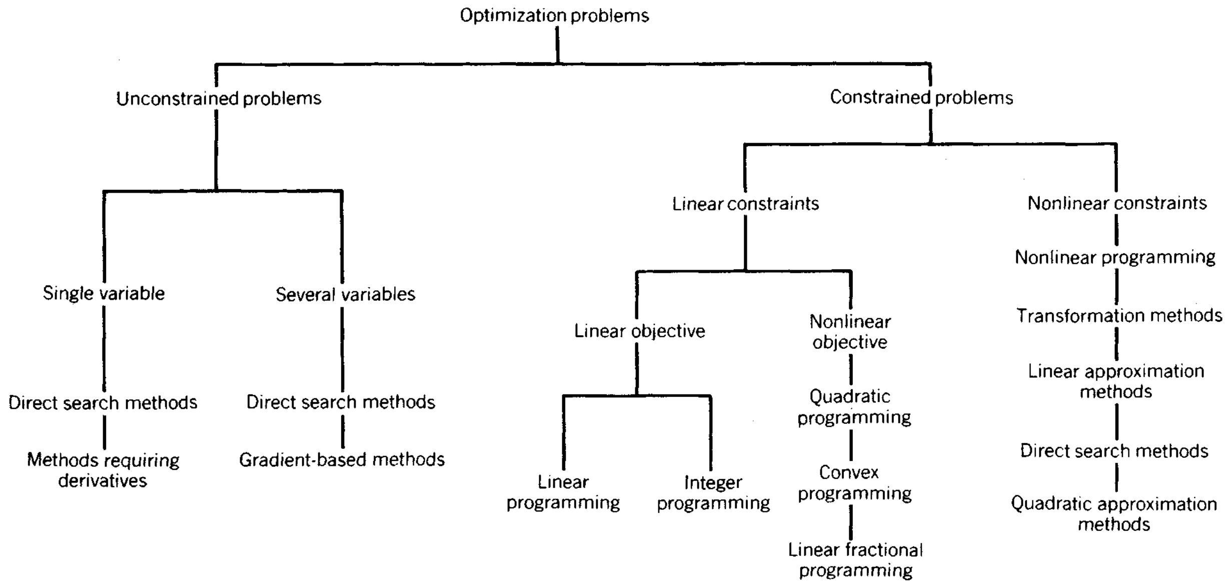


Fig. 17.5 Classification of optimization problems.

3. Linear fractional programming, whose objective is the ratio of two linear functions.

Special-purpose algorithms that take advantage of the particular form of the objective functions are available for solving these problems.

The most general optimization problems involve nonlinear objective functions and nonlinear constraints and are generally grouped under the term "nonlinear programming." The majority of engineering design problems fall into this class. Unfortunately, there is no single method that is best for solving every nonlinear programming problem. Hence, a host of algorithms is available for solving the general nonlinear programming problem, some of these algorithms are reviewed in the next section.

Nonlinear programming problems wherein the objective function and the constraints can be expressed as the sum of generalized polynomial functions are called *geometric programming* problems. A number of engineering design problems fall into the geometric programming framework. Since its earlier development in 1961, geometric programming has undergone considerable theoretical development, has experienced a proliferation of proposals for numerical solution techniques, and has enjoyed considerable practical engineering applications (see Refs. 18 and 19).

Nonlinear programming problems where some of the design variables are restricted to be discrete or integer valued are called *mixed integer nonlinear programming* (MINLP) problems. Such problems arise in process design, simulation optimization, industrial experimentation, and reliability optimization. MINLP problems are generally more difficult to solve since the problems have several local optima. Recently, *simulated annealing* and *genetic algorithms* have been emerging as powerful heuristic algorithms to solve MINLP problems. Simulated annealing has been successfully applied to solve problems in a variety of fields, including mathematics, engineering, and mathematical programming (see, for example, Refs. 20–22).

Genetic algorithms are heuristic search methods based on the two main principles of natural genetics, namely, entities in a population reproduced to create offspring and the survival of the fittest (see, for example, Refs. 23 and 21). For a discussion of the successful applications of genetic algorithms and the areas of research in the field, see Ref. 24.

17.5 OVERVIEW OF OPTIMIZATION METHODS

Optimization methods can be viewed as nothing more than numerical hill-climbing procedures in which the objective function, presenting the topology of the hill, is searched to identify the highest point—or maximum—subject to constraining relations that might be equality constraints (stay on winding path) or inequality constraints (stay within fence boundaries). While the constraints do serve to reduce the area that must be searched, the numerical calculations required to ensure that the search stays on the path or within the fences generally do constitute a considerable burden. Accordingly, optimization methods for unconstrained problems and methods for linear constraints are less complex than those designed for nonlinear constraints. In this section, a selection of optimization techniques representative of the main families of methods will be discussed. For a more detailed presentation of individual methods the reader is invited to consult Ref. 25.

17.5.1 Unconstrained Optimization Methods

Methods for unconstrained problems are divided into those for single-variable functions and those appropriate for multivariable functions. The former class of methods are important because single-variable optimization problems arise commonly as subproblems in the solution of multivariable problems. For instance, the problem of minimizing a function $f(x)$ for a point x^0 in a direction d (often called a *line search*) can be posed as a minimization problem in the scalar variable α:

$$\text{Minimize } f(x^0 + \alpha d)$$

Single Variable Methods

These methods are roughly divided into *region elimination methods* and *point estimation methods.* The former use comparison of function values at selected trial points to reject intervals within which the optimum of the function does not lie. The latter typically use polynomial approximating functions to estimate directly the location of the optimum. The simplest polynomial approximating function is the quadratic

$$\tilde{f}(x) = ax^2 + bx + c$$

whose coefficients a, b, c can be evaluated readily from those trial values of the actual function. The point at which the derivative of $\tilde{f}$ is zero is used readily to predict the location of the optimum of the true function

$$\tilde{x} = -b/2a$$

The process is repeated using successively improved trial values until the differences between successive estimates $\tilde{x}$ become sufficiently small.

Multivariable Unconstrained Methods

These algorithms can be divided into *direct search methods* and *gradient-based methods.* The former methods only use direct function values to guide the search, while the latter also require the computation of function gradient and, in some cases, second derivative values. Direct search methods in widespread use in engineering applications include the simplex search, the pattern search method of Hooke and Jeeves, random-sampling-based methods, and the conjugate directions method of Powell (see Chap. 3 of Ref. 25). All but the last of these methods make no assumptions about the smoothness of the function contours and hence can be applied to both discontinuous and discrete-valued objective functions.

Gradient-based methods can be grouped into the classical methods of steepest descent (Cauchy) and Newton's method and the modern quasi-Newton methods such as the conjugate gradient, Davidson–Fletcher–Powell, and Broyden–Fletcher–Shanno algorithms. All gradient-based methods employ the first derivative or gradient of the function at the current best solution estimate $\bar{x}$ to compute a direction in which the objective function value is guaranteed to decrease (a *descent* direction). For instance, Cauchy's classical method used the direction.

$$d = -\nabla f(\bar{x})$$

followed by a line search from $\bar{x}$ in this direction. In Newton's method the gradient vector is premultiplied by the matrix of second derivatives to obtain an improved direction vector

$$d = -(\nabla^2 f(\bar{x}))^{-1}\nabla f(\bar{x})$$

which in theory at least yields very good convergence behavior. However, the computation of $\nabla^2 f$ is often too burdensome for engineering applications. Instead, in recent years quasi-Newton methods have found increased application. In these methods, the direction vector is computed as

$$d = -H\nabla f(\bar{x})$$

where H is a matrix whose elements are updated as the iterations proceed using only values of gradient and function value difference from successive estimates. Quasi-Newton methods differ in the details of H updating, but all use the general form

$$H^{n+1} = H^n + C^n$$

where H^n is the previous value of H and C^n is a suitable correction matrix. The attractive feature of this family of methods is that convergence rates approaching those of Newton's method are attained without the need for computing $\nabla^2 f$ or solving the linear equation set

$$\nabla^2 f(\bar{x}) \cdot d = -f(\bar{x})$$

to obtain d. Recent developments in these methods have focused on strategies for eliminating the need for detailed line searching along the direction vectors and on enhancements for solving very large problems. For a detailed discussion of quasi-Newton methods the reader is directed to Refs. 25 and 26.

17.5.2 Constrained Optimization Methods

Constrained optimization methods can be classified into those applicable to totally linear or at least linearly constrained problems and those applicable to general nonlinear problems. The linear or linearly constrained problems can be well solved using methods of linear programming and extensions, as discussed earlier. The algorithms suitable for general nonlinear problems comprise four broad categories of methods:

1. Direct search methods that use only objective and constraint function values.
2. Transformation methods that use constructions that aggregate constraints with the original objective function to form a single composite unconstrained function.
3. Linearization methods that use linear approximations of the nonlinear problem functions to produce efficient search directions.
4. Successive quadratic programming methods that use quasi-Newton constructions to solve the general problem via a series of subproblems with quadratic objective function and linear constraints.

Direct Search

The direct search methods essentially consist of extensions of unconstrained direct search procedures to accommodate constraints. These extensions are generally only possible with inequality constraints or linear equality constraints. Nonlinear equalities must be treated by implicit or explicit variable elimination. That is, each equality constraint is either explicitly solved for a selected variable and used to eliminate that variable from the search or the equality constraints are numerically solved for values of the dependent variables for each trial point in the space of the independent variables.

For example, the problem

$$\begin{aligned} &\text{Minimize} && f(\mathbf{x}) = x_1x_2x_3 \\ &\text{Subject to} && h_1(\mathbf{x}) = x_1 + x_2 + x_3 - 1 = 0 \\ &&& h_2(\mathbf{x}) = x_1^2x_3 + x_2x_3^2 + x_2^{-1} - 2 = 0 \\ &&& \mathbf{0} \leq (x_1, x_3) \leq \tfrac{1}{2} \end{aligned}$$

involves two equality constraints and hence can be viewed as consisting of two dependent variables and one independent variable. Clearly, h_1 can be solved for x_1 to yield

$$x_1 = 1 - x_2 - x_3$$

Thus, on substitution the problem reduces to

$$\begin{aligned} &\text{Minimize} && (1 - x_2 - x_3)x_2x_3 \\ &\text{Subject to} && (1 - x_2 - x_3)^2x_3 + x_2x_3^2 + x_2^{-1}(1 - x_2 - x_3)\ 2 = 0 \\ &&& 0 \leq 1 - x_2 - x_3 \leq \tfrac{1}{2} \\ &&& 0 \leq x_3 \leq \tfrac{1}{2} \end{aligned}$$

Solution of the remaining equality constraint for one variable, say x_3, in terms of the other is very difficult. Instead, for each value of the independent variable x_2, the corresponding value of x_3 would have to be calculated numerically using some root-finding method.

Some of the more widely used direct search methods include the adaptation of the simplex search due to Box (called the *complex method*), various direct random-sampling-type methods, and combined random sampling/heuristic procedures such as the combinatorial heuristic meethod[27] advanced for the solution of complex optimal mechanism design problems.

A typical direct sampling procedure is given by the formula,

$$x_ip = \bar{x}_i \times z_i(2r - 1)^k, \qquad \text{for each variable } x_i, \quad i = 1, \ldots, n$$

where $\bar{x}_i$ = the current best value of variable i
z_i = the allowable range of the variable i
r = a random variable uniformly distributed on the interval 0–1
k = an adaptive parameter whose value is adjusted based on past successes or failures in the search

For given $\bar{x}$, z, and k, r is sampled N times and the new point x^p evaluated. If x^p satisfies all constraints, it is retained; if it is infeasible, it is rejected and a new set of N r values is generated. If x^p is feasible, $f(x^p)$ is compared to $f(\bar{x})$, and if improvement is found, x^p replaces $\bar{x}$. Otherwise x^p is rejected. The parameter k is an adaptive parameter whose value will regulate the contraction or expansion of the sampling region. A typical adjustment procedure for k might be to increase k by 2 whenever a specified number of improved points is found or to decrease it by 2 when no improvement is found after a certain number of trials.

The general experience with direct search and especially random-sampling-based methods for constrained problems is that they can be quite effective for severely nonlinear problems that involve multiple local minima but are of low dimensionality.

Transformation Methods

This family consists of strategies for converting the general constrained problem to a parametrized unconstrained problem that is solved repeatedly for successive values of the parameters. The approaches can be grouped into the penalty/barrier function constructions, exact penalty methods, and augmented Lagrangian methods. The classical penalty function approach is to transform the general constrained problem to the form

$$P(x, R) = f(x) + \Omega(R, g(x), h(x))$$

where R = the penalty parameter
Ω = the penalty term

The ideal penalty function will have the property that

$$P(x, R) = \begin{cases} f(x), & \text{if } x \text{ is feasible} \\ \infty, & \text{if } x \text{ is infeasible} \end{cases}$$

Given this idealized construction, $P(x, R)$ could be minimized using any unconstrained optimization method, and, hence, the underlying constrained problem would have been solved. In practice such radical discontinuities cannot be tolerated from a numerical point of view, and, hence, practical penalty functions use penalty terms of the form

$$\Omega(R, g, h) = R\left(\sum_k h_k(x)\right)^2 + R(\Sigma(\min(0, g_j(x)))^2)$$

A series of unconstrained minimizations of $P(x, R)$ with different values of R are carried out beginning with a low value of R (say $R = 1$) and progressing to very large values of R. For low values of R, the unconstrained minima of $P(x, R)$ obtained will involve considerable constraint violations. As R increases, the violations decrease until in the limit as $R \rightarrow \infty$, the violations will approach zero. A large number of different forms of the Ω function have been proposed; however, all forms share the common feature that a sequence of problems must be solved and that, as the penalty parameter R becomes large, the penalty function becomes increasingly distorted and thus its minimization becomes increasingly more difficult. As a result the penalty function approach is best used for modestly sized problems (2–10 variables), few nonlinear equalities (2–5), and a modest number of inequalities. In engineering applications, the unconstrained subproblems are most commonly minimized using direct search methods, although successful use of quasi-Newton methods is also reported.

The exact penalty function and augmented Lagrangian approaches have been developed in an attempt to circumvent the need to force convergence by using increasing values of the penalty parameter. One typical representative of this type of method is the so-called meethod of multipliers.[28] In this method, once a sufficiently large value of R is reached, further increases are not required. However, the method does involve additional finite parameters that must be updated between subproblem solutions. Computational evidence reported to date suggests that, while augmented Lagrangian approaches are more reliable than penalty-function methods, they, as a class, are not suitable for larger dimensionality problems.

Linearization Methods

The common characteristic of this family of methods, is the use of local linear approximations to the nonlinear problem functions to define suitable, preferably feasible, directions for search. Well-known members of this family include the method of feasible directions, the gradient projection meethod, and the generalized reduced gradient (GRG) method. Of these, the GRG method has seen the widest engineering application.

The key constructions of the GRG method are the following:

1. The calculation of the reduced objective function gradient $\nabla\tilde{f}$.
2. The use of the reduced gradient to determine a direction vector in the space of the independent variables.
3. The adjustment of the dependent variable values using Newton's method so as to achieve constraint satisfaction.

Given a feasible point x^0, the gradients of the equality constraints are evaluated and used to form the constraint Jacobian matrix A. This matrix is partitioned into a square submatrix J and the residual rectangular matrix C where the variable associated with the columns of J are the dependent variables and those associated with C are the independent variables.

If J is selected to have nonzero determinant, then the reduced gradient is defined as

$$\nabla\tilde{f}(x^0) = \nabla\tilde{f} - \nabla\hat{f}J^{-1}C$$

where $\nabla\bar{f}$ is the subvector of objective function partial derivatives corresponding to the (dependent) variables and $\hat{f}$ is the corresponding subvector whose components correspond to the independent variables. The reduced gradient $\nabla\tilde{f}$ provides an estimate of the rate of change of $f(x)$ with respect to the independent variables when the dependent variables are adjusted to satisfy the linear approximations to the constraints.

Given $\nabla \tilde{f}$, in the simplest version of GRF algorithm, the direction subvector for the independent variables $\bar{d}$ is selected to be the reduced gradient descent direction

$$\bar{d} = -\nabla \tilde{f}$$

For a given step α in that direction, the constraints are solved iteratively to determine the value of the dependent variables $\hat{x}$ that will lead to a feasible point. Thus, the system

$$h_k(\bar{x}^0 + \alpha \bar{d}, \hat{x}) = 0, \qquad k = 1, \ldots, K$$

is solved for the K unknown variables $\hat{x}$. The new feasible point is checked to determine whether an improved objective value has been obtained and, if not, α is reduced and the solution for $\hat{x}$ repeated. The overall algorithm terminates when a point is reached at which the reduced gradient is sufficiently close to zero.

The GRG algorithm has been extended to accommodate inequality constraints as well as variable bounds. Moreover, the use of efficient equation solving procedures, line search procedures for α, and quasi-Newton formulas to generate improved direction vectors $\bar{d}$ have been investigated. A commercial quality GRG code will incorporate such developments and thus will constitute a reasonably complex software package. Computational testing using such codes indicates that GRG implementations are among the most robust and efficient general purpose nonlinear optimization methods currently available.[29] One of the particular advantages of this algorithm, which can be critical in engineering applications, is that it generates *feasible* intermediate points; hence, it can be interrupted prior to final convergence to yield a feasible solution. Of course, this attractive feature and the general efficiency of the method are attained at the price of providing (analytically or numerically) the values of the partial derivatives of all of the model functions.

Successive Quadratic Programming (SQP) Methods

This family of methods seeks to attain superior convergence rates by employing subproblems constructed using higher-order approximating functions than those employed by the linearization methods. The SQP methods are still the subject of active research; hence, developments and enhancements are proceeding apace. However, the basic form of the algorithm is well established and can be sketched out as follows.

At a given point x^0, a direction finding subproblem is constructed, which takes the form of a quadratic programming problem:

$$\begin{aligned} \text{Minimize} \quad & \nabla^T f \cdot d + \tfrac{1}{2} d^T H\, d \\ \text{Subject to} \quad & h_k(x^0) + \nabla^T h_k(x^0)\, d = 0 \\ & g_j(x^0) + \nabla^T g_j(x^0)\, d \ge 0 \end{aligned}$$

The symmetric matrix H is a quasi-Newton approximation of the matrix of second derivatives of a composite function (the Lagrangian) containing terms corresponding to all of the functions f, h_k, and g_j. H is updated using only gradient differences as in the unconstrained case. The direction vector d is used to conduct a line search, which seeks to minimize a penalty function of the type discussed earlier. The penalty function is required because, in general, the intermediate points produced in this method will be infeasible. Use of the penalty function ensures that improvements are achieved in either the objective function values or the constraint violations or both. One major advantage of the method is that very efficient methods are available for solving large quadratic programming problems and, hence, that the method is suitable for large scale applications. Recent computational testing indicates that the SQP approach is very efficient, outperforming even the best GRG codes.[30] However, it is restricted to models in which infeasibilities can be tolerated and will produce feasible solutions only when the algorithm has converged.

17.5.3 Code Availability

With the exception of the direct search methods and the transformation-type methods, the development of computer programs implementing state-of-the-art optimization algorithms is a major effort requiring expertise in numerical methods in general and numerical linear algebra in particular. For that reason, it is generally recommended that engineers involved in design optimization studies take advantage of the number of good quality implementations now available through various public sources.

Commercial computer codes for solving LP/IP/NLP problems are available from many computer manufacturers and private companies who specialize in marketing software for major computer systems. Depending on their capabilities, these codes vary in their complexity, ease of use, and cost (see, for example, Ref. 34). LP models with a few hundred constraints can now be solved on personal computers (PCs). There are now at least a hundred small companies marketing LP software for PCs. For a 1995 survey of LP software for personal computers, see Ref. 35.

Nash[36] presents a 1995 survey of nonlinear programming software that will run on PC compatibles, Macintosh systems, and UNIX-based workstations. Detailed product descriptions, prices, and capabilities of 30 NLP software are included in the survey. There are now LP/IP/NLP solvers that can be invoked directly from inside spreadsheet packages. For example, Microsoft Excel and Microsoft Office for Windows and Macintosh contain a general-purpose optimizer for solving small-scale linear, integer, and nonlinear programming problems. Borland's Quattro-Pro also has a built-in solver for optimization. In both spreadsheet programs, the LP optimizer is based on the simplex algorithm, while the NLP optimizer is based on the GRG algorithm.

There are now modeling languages that allow the user to express a model in a very compact algebraic form, with whole classes of constraints and variables defined over index sets. Models with thousands of constraints and variables can be defined in a couple of pages, in a syntax that is very close to standard algebraic notation. The algebraic form of the model is kept separate from the actual data for any particular instance of the model. The computer takes over the responsibility of transforming the abstract form of the model and the specific data into a specific constraint matrix. This has greatly simplified the building, and even more the changing, of optimization models. There are several modeling languages available for PCs. The two high-end products are GAMS (General Algebraic Modeling System) and AMPL (A Mathematical Programming Language). For a reference on GAMS, see Ref. 37. For a general introduction to modeling languages, see Refs. 34 and 38, and for an excellent discussion of AMPL, see Ref. 39.

Readers with access to the Internet can get a complete list of optimization software available for LP, IP, and NLP problems at the following NEOS web site:

http://www.mcs.anl.gov/home/otc

This site provides access not only to the software guide but also to the other optimization-related sites that are continually updated. The NEOS guide on optimization software is based on the textbook by More and Wright,[40] an excellent resource for those interested in a broad review of the various optimization methods and their computer codes. The book is divided into two parts. Part I has an overview of algorithms for different optimization problems, categorized as unconstrained optimization, nonlinear least squares, nonlinear equations, linear programming, quadratic programming, bound-constrained optimization, network optimization, and integer programming. Part II includes product descriptions of 75 software packages that implement the algorithms described in Part I. Much of the software described in this book is in the public domain and can be obtained through the Internet.

17.6 SUMMARY

In this chapter an overview was given of the elements and methods comprising design optimization methodology. The key element in the overall process of design optimization was seen to be the engineering model of the system constructed for this purpose. The assumptions and formulation details of the model govern the quality and relevance of the optimal design obtained. Hence, it is clear that design optimization studies cannot be relegated to optimization software specialists but are the proper domain of the well-informed design engineer.

The chapter also gave a structural classification of optimization problems and a broad brush review of the main families of optimization methods. Clearly this review can only hope to serve as entry point to this broad field. For a more complete discussion of optimization techniques with emphasis on engineering applications, guidelines for model formulation, practical solution strategies, and available computer software, the readers are referred to the text by Reklaitis, Ravindran, and Ragsdell.[25]

The Design Automation Committee of the Design Engineering Division of ASME has been sponsoring conferences devoted to engineering design optimization. Several of these presentations have subsequently appeared in the *Journal of Mechanical Design, ASME Transactions.* Ragsdell[31] presents a review of the papers published up to 1977 in the areas of machine design applications and numerical methods in design optimization. ASME published, in 1981, a special volume entitled *Progress in Engineering Optimization,* edited by Mayne and Ragsdell.[32] It contains several articles pertaining to advances in optimization methods and their engineering applications in the areas of mechanism design, structural design, optimization of hydraulic networks, design of helical springs, optimization of hydrostatic journal bearing, and others. Finally, the persistent and mathematically oriented reader may wish to pursue the fine exposition given by Avrial,[33] which explores the theoretical properties and issues of nonlinear programming methods.

REFERENCES

1. M. Zeleny, *Multiple Criteria Decision Making,* McGraw-Hill, New York, 1982.
2. T. L. Vincent and W. J. Grantham, *Optimality in Parametric Systems,* Wiley, New York, 1981.

3. K. E. Bett, J. S. Rowlinson, and G. Saville, *Thermodynamics for Chemical Engineers,* MIT Press, Cambridge, MA, 1975.
4. F. C. Jen, C. C. Pegels, and T. M. Dupuis, "Optimal Capacities of Production Facilities," *Management Science* **14B,** 570–580 (1968).
5. J. E. Shigley, *Mechanical Engineering Design,* McGraw-Hill, New York, 1973, p. 271.
6. S. Timoshenko and J. Gere, *Theory of Elastic Stability,* McGraw-Hill, New York, 1961, p. 257.
7. K. M. Ragsdell and D. T. Phillips, "Optimal Design of a Class of Welded Structures Using Geometric Programming," *ASME J. Eng. Ind. Ser. B* **98**(3), 1021–1025 (1975).
8. R. H. Philipson and A. Ravindran, "Application of Goal Programming to Machinability Data Optimization," *Journal of Mechanical Design, Trans. of ASME* **100,** 286–291 (1978).
9. E. J. A. Armarego and R. H. Brown, *The Machining of Metals,* Prentice-Hall, Englewood Cliffs, NJ, 1969.
10. J. L. Arthur and A. Ravindran, "PAGP-Partitioning Algorithm for (Linear) Goal Programming Problems," *ACM Transactions on Mathematical Software* **6,** 378–386 (1980).
11. R. H. Philipson and A. Ravindran, "Application of Mathematical Programming to Metal Cutting," *Mathematical Programming Study* **11,** 116–134 (1979).
12. S. M. Lee, *Goal Programming for Decision Analysis,* Auerbach Publishers, Philadelphia, PA, 1972.
13. N. K. Karmarkar, "A New Polynomial Time Algorithm for Linear Programming," *Combinatorica* **4,** 373–395 (1984).
14. A. Arbel, *Exploring Interior Point Linear Programming: Algorithms and Software,* MIT Press, Cambridge, MA, 1993.
15. S.-C. Fang and S. Puthenpura, *Linear Optimization and Extensions,* Prentice-Hall, NJ, 1993.
16. K. G. Murty, *Operations Research: Deterministic Optimization Models,* Prentice-Hall, Englewood Cliffs, 1995.
17. G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization,* Wiley, New York, 1988.
18. C. S. Beightler and D. T. Phillips, *Applied Geometric Programming,* Wiley, New York, 1976.
19. M. J. Rijckaert, "Engineering Applications of Geometric Programming," in *Optimization and Design,* M. Avriel, M. J. Rijckaert, and D. J. Wilde (eds.), Prentice-Hall, Englewood Cliffs, NJ, 1974.
20. I. O. Bohachevsky, M. E. Johnson, and M. L. Stein, "Generalized Simulated Annealing for Function Optimization," *Technometrics* **28,** 209–217 (1986).
21. L. Davis (ed.), *Genetic Algorithms and Simulated Annealing,* Pitman, London, 1987.
22. S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," *Science,* **220,** 670–680 (1983).
23. D. E. Goldberg, *Genetic Algorithm in Search, Optimization, and Machine Learning,* Addison-Wesley, Reading, MA, 1989.
24. A. Maria, "Genetic Algorithms for Multimodal Continuous Optimization Problems," Ph.D. Diss., University of Oklahoma, Norman, OK, 1995.
25. G. V. Reklaitis, A. Ravindran, and K. M. Ragsdell, *Engineering Optimization: Methods and Applications,* Wiley, New York, 1983.
26. R. Fletcher, *Practical Methods of Optimization,* 2nd ed., Wiley, New York, 1987.
27. T. W. Lee and F. Fruedenstein, "Heuristic Combinatorial Optimization in the Kinematic Design of Mechanisms: Part 1: Theory," *J. Eng. Ind. Trans. ASME,* 1277–1280 (1976).
28. S. B. Schuldt, G. A. Gabriele, R. R. Root, E. Sandgren, and K. M. Ragsdell, "Application of a New Penalty Function Method to Design Optimization," *J. Eng. Ind. Trans. ASME,* 31–36 (1977).
29. E. Sandgren and K. M. Ragsdell, "The Utility of Nonlinear Programming Algorithms: A Comparative Study—Parts 1 and 2," *Journal of Mechanical Design, Trans. of ASME* **102,** 540–541 (1980).
30. K. Schittkowski, *Nonlinear Programming Codes: Information, Tests, Performance,* Lecture Notes in Economics and Mathematical Systems, Vol. 183, Springer-Verlag, New York, 1980.
31. K. M. Ragsdell, "Design and Automation," *Journal of Mechanical Design, Trans. of ASME* **102,** 424–429 (1980).
32. R. W. Mayne and K. M. Ragsdell (eds.), *Progress in Engineering Optimization,* ASME, New York, 1981.
33. M. Avriel, *Nonlinear Programming: Analysis and Methods,* Prentice-Hall, Englewood Cliffs, NJ, 1976.

34. R. Sharda, *Linear and Discrete Optimization and Modeling Software: A Resource Handbook,* Lionheart, Atlanta, GA, 1993.

35. R. Sharda, "Linear Programming Solver Software for Personal Computers: 1995 Report," *OR/MS Today* **22,** 49–57 (1995).

36. S. G. Nash, "Software Survey NLP," *OR/MS Today* **22,** 60–71 (1995).

37. A. Brooke, D. Kendrick, and A. Meeraus, *GAMS: A User's Guide,* Scientific Press, Redwood City, CA, 1988.

38. R. Sharda and G. Rampal, "Algebraic Modeling Languages on PC's," *OR/MS Today* **22,** 58–63, 1995.

39. R. Fourer, D. M. Gay, and B. W. Kernighan, "A Modelling Language for Mathematical Programming," *Management Science* **36,** 519–554 (1990).

40. J. J. More and S. J. Wright, *Optimization Software Guide,* SIAM Publications, Philadelphia, PA, 1993.