Chapter 2

OPTIMIZATION – AT A GLANCE

2.1 OPTIMIZATION METHODS

In most of the engineering design problems the main objective is either to minimize the cost or to maximize the efficiency of production. An optimization algorithm is a procedure which is executed iteratively by comparing various possible solutions till an optimum or satisfactory solution is arrived at. Choice of a method depends mainly on the type of the problem to be solved due to the growing complexity of engineering design problems. Generally, global optimization algorithms can be divided into two basic classes i.e. deterministic and probabilistic algorithms.

(i) **Deterministic Type**: They have specific rules for moving from one solution to other and have been successfully applied to many engineering problems. Deterministic algorithms are mostly used if a clear relation between the characteristics of the possible solutions and their utility for a given problem can be observed.

(ii) **Probabilistic Type**: They are a new class of algorithms involving heuristics and are superior to the former type in certain aspects. If the relation between a solution candidate and its "goodness" however is not obvious or neighboring solution candidates may differ largely in their utility, it becomes harder to solve a problem deterministically. It would probably result in exhaustive enumeration of the search space, which is not feasible even for relatively small problems. Then, probabilistic algorithms come into play.

Further, optimization methods can be subdivided into two categories depending on the type of design variables i.e. continuous and discrete variable optimization methods. In continuous methods, the design variables can have any values in a specified range whereas in discrete variable methods they can only have values from a specified set of discrete values. The discrete variables methods are very useful for practical applications where such variables occur naturally. Sequential linear programming, sequential quadratic programming (SQP) and hybrid methods are some of the continuous variable optimization methods. The enumeration method, branch and bound methods, simulated annealing and genetic algorithms

are some of the methods for discrete variable optimization. If an application has mixed variables, a combination of these algorithms can be used to solve it.

The existence of optimization methods can be traced to the days of Newton, Lagrange and Cauchy. The development of differential calculus methods of optimization was possible because of the contributions of Newton and Leibnitz to calculus. The foundation of calculus of variations, which deals with the minimization of functionals, was laid by Bernoulli, Euler and Lagrange. The method of constrained problem optimization, which involves the addition of unknown multiplier was invented by Lagrange and became popular by his name Lagrange's multiplier method. The steepest descent method was first used by Cauchy to solve unconstrained optimization problems. Due of availability of high-speed computers in the middle of twentieth century, several well-defined new areas in optimization emerged.

The development in area of numerical optimization methods started with emergence of simplex method by Dantzig in 1947 for linear programming problems and the annunciation of the principle of optimality in 1957 by Bellman for dynamic programming problems. Although no single technique was found to be universally applicable for nonlinear programming problems, Carroll et. al. developed techniques to solve many difficult problems of unconstrained optimization. In 1960 Duffin et. al. developed well-known geometric programming. Gomory did pioneering work in one of the most exciting and rapidly developing area of optimization, the integer programming. The most real-world application fall under this category. Dantzig et. al. developed stochastic programming techniques. The Goal programming was proposed by Charnes and Cooper in 1961 for linear problems but later became well-known for specific types of multi-objective optimization problems.

Simulated annealing, genetic algorithms, neural network and fuzzy logic form a new class of mathematical programming techniques that have come in to prominence during last two decades. These are all heuristic search methods which have become popular for solving vast variety of single and multi-objective optimization problems.

Thus optimization methods can be broadly classified in three classes: (i) Calculus based methods (ii) Numerical methods and (iii) Random search methods. Classical methods are not robust as they are applicable to only limited class of problems. However, numerous hybrid methods developed by the researchers have been able to solve many complex problems.

Many newly evolved soft computing techniques cited above and their hybrid models have been and continue to be the best among all available methods for optimization problems.

2.2 Local and Global Optima

Optimization can be defined as the branch of applied mathematics that deals with the maximization or minimization of single or may be even multiple criteria. These criteria are expressed as a set of mathematical functions $F = \{f_1, f_2, ..., f_n\}$, the so-called objective functions. The result of the optimization process is the set of inputs for which these objective functions return optimal values.

In the case of a single function, i.e. $F = \{f\}$, an optimum is either a maximum or a minimum. As illustrated in Fig. 2.1 local and global optima can be distinguished. A global optimum is an optimum of the whole domain X while a local optimum is only an optimum of one of its subsets.

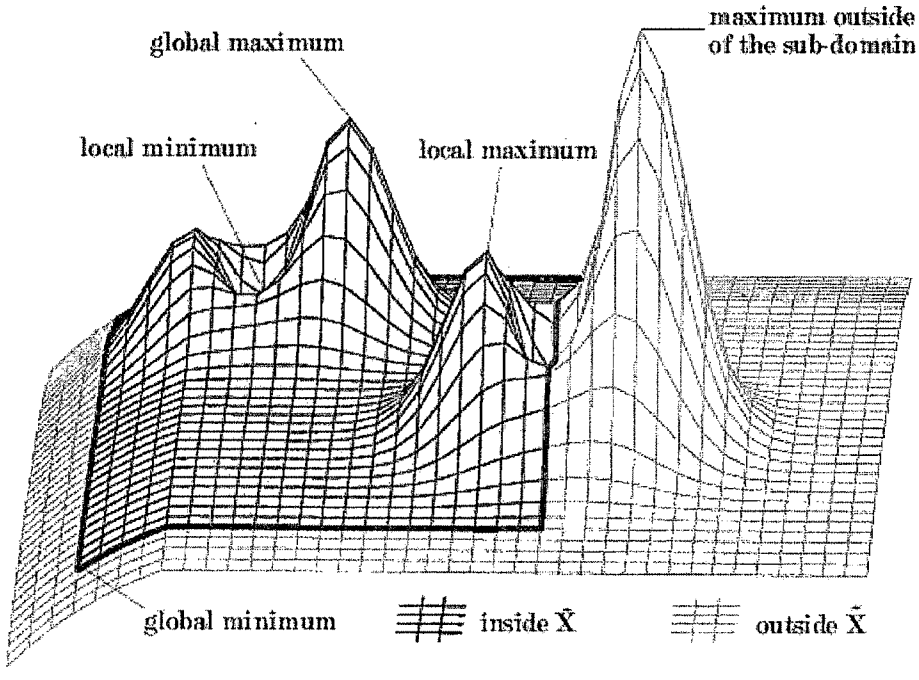


Fig. 2.1 Local and Global Optimum of a Two-Dimensional Function

2.3 Major Issues of Global Optimization

2.3.1 Premature Convergence to a Local Optimum

One of the greatest problems in global optimization is that most often it is difficult to determine if the best solution currently known is a local or a global optimum. In other words, it can not be said whether to concentrate on refining the current optimum or to

examine other parts of the search space instead. A global optimization process is said to have prematurely converged to a local optimum if it is no longer able to explore other parts of the search space than the currently examined area and there exists such another region in the search space that contains a solution superior to the currently exploited one which could be found with reasonable effort. Figure 2.2 illustrates how an optimization algorithm prematurely converges. There are many features and parameter settings of optimization algorithms that influence the convergence behavior. Some of the important factors are self-adaptation and the operations that create new solutions from existing ones.

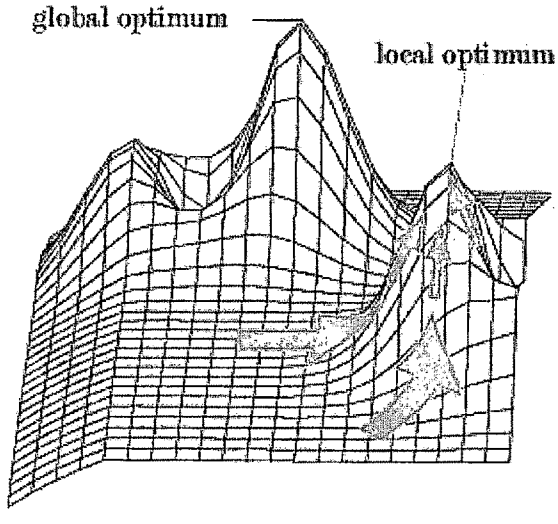


Fig. 2.2 Premature Convergence

2.3.2 Exploration vs. Exploitation

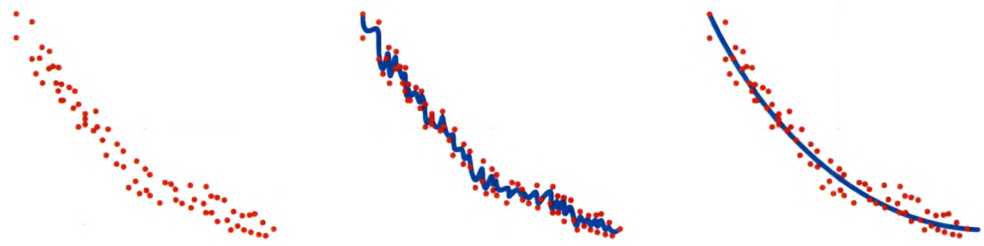
All optimization algorithms have to trade-off between exploration and exploitation. Exploration in terms of optimization means finding new points in the search space. It is the only mean to find new and may be better solutions but leads to performance degradation at least until a new good solution is found – which is not guaranteed at all. Exploitation means trying to improve the currently known solution(s) by small changes which lead to new individuals very close to them. This way, performance improvements can be achieved. If another, maybe better solution exists in a distant area whatsoever, one will not be able to find it. Almost all parts of optimization strategies can either be used for increasing exploitation or in favour of exploration. While algorithms that favour exploitation have a fast convergence, they run a great risk of not finding the optimal solution and maybe get stuck at a local optimum. Algorithms that perform excessive exploration may find the global optimum but it

will take them very long to do so. Exploration supports diversity whereas exploitation works against it. Diversity preservation is a major concern in optimization because the loss of it can lead to premature convergence to a local optimum.

2.3.3 Overfitting

Overfitting is fitting a model that has too many parameters. This phenomenon can often be encountered in the field of artificial neural networks or in curve fitting. The major problem that results from overfitted solutions is the loss of generality which can be explained as under.

A solution of an optimization process is considered to be general if it is valid for all possible inputs. An overfitted solution will not be able to produce valid results for inputs which differ from the training data used to create it. Figure 2.3 (b) shows overfitted solution and Fig. 2.3(c) shows correct solution for the sample input data of Fig. 2.3(a) illustrating generality.



(a) Sample Data **(b) Overfitted Result** **(c) Correct Result**

Fig. 2.3 Overfitting in Curve Fitting

2.3.4 Evolvability

Evolvability can be defined in the contexts of biology and global optimization. A biological system is evolvable if its properties show heritable genetic variation and if natural selection can change these properties or if it can acquire new characteristics via genetic change. The degree of evolvability in an optimization process in its current state defines how likely the reproduction operations will yield solution candidates with new fitness values.

2.4 Optimization Parameters and Problem Formulation

Optimization is the branch of Operation Research - the discipline of applying advanced analytical methods to help make better decision. It is the process of finding maximum

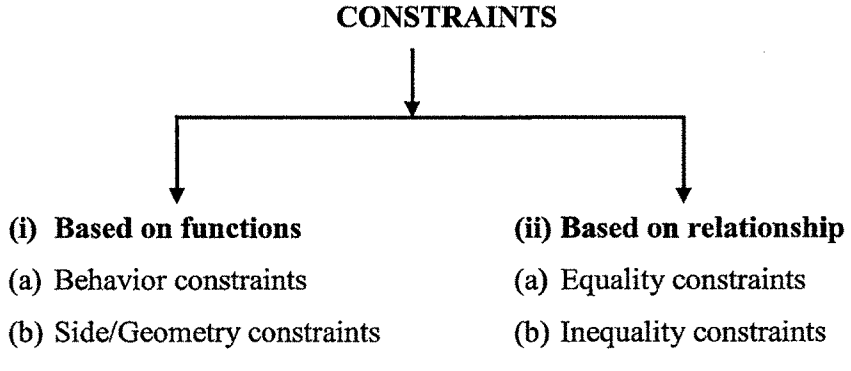
benefits from minimum resources while satisfying various limitations or restrictions known as constraints. Benefits may be in the form of minimum cost and maximum profit in the business and maximum safety and serviceability in structural engineering projects. While formulating optimization problem following parameters should be judiciously thought of and selected.

2.4.1 Design variables

In general, design variables are quantities controlled to improve the objective which should completely describe the set of decisions to be made. The efficiency and speed of optimization algorithms depends mainly on the number of chosen design variables. Thus the first thumb rule in the optimum design problem formulation is to choose as few design variables as possible. In truss optimization problem, for example, the design variables are coordinates of the nodes, c/s areas of the members, number of nodes, number of members, position and type of supports etc. The second point is to select the working range of values for each of them since the optimization algorithm finds the best result from this range. This requires thorough knowledge of the problem under consideration.

2.4.2 Constraints

These are the limitations or restrictions on the values of the decision variables and functions of these variables. The constraint represents some function relationships among the design variables and other design parameters satisfying certain physical phenomenon and certain resource limitations.



(i) Based on functions

Behavior constraints are constraints, which are related to design variables implicitly. Usually structural analysis is necessary to evaluate them. These constraints represent limitation on the behavior or performance of system. Examples of behavior constraints are limitations on

stresses, displacement and stability requirements. On the other hand, constraints that represent physical limitations on design variables such as availability, fabricability, and transportability are known as *geometric constraints*. The codal provisions of minimum or maximum values on design variables are the examples of these constraints. In the design of two-way slabs, for example, the ratio of longer span to shorter span should be less than 2.0.

(ii) Based on relationship

Equality constraints need the design variable or function of design variables to exactly match a resource value, e.g. $\delta(x) = 5$ i.e. deflection of a node must be equal to 5 mm. Equality constraints are usually more difficult to handle. It may be possible to reduce the number of design variables by using equality constraints. For example, in design of structures specific size/dimension of the structure or structural component may be held constant in terms of constraint.

In many engineering design problems it may be possible to relax an equality constraint by including two inequality constraints. For example the above deflection equality constraint can be replaced by two inequality constraints:

$$\delta(x) \leq 4 \text{ and } \delta(x) \geq 6 \qquad \ldots (2.1)$$

Thus, the second important point in formulation of optimal design problem is that number of complex equality constraints should be kept as low as possible.

Inequality constraint requires the functional relationships among design variables to be either greater than or smaller than resource value. For example $\sigma(x)$ $S_{allowable}$ i.e. stress developed in any component must be less than the allowable stress. Most of the constraints encountered in engineering design are of this type. One type of inequality constraint can be transformed into other type by multiplying both the sides by -1 or by interchanging the left and right sides. e.g. $-\sigma(x) \geq -S_{allowable}$ or $S_{allowable} \leq \sigma(x)$.

2.4.3 Objective Function

Objective function is the value measure used to rank alternative solutions to a given problem in the search space. This is the function in terms of the design variables and other problem parameters with respect to which the design is optimized. The common engineering

objectives involve minimization of overall cost of manufacturing or weight of a component, or maximization of net profit earned or total life of a product. The choice of objective function is governed by the nature of problem.

Most of the optimization problems have a single objective function with the following two exceptions:

(i) ***No objective function:*** In some cases the goal is to find a set of variables that satisfies the constraints of the model. The user does not particularly want to optimize anything so there is no reason to define an objective function. This type of problems is usually called a *feasibility problem*

(ii) ***Multiple objective functions:*** Optimization techniques are not just applied to find the maxima or minima of single objective function *f*. In many real-world design or decision making problems they are applied to sets F of n functions f_i which represent multiple criteria,

$$F = \{f_i(X) : 1 < i \leq n\} \quad \ldots (2.2)$$

Algorithms designed to optimize such a set F of objective functions are usually named with the prefix multi-objective, like multi-objective evolutionary algorithms. Usually, the different objectives are not compatible; the variables that optimize one objective may be far from optimal for the others. Thus, multi-objective optimization often means to compromise conflicting goals. In practice, problems with multiple objectives are reformulated as single-objective problems by forming a weighted combination g(x) of the different objective functions $f_i(x) \in F$, given as,

$$g(x) = \sum_{i=1}^{n} w_i f_i(x) \quad \ldots (2.3)$$

The weights w_i represent the importance of the single functions and also determine if the function should be maximized ($w_i > 0$) or minimized ($w_i < 0$). Alternative method to solve multi-objective optimization is Pareto optimization. Pareto optimization, Pareto efficiency or Pareto optimality, is an important notion in neoclassical economics with broad applications in game theory, engineering and the social sciences. It defines the front of solutions that can be reached by trading-off the conflicting objects in an optimal manner from which one can finally choose the configuration that, in his opinion, suits the best. The notation of Pareto optimal is strongly based on the definition of domination - An element x_1 dominates (is

preferred to) an element x_2 if x_1 is better than x_2 in at least one objective function and not worse with respect to all other objective functions.

In structural engineering, the designer is interested in minimization of cost or weight. In the present work, for example, cost is considered as the objective function for R.C.C. structures and weight for the steel structures.

2.4.4 Variable Bounds

The final task of the formulation procedure is to set the minimum and the maximum bounds on each design variable. Certain optimization algorithms do not require this information. In these problems, the constraints completely surround the feasible region. Other problems require this information in order to confine the search algorithm within these bounds. In general, all *N* design variables are restricted to lie within the minimum and maximum bounds as follows:

$$(Xi)^L \leq Xi \leq (Xi)^U, \quad \text{for } i = 1, 2, \ldots\ldots, N \qquad \ldots (2.4)$$

The determination of the variables bounds $(Xi)_L$ and $(Xi)_U$ may be difficult. One way to remedy this situation is to make a guess about the optimal solution and set the minimum and maximum bounds so that the optimal solution lies within these two bounds. After simulating the optimization algorithm once, if the optimum solution is found to lie within the chosen variable bounds, there is no problem. On the other hand, if any design variable corresponding to the optimal solution is found to lie on or near the minimum or maximum bound, the chosen bound may not be correct. The chosen bound may be readjusted and the optimization algorithm may be simulated again. Although this strategy may seem to work only with linear problems, it has been found useful in many real-world engineering optimization problems.

2.4.5 Mathematical Modeling

After the above four tasks are completed, the optimization problem can be mathematically written in a special format as shown below:

$$\text{Find}, X = \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{Bmatrix}, \text{ which minimizes}, f(X)$$

subject to the constraints,

$$g_i(X) \leq 0, \qquad i = 1, 2, \ldots, m,$$
$$l_j(X) = 0, \qquad j = 1, 2, \ldots, q,$$
$$X_k^L \leq X_k \leq X_k^U \qquad k = 1, 2, \ldots, n, \qquad \ldots (2.5)$$

where X is n - dimensional vector called *design vector*, $f(X)$ is the *objective function*, $g_i(X)$ and $l_j(X)$ are m inequality and q equality constraints respectively and X_k^L and X_k^U = upper and lower bounds of k^{th} design variable. The following pseudo code shows the basic steps involved in optimal design procedure.

```
[Choose design parameters],
[Formulate objective function/s],
[Formulate design constraints],
[Set up variable bounds],
[Select Optimization algorithm],
[Obtain solution],
[Stop].
```