Optimization Methods

1.0. Introduction:

In optimization of a design, the design objective could be simply to minimize the cost of production or to maximize the efficiency of production. An optimization algorithm is a procedure which is executed iteratively by comparing various solutions till an optimum or a satisfactory solution is found.

With the advent of computers, optimization has become a part of computer-aided design activities. There are two distinct types of optimization algorithms widely used today.

(a) Deterministic Algorithms.

They use specific rules for moving one solution to other. These algorithms are in use to suite some times and have been successfully applied for many engineering design problems.

(b) Stochastic Algorithms.

The stochastic algorithms are in nature with probabilistic translation rules. These are gaining popularity due to certain properties which deterministic algorithms do not have.

2.0 ***Optimal problem formulation:***

A naive optimal design is achieved by comparing a few (limited up to ten or so) alternative solutions created by using a priori problem knowledge. In this method feasibility of each design solution is first investigated. Thereafter an estimate of underlying objective (cost, profit, etc.,) of each solution is compared and best solution is adopted.

It is impossible to apply single formulation procedure for all engineering design problems, since the objective in a design problem and associated therefore, design parameters vary product to product different techniques are used in

different problems. Purpose of formulation is to create a mathematical model of the optimal design problem, which then can be solved using an optimization algorithm. Figure 1 shows an outline of the steps usually involved in an optimal design formulation.

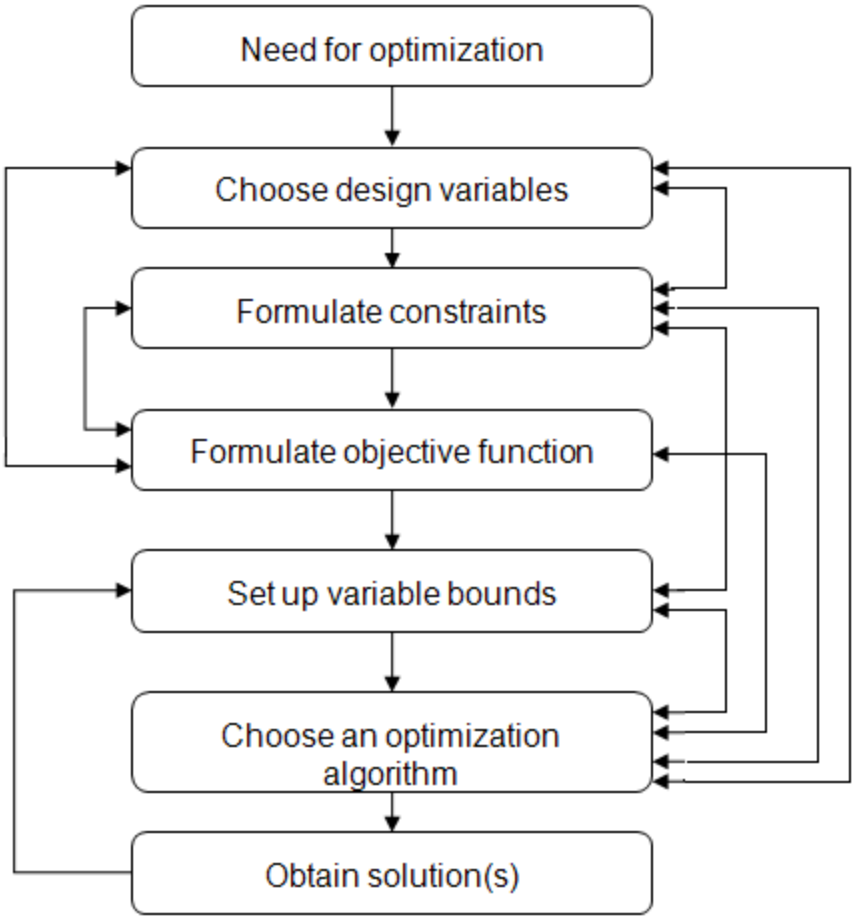

Fig.1 A flowchart of the optimal design procedure.

Design variables:

The formulation of an optimization problem begins with identifying the underlying design variables, which are primarily varied during the optimization process. A design problem usually involves many design parameters, of which some are highly sensitive to the proper working of the design. These parameters are called design variables in the parlance of optimization procedures. Other (not so important) design parameters usually remain fixed or vary in relation to the design variables.

The first thumb rule of the formulation of an optimization problem is to choose as few design variables as possible. The outcome of that optimization procedure may indicate whether to include more design variables in a revised formulation or to replace some previously considered design variables with new design variables.

Constraints:

The constraints represent some functional relationships among the design variables and other design parameters satisfying certain physical phenomenon and certain resource limitations. The nature and number of constraints to be included in the formulation depend on the user. Constraints may have exact mathematical expressions or not.

For example, maximum stress is a constraint of a structure. If a structure has regular shape they have an exact mathematical relation of maximum stress with dimensions. But incase irregular shape, finite element simulation software may be necessary to compute the maximum stress.

The following two types of constraints emerge from most considerations:

1. Inequality type constraints.
2. Equality type constraints.

Inequality constraints state that the functional relationships among variables are either greater than, smaller than or equal to, a resource value.

Example:

The stress $\sigma(x)$ developed anywhere in a component must be smaller than or equal to the allowable strength $(S_{allowable})$ of the material.

$$\sigma(x) \leq S_{allowable}$$

Some constraints may be of greater-than / equal-to type. For example, the natural frequency $(f(x))$ of a system to be greater than 2 Hz or by notation $f(x) \geq 2$.

Equality constraints state that functional relationships should exactly match a resource value.

Example:

The deflection $\delta(x)$ of a point in the component must be exactly equal to 5 mm. Then $\delta(x) = 5$.

It is very difficult to handle the equality constraints in the algorithms. In such cases, equality constraint is relaxed by including two inequality constraints as given below.

Example:

Previously $\delta(x) = 5$

Now it is changed to inequality constraints as given below:

$\delta(x) \geq 4$,

$\delta(x) \leq 6$.

Objective functions:

The next task in the formulation procedure is to find the objective function in terms of the design variables and other problem parameters. The common engineering objectives involve minimization of overall cost of manufacturing or minimization of overall weight of a component or maximization of total life of a product or others.

Although most of the objectives can be quantified (expressed in mathematical form), there are some objectives (such as aesthetic aspect of a design, ride characteristics of a car suspension design and reliability of a design) that may not be possible to formulate mathematically. In such a case an approximating mathematical expression is used.

In real world optimization, there could be more than one objective that the designer may want to optimize simultaneously. The multiple objective optimization algorithms are complex and computationally expensive. Therefore the most important objective is chosen as the objective function and the other objectives are included as constraints by restricting their values within a certain range.

For example, consider optimal truss structure design problem. The designer may be interested in minimizing the overall weight of the structure and simultaneously be concerned in minimizing the deflection of a specific point in the truss. In the optimal problem formulation, the designer may like to use the weight of the truss (as a function of the cross sections of the members) as the objective function and have a constraint with the deflection of the concerned point to be less than a specific limit.

The objective function can be of two types. Either it is to be maximized or it has to be minimized. Usually the optimization algorithms were written for minimization problems or maximization problems. Although in some algorithms, some minor structural changes would enable to perform either minimization (or) maximization; this requires extensive knowledge of the algorithm.

The duality principle helps by allowing the same algorithm to be used for minimization or maximization with a minor change in the objective function instead of a change in the entire algorithm. If the algorithm is for solving a minimization problem, it can be easily changed to a maximization problem by multiplying the objective function by -1 and vice versa.

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 the search algorithm with in these bounds.

In general, all *N* design variables are restricted to lie within the minimum and the maximum bounds as follows.

$$x_i^{(L)} \leq x_i \leq x_i^{(U)} \quad \text{for i = 1, 2, 3, … } N. \qquad (1)$$

In any given problem, the determination of the variables bounds $x_i^{(L)}$ and $x_i^{(U)}$ 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

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 be adjusted and optimization algorithm may be simulated again.

After the above four tasks are completed, the optimization problem can be mathematically written in a special format, known as nonlinear programming (NLP) format.

General format:

Denoting the design variables as a column vector $x = (x_1, x_2 \ldots x_N)^T$, the objective function as a scalar quantity $f(x)$, J inequality constraints as $g_j(x) \geq 0$ and K equality constraints as $h_k(x) = 0$, we write the NLP problem:

Minimize $f(x)$ Subject to,

$g_j(x) \geq 0$ $j = 1, 2, 3, \ldots\ldots J;$

$h_k(x) = 0$ $k = 1, 2, 3, \ldots\ldots K;$

$x_i^{(L)} \leq x_i \leq x_i^{(U)}$ $i = 1, 2, 3, \ldots N.$

Example:1 Optimal design of a truss structure

Consider the seven bar truss structure shown in the Fig.2. The loading is also shown in the figure. The length of the members $AC = CE = l = 1$m

Optimize,

1. Topology of the truss structure (the connectivity of the elements in a truss).

2. Once optimal layout is known, cross section of every element is another optimization problem.

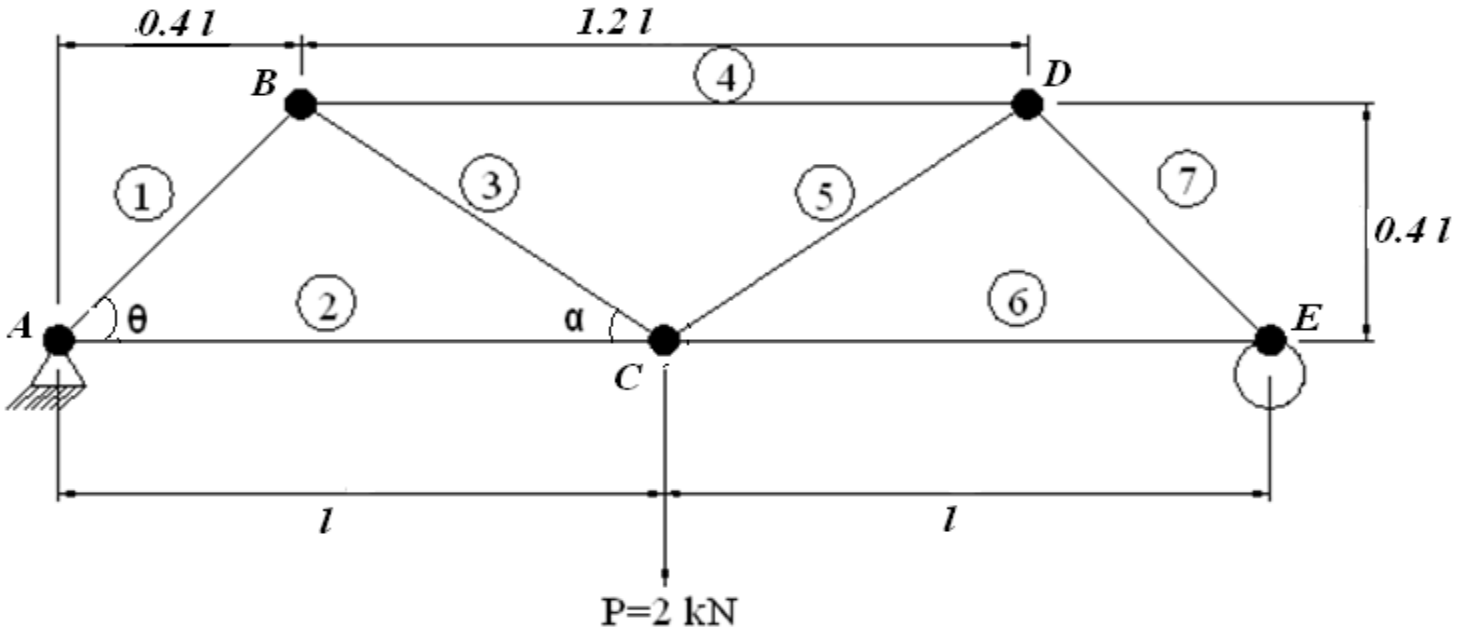

Fig.2. A typical seven bar truss structure

Since connectivity of truss is given, the cross-sectional area and material properties of the members are the design parameters. We choose cross sectional area of the members as the design variables. Using the symmetry of the truss,

$A_7 = A_1$; $A_6 = A_2$; $A_3 = A_5$

Thus, there are practically four design variables (A_1 to A_4).

Formulation of the constraints:

The truss carry the given load $P = 2\ kN$, the tensile and compressive stress generated in each member must not be more than the corresponding allowable strength S_{yt} and S_{yc} of the material.

Let us assume, $S_{yt} = S_{yc} = 500$ MPa and modulus of elasticity $E = 200$ GPa. Axial forces in each members of the truss are

Member AB ⇒ – 0.5 Pcsc θ; Member BC⇒ + 0.5 Pcsc α;

Member AC ⇒ + 0.5 Pcot θ; Member BD ⇒ – 0.5 P(cot θ + cot α);

Now, the axial stress can be calculated by dividing the axial load by the cross-sectional area of that member. Thus, the first set of constraints can be written as

$$\frac{P\csc\theta}{2A_1} \le S_{yc},$$

$$\frac{P\cot\theta}{2A_2} \le S_{yt},$$

$$\frac{P\csc\alpha}{2A_3} \le S_{yt},$$

$$\frac{P}{2A_4}(\cot\theta + \cot\alpha) \le S_{yc}.$$

In the above structure, *tan θ = 1.0* and *tan α = 2/3*. The other set of constraints arises from the stability consideration of the compression members *AB, BD,* and *DE.* According to the Euler buckling conditions for the axial load in members *AB* and *BD:*

$$\frac{P}{2\sin\theta} \le \frac{\pi E A_1^2}{1.281 l^2},$$

$$\frac{P}{2}(\cot\theta + \cot\alpha) \le \frac{\pi E A_4^2}{5.76 l^2}.$$

In most structures, deflection is a major consideration. In the above truss structure, let us assume that the maximum vertical deflection at C is δmax = 2 mm. By using Castigliano's theorem, we obtain the deflection constraint:

$$\frac{Pl}{E}\left(\frac{0.566}{A_1} + \frac{0.500}{A_2} + \frac{2.236}{A_3} + \frac{2.700}{A_4}\right) \le \delta_{\max}$$

In this problem, we are interested in minimizing the weight of the truss structure. Since we assumed the same material for all members, the minimization of the total volume of material will yield the same optimal solution as the minimization of the total weight. Thus, we write the objective function as

$$Minimize \qquad 1.132A_1l + 2A_2l + 1.789A_3l + 1.2A_4l$$

The next task is to set lower and upper bounds for the four cross sectional areas. We may choose to make all four areas lie between 10 and 500 mm^2. Thus the variable bounds are as

$$10\text{x}10^{-6} \le A_1, A_2, A_3, A_4 \le 500\,\text{x}10^{-6}$$

In the following, we present the above truss structure problem in NLP format.

$$\textit{Minimize} \qquad 1.132A_1 l + 2A_2 l + 1.789A_3 l + 1.2A_4 l$$

Subject to

$$S_{yc} - \frac{P}{2A_1 \sin\theta} \geq 0,$$

$$S_{yt} - \frac{P}{2A_2 \cot\theta} \geq 0,$$

$$S_{yt} - \frac{P}{2A_3 \sin\alpha} \geq 0,$$

$$S_{yc} - \frac{P}{2A_4}(\cot\theta + \cot\alpha) \geq 0,$$

$$\frac{\pi E A_1^2}{1.281 l^2} - \frac{P}{2\sin\theta} \geq 0,$$

$$\frac{\pi E A_4^2}{5.76 l^2} - \frac{P}{2}(\cot\theta + \cot\alpha) \geq 0,$$

$$\delta_{\max} - \frac{Pl}{E}\left(\frac{0.566}{A_1} + \frac{0.500}{A_2} + \frac{2.236}{A_3} + \frac{2.700}{A_4}\right) \geq 0,$$

$$10 \text{x} 10^{-6} \leq A_1, A_2, A_3, A_4 \leq 500 \text{x} 10^{-6}$$

Example: 2 Optimal design of a car suspension

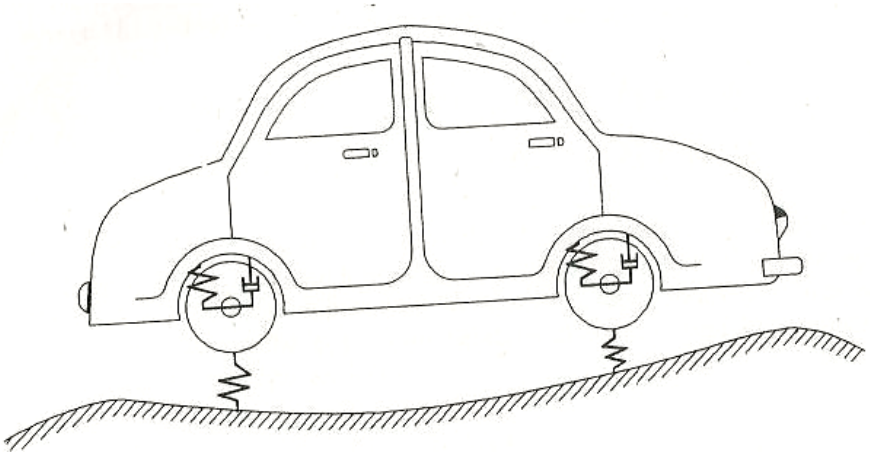

Fig.4. A two-dimensional model of a car suspension system

The comfort in riding a car largely depends on the suspension characteristics. The car body is usually supported by a suspension coil spring and a damper at each wheel (Figure 4). In order to formulate the optimal design problem, the first task is to identify the important design variables.

Sprung mass m_s, Front coil stiffness k_{fs},

Front unsprung mass m_{fu}, Rear coil stiffness k_{rs},

Rear unsprung mass m_{ru}, Front tyre stiffness k_{ft},

Rear damper coefficient α_r Rear tyre stiffness k_{rt},

Front damper coefficient α_f Axle-to-axle distance l,

Polar moment of inertia of the car J,

As long time is taken for the convergence of the optimization with all parameters as design variables, only four important parameters-front coil stiffness k_{fs}, rear coil stiffness k_{rs}, front damper coefficient α_f, and rear damper coefficient α_r are considered as design variables. Other design parameters are kept constant:

$m_s = 1000$ kg $l = 3.2$ m

$m_{fu} = 70$ kg $l_1 = 1.6$ m

$m_{ru} = 150$ kg $l_2 = 1.6$ m

$k_{ft} = 20$ kg/mm $J = 550$ kg-m^2

$k_{rt} = 20$ kg/mm

Using these parameters, differential equations governing the vertical motion of the unsprung mass at the front axle (q_1), the sprung mass (q_2), and the unsprung mass at the rear axle (q_4), and the angular motion of the sprung mass (q_3) are written (Fig. 5):

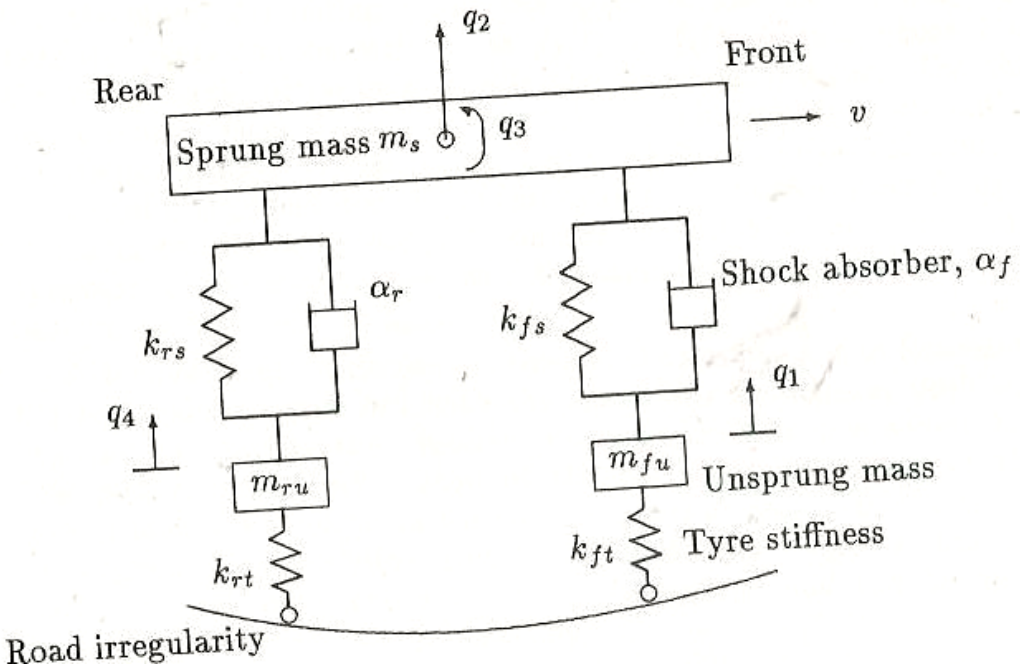

Fig.5. The dynamic model of the car suspension system. The above model has four degrees-of-freedom (q_1 to q_4)

$$\ddot{q}_1 = (F_2 + F_3 - F_1)/m_{fu}, \quad (9)$$

$$\ddot{q}_2 = -(F_2 + F_3 + F_4 + F_5)/m_s, \quad (10)$$

$$\ddot{q}_3 = [(F_4 + F_5)l_2 - (F_2 + F_3)l_1]/J, \quad (11)$$

$$\ddot{q}_4 = (F_4 + F_5 - F_6)/m_{ru}, \quad (12)$$

Where the forces F_1 to F_6 are calculated as follows:

$$F_1 = k_{ft}d_1, \quad F_2 = k_{fs}d_2, \quad F_3 = \alpha_f d_2, \quad (13)$$

$$F_4 = k_{rs}d_4, \quad F_5 = \alpha_r d_4, \quad F_6 = k_{rt}d_3.$$

The parameters d_1, d_2, d_3, and d_4 are the relative deformations in the front tyre, the front spring, the rear tyre, and the rear spring respectively. Figure 5 shows all the four degrees of freedom of the above system (q_1 to q_4). The relative deformations in springs and tyres can be written as follows:

$$
\begin{aligned}
d_1 &= q_1 - f_1(t), \\
d_2 &= q_2 + l_1 q_3 - q_1, \\
d_3 &= q_4 - f_2(t), \\
d_4 &= q_2 - l_2 q_3 - q_4.
\end{aligned}
\qquad (14)
$$

The time varying functions $f_1(t)$ and $f_2(t)$ are road irregularities as functions of time. Any function can be used for $f_1(t)$. For example, a bump can be modeled as $f_1(t) = A \sin \pi t / T$, where A is the amplitude of the bump and T is the time required to cross the bump. When a car is moving forward, the front wheel experiences the bump first, while the rear wheel experiences the same bump a little later, depending upon the speed of the car. Thus, the function $f_2(t)$ can be written as $f_2(t) = f_1(t - l/v)$, where l is the axle-to-axle distance and v is the speed of the car. For the above bump, $f_2(t) = A \sin(\pi (t - l/v)/T)$.

The coupled differential equations specified in equations (9) to (12) can be solved using a numerical integration technique (for example, a fourth-order Runge-Kutta method can be used) to obtain the pitching and bouncing dynamics of the sprung mass m_s. Equations can be integrated for a time range from zero to t_{max}.

After the design variables are chosen, the next task is to formulate the constraints associated with the above car suspension problem. In order to simplify the problem, we consider only one constraint. The jerk (the rate of change of the vertical acceleration of the sprung mass) is a major factor concerning the comfort of the riding passengers. The guideline used in car industries suggests that the maximum jerk experienced by the passengers should not be more than about 18 m/s^3. Mathematically,

$$max\ \ddot{q}_2'''(t) \leq 18$$

When the four coupled differential equations (9) to (12) are solved, the above constraint can be computed by numerically differentiating the vertical movement of the sprung mass (q_2) thrice with respect to time.

The next task is to formulate the objective function. In this problem, the primary objective is to minimize the transmissibility factor which is calculated as the ratio of the bouncing amplitude $q_2(t)$ of the sprung mass to the road excitation amplitude A. Thus, we write the objective function as

$$Minimize \quad \frac{max\ abs\, q_2(t)}{A}$$

The above objective function can be calculated from the solution of the four differential equations mentioned earlier. A minimum value of the transmissibility factor suggests the minimum transmission of road vibration to the passengers. This factor is also directly related to the ride characteristics as specified by the ISO standard.

Thus, the optimized design of the above car suspension system would provide the minimum transmissibility of the road vibration to the passengers with a limited level of jerk.

Finally, a minimum and maximum limit for each design variable can be set. This may require some previous experience with a car suspension design, but the following limits for the above car may include the optimal solution:

$$0 \le k_{fs}, k_{rs} \le 2kg/mm,$$

$$0 \le \alpha_f, \alpha_r \le 300kg/(m/s).$$

Thus, the above optimal car suspension design problem can be written in NLP form as follows:

$$Minimize \quad \frac{max\ abs\, q_2(t)}{A}$$

Subject to

$$18 - max\ q_2'''(t) \ge 0,$$

$$0 \le k_{fs}, k_{rs} \le 2,$$

$$0 \le \alpha_f, \alpha_r \le 300.$$

Example: 3 Optimal design of a transit schedule

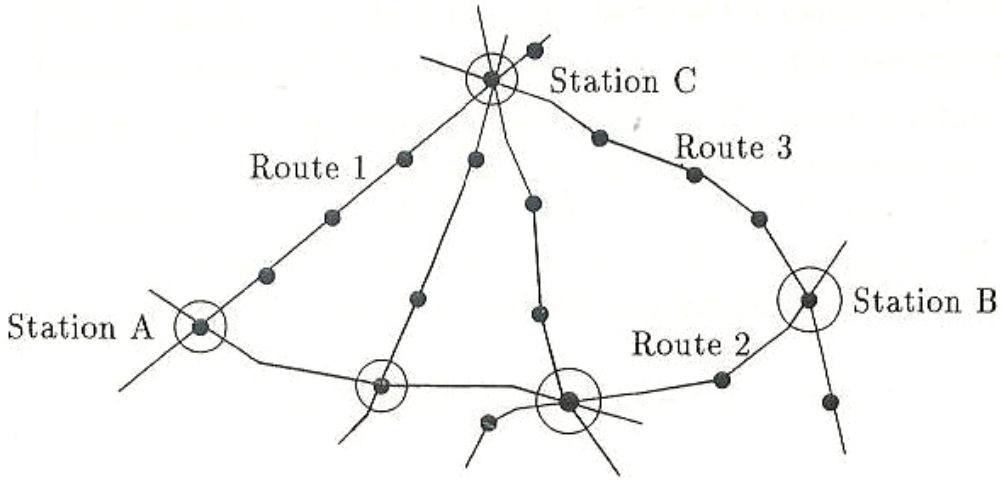

Fig. 2

Figure 2 shows a typical transit system network. The solid lines represent different routes, the points on the lines represent the stops and the circled intersections of the routes represent the transfer stations. The problem is to determine schedules for the routes such that the transit system provides the best Level of Service (LOS) to its passengers, within the resources available.

One of the good measures of the LOS is the amount of time passengers wait during their journey- the lesser the waiting time, the better is the LOS. On any transit network, passengers wait either to board the vehicle at the station of origin or they wait at a transfer station at which they transfer from one vehicle to another.

Let Initial Wait Time (IWT),

Transit Time (TT)

Schedule the vehicles such that (IWT + TT) is minimum.

The design variables are:

Arrival time :a_i^k k: vehicles

Departure time :d_i^k i : route.

If the routes are M and vehicle are K

The design variables are 2MK.

Minimum stopping time:

$(d_i^k - a_i^k) \geq s_{min}$ for all *i* and *k* (1)

Maximum stopping time:

$(d_i^k - a_i^k) \leq s_{max}$ for all *i* and *k* (2)

Maximum allowable transfer time:

No passenger on the transit network should have to wait more than a certain period of time *T* at any transfer station. This can be enforced by checking all possible differences between departure and arrival times and limiting those values to *T.* This constraint can be formulated by introducing a new set of variables $\delta_{i,j}^{k,l}$ between the *k-th* vehicle of the *i-th* route and the *l-th* vehicle of the *j-th* route.

These variables can take either a zero or one. A value of zero means that the transfer of passengers between those two vehicles is not feasible. A value of one means otherwise.

Consider the arrival and departure times of vehicles in two different routes at a particular station, as shown in Fig.3.

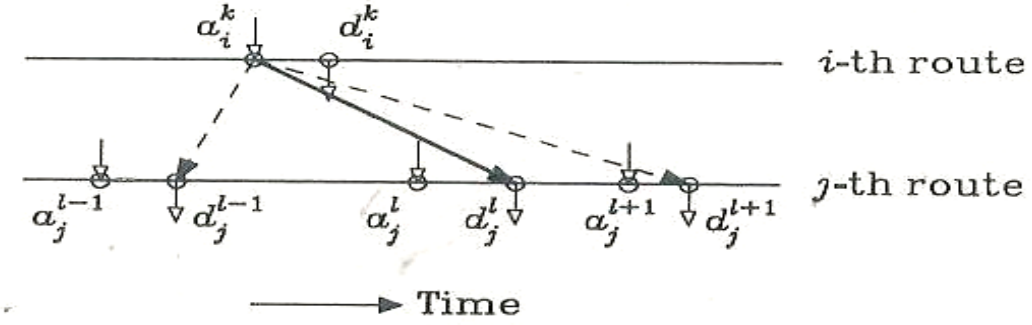

Fig.3. Transfers from k-th vehicle on the i-route to three consecutive vehicles in the j-th route

A passenger from the k_{th} vehicle in the i_{th} route can only transfer to a vehicle in the j_{th} *route* which is arriving at the station after a_i^k. According to the figure, the transfer of a passengers from the k^{th} vehicle in the i^{th} route is not possible to the $(l-1)_{th}$ vehicle in the j_{th} route, because the departure time of the latter vehicle d_j^{l-1} is earlier than a_i^k. Thus, the parameter $\delta_{i,j}^{k,l-1}$ takes the value zero, whereas the parameter $\delta_{i,j}^{k,l}$ takes a value one. In order to simplify the model, we assume that transfers to vehicles departing after l_{th} vehicle in the j_{th} route are also not possible. All parameters $\delta_{i,j}^{k,q}$ for $q = (l+1), (l+2), \ldots$ are also zero. Thus, between any two vehicles, the following condition must be satisfied:

$$(d_j^l - a_i^k)\delta_{i,j}^{k,l} \le T \qquad \textit{for all i, j, k and l.} \qquad (3)$$

The left side expression of the above condition is zero for those transfers that are not feasible. Since transfers only to the next available vehicle are assumed, only one $\delta_{i,j}^{k,l}$ for *(l = 1, 2, …)* is one and the rest all are zeros for fixed values of i, j, and k.

Mathematically,

$$\sum_l \delta_{i,j}^{k,l} = 1 \qquad \textit{for all i, j and k.} \qquad (4)$$

The introduction of the artificial variables $\delta_{i,j}^{k,l}$ makes the formulation easier, but causes a difficulty. Many optimization algorithms cannot handle discrete design variables efficiently. Since the artificial design variables $\delta_{i,j}^{k,l}$ can only take a zero or one, another set of constraints is added to enforce the binary values:

$$(d_j^l - a_i^k) + M(1 - \delta_{i,j}^{k,l}) \ge 0 \quad \textit{for all i, j, k and l.} \qquad (5)$$

Where M is a large positive number. The above constraint ensures that the variable $\delta_{i,j}^{k,l}$ always takes a value one whenever a transfer is possible and the value zero whenever transfer is not possible.

Maximum headway:

The headway between two consecutive vehicles should be less than or equal to the policy headway, h_i, or

$$(a_i^{k+1} - a_i^k) \leq h_i \quad \textit{for all i, and k.}$$

The objective function consists of two terms: the first term represents the total transfer time (TT) over all the passengers and the second term represents the initial waiting time (IWT) for all the passengers. The objective is to minimize the following function:

$$\sum_i \sum_j \sum_k \sum_l \delta_{i,j}^{k.l} (d_j^l - a_i^k) w_{i,j}^k + \sum_i \sum_l \int_0^{a_i^k - a_i^{k-1}} v_{i,k}(t)[(a_i^k - a_i^{k-1}) - t]dt \quad (6)$$

The parameter $w_{i,j}^{k}$ is the number of passengers transferring from the *k-th* vehicle of the *i-th* route to the *j-th* route. The first term is obtained by summing the individual transfer time $(d_j^l - a_i^k)$ over all passengers for all the vehicles for every pair of routes.The parameter $v^{i,k}(t)$ is the number of passengers arriving at the stop for the *k-th* vehicle in the *i-th* route at a given time *t.*Since the arrivaltime for passengers can be anywhere between $t = 0 \ \ to \ \ t = (a_i^k - a_i^{k-1})$(the headway), the initial waiting time also differs from one passenger to another.

For example, a passenger arriving at the stop just after the previous vehicle has left has to wait for the full headway time $(a_i^k - a_i^{k-1})$ before the next vehicle arrives. On the other hand, a passenger arriving at the stop later has to wait for a shorter time. The calculation of the second term assumes that passengers arrive at the stop during the time interval $a_i^{k-1} \ \ to \ \ a_i^k$

according to the known time-varying function $v_{i,k}(t)$, where t is measured from a_i^{k-1} .Then the quantity

$$\int_0^{a_i^k - a_i^{k-1}} v_{i,k}(t)[(a_i^k - a_i^{k-1}) - t]dt \qquad (7)$$

gives the sum of the initial waiting times for all passengers who board the *k-th* vehicle of the *i-th* route. We then sum it over all the routes and vehicles to estimate the network total of the IWT. Thus, the complete NLP problem can be written as follows:

Minimize

$$\sum_i \sum_j \sum_k \sum_l \delta_{i,j}^{k.l}(d_j^l - a_i^k) w_{i,j}^k + \sum_i \sum_l \int_0^{a_i^k - a_i^{k-1}} v_{i,k}(t)[(a_i^k - a_i^{k-1}) - t]dt \qquad (8)$$

Subject to,

$$s_{max} - (d_i^k - a_i^k) \geq 0 \qquad \text{for all } i, \text{ and } k,$$

$$(d_i^k - a_i^k) - s_{min} \geq 0 \qquad \text{for all } i, \text{ and } k,$$

$$T - (d_j^l - a_i^k)\delta_{i,j}^{k,l} \geq 0 \qquad \text{for all } i, j, k \text{ and } l,$$

$$(d_j^l - a_i^k) + M(1 - \delta_{i,j}^{k,l}) \geq 0 \qquad \text{for all } i, j, k \text{ and } l,$$

$$h_i - (a_i^{k+1} - a_i^k) \geq 0 \qquad \text{for all } i, \text{ and } k,$$

$$\sum_l \delta_{i,j}^{k,l} = 1 \qquad \text{for all } i, j, \text{ and } k.$$

In the above NLP problem, the variables $\delta_{i,j}^{k,l}$ are binary variables taking only a value zero or a one and other variables a_i^k and d_i^k are real-valued

3. Optimization Algorithms

The formulation of engineering design problems differ from problem to problem. They are

(i) Linear terms for constraints and objective function

(ii) Non linear terms for constraints and objective function.

The terms are not explicit functions of the design variables. No single optimization algorithm which will work in all optimization problems equals efficiently.

For the sake of clarity, the optimization algorithms are classified into a number of groups, which are now briefly discussed.

(a) Single-variable optimization algorithms.

These algorithms are classified into two categories

i. Direct methods

ii. Gradient based methods

Direct methods do not use any derivative information of the objective function; only objective function values are used to guide the search process. However, gradient-based methods use derivative information (first and/ or second order) to guide the search process.

Although engineering optimization problems usually contain more than one variable, single-variable optimization algorithms are mainly used as unidirectional search methods in multivariable optimization algorithms.

(b) Multi- variable optimization algorithms.

These algorithms demonstrate how the search for the optimum point progresses in multiple dimensions. Depending on whether the gradient information is used or not used, these algorithms are also classified into direct and gradient-based techniques.

(c) Constrained optimization algorithms.

These algorithms use the single variable and multivariable optimization algorithms repeatedly and simultaneously maintain the search effort inside the feasible search region. These algorithms are mostly used in engineering optimization problems.

(d) Specialized optimization algorithms.

Two of these algorithms - integer programming and geometric programming - are often used in engineering design problems. Integer programming methods can solve optimization problems with integer design variables. Geometric programming methods solve optimization problems with objective functions and constraints written in a special form.

(e) Non-traditional optimization algorithms.

There are two algorithms which are nontraditional, these are:

a) Genetic algorithms
b) Simulated annealing.

4.0 Single-variable optimization algorithms

The algorithms described in this section can be used to solve minimization problems of the following type:

Minimize f(x)

Where f(x) is the objective function and x is a real variable. The purpose of an optimization algorithm is to find a solution x, for which the function f(x) is minimum.

4.1 Optimality criteria

There are three different types of optimal points are:

(i) Local Optimal point:

A point or solution x* is said to be a local optimal point, if no point in the neighbourhood has a function value smaller than f(x*).

(ii) Global Optimal point:

A point or solution x** is said to be a global optimal point, if no point in the entire search space has a function value smaller than f(x**).

(iii) Inflection point:

x* is an inflection point if

f(x*) increases locally as x* increases & decreases locally as x* reduces

or f(x*) decreases locally as x* increases and increases locally as x* decreases

Let the objective function f(x) is the chosen search space

f '(x) and f ''(x) are first and second derivatives

A point x is a minimum if f'(x) = 0 & f''(x) > 0.

If f ‘(x) = 0, the point is either a minimum, a maximum or an inflection point

Suppose at point x*, the first derivative is zero and the first non-zero higher order derivative is denoted by n, then

- If n is odd, x* is an inflection point
- If n is even, x* is a local optimum

(i) If the derivative is +ve, x* is a local minimum

(ii) If the derivative is –ve, x* is a local maximum

Example: 4

Consider f (x) = x^3, optimal point x = 0 as shown in Fig.6.

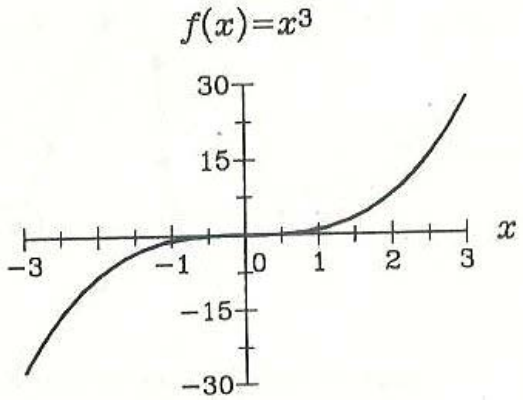

Fig.6. The function f(x) = x^3

From the figure, we can see that point x = 0 is an inflection point as

f(x) increases for x ≥ 0

decreases for x ≤ 0

Using the sufficient conditions

$$f'(x=0) = 3x^2\Big|_{x=0} = 0$$

$$f''(x=0) = 6x\Big|_{x=0} = 0$$

$$f'''(x=0) = 6\Big|_{x=0} = 6(Nonzero \quad value)$$

Third derivative, n =3 is odd, hence x = 0 is an inflection point.

Example : 5

Consider $f(x) = x^4$, optimal point x = 0 as shown in Fig.6a.

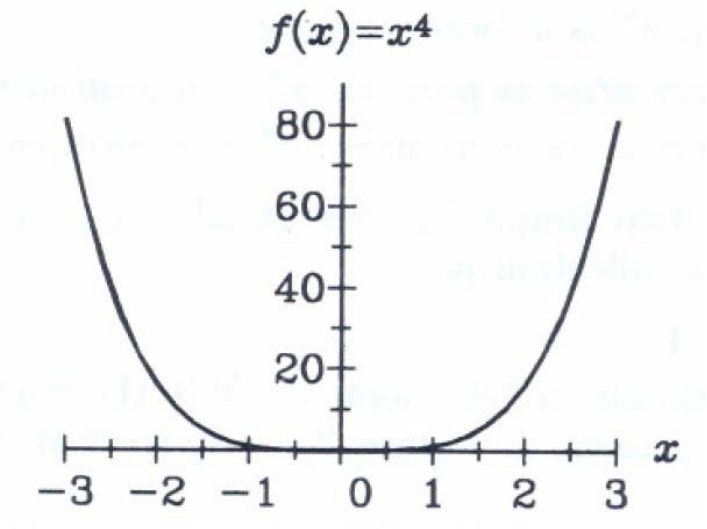

Fig.6a. The function $f(x) = x^4$

From the figure, we can see that the point x = 0 is a minimal point.

Using sufficient conditions

$$f'(x=0) = 4x^3\Big|_{x=0} = 0$$

$$f''(x=0) = 12x^2\Big|_{x=0} = 0$$

$$f'''(x=0) = 24x\Big|_{x=0} = 0$$

$$f''''(x=0) = 24\Big|_{x=0} = 24(Nonzero \quad value)$$

Fourth order derivative is positive, n = 4 is even, hence x = 0 is a local minimum point.

4.2 Bracketing Methods:

The minimum of a function is found in two phases. Initially an approximate method is used to find a lower and an upper bound of the minimum. Next, a sophisticated technique is used to search within these two limits to find the optimal solution.

(a) Exhaustive search method

It is the simplest of all search methods. The optimum of a function is bracketed by calculating the function values at a number of equally spaced points(Fig.7).

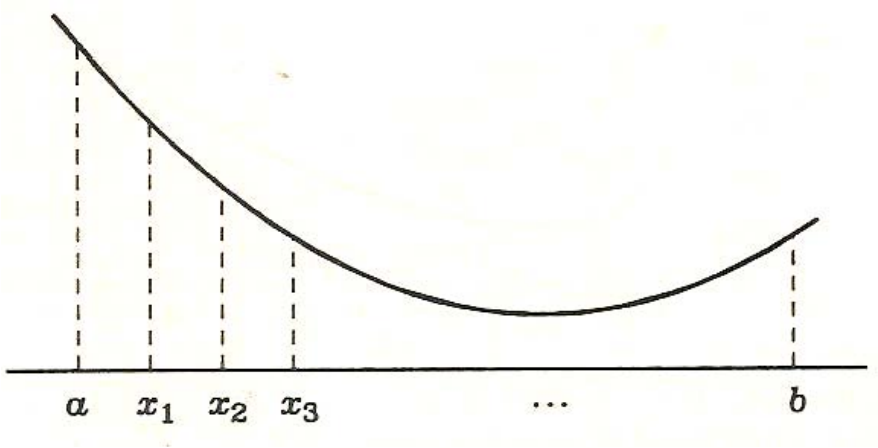

Fig.7 The exhaustive search method that uses equally spaced points

Usually the search begin from a lower bound on the variable and three consecutive function values are compared at a time based on the assumption of unimodality of the function. Based on the outcome of comparison, the search is either terminated or continued by replacing one of the three points with a new point.

Algorithm:

Step 1.

Set

$x_1 = a,\ \Delta x = (b-a)/n$ (*n* is number of intermediate points)

$x_2 = x_1 + \Delta x,\ x_3 = x_2 + \Delta x$

Step 2

If $f(x_1) \geq f(x_2) \leq f(x_3)$, the minimum point lies between (x_1, x_3).
Hence, **terminate.**

Else $x_1 = x_2,\ x_2 = x_3,\ x_3 = x_2 + \Delta x$

Step 3

Is $x_3 \leq b$? If yes, goto Step 2.

Else no minimum exists in (a,b) or a boundary point (a or b) is the minimum point.

Example:6

Minimize $f(x) = x^2 + 54/x$ in the interval (0,5)

A plot of function is shown in Fig.8

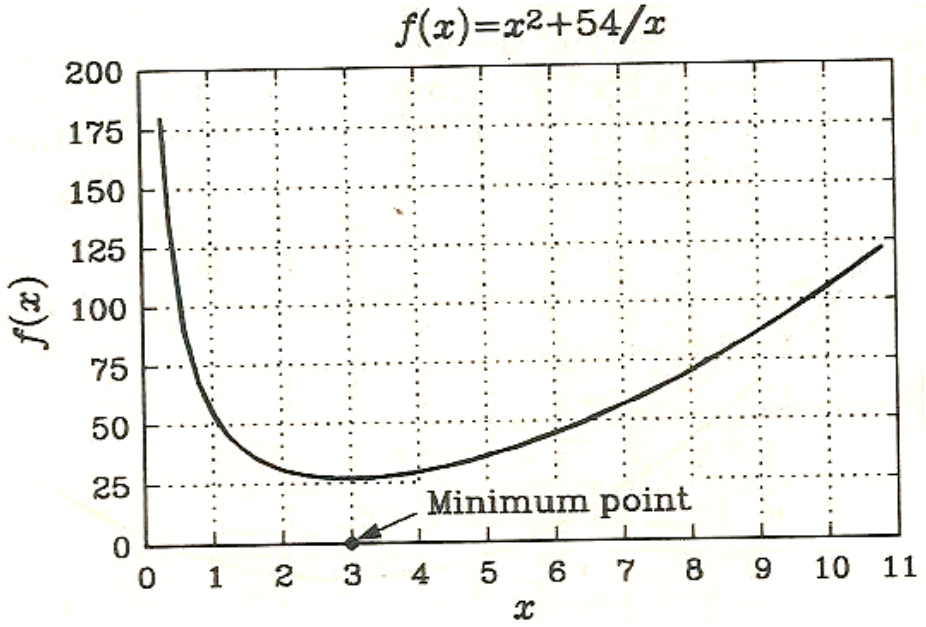

Fig.8 The unimodal, single-variable function used in the exercise problems.

The plot shows that the minimum lies at x =3

$$f(3) = 27$$

$$f'(3) = 0$$

$$f''(3) = 6$$

According to sufficiency conditions, x =3 is a local minimum.

Now consider n = 10 for exhaustive search.

Step 1

According to the parameter chosen,

$$x_1 = a = 0 \ and \ b = 5$$

$$\therefore \Delta x = (5-0)/10 = 0.5$$

We set

$$x_2 = 0 + 0.5 = 0.5$$

and

$$x_3 = 0.5 + 0.5 = 1.0$$

Step 2 Computing function values at various points, we have

$$f(0) = \infty. \qquad f(0.5) = 108.25 \qquad f(1.0) = 55.00$$

$$f(x_1) > f(x_2) > f(x_3)$$

Thus, the minimum does not lie between (0,1)

$$\therefore Set \qquad x_1 = 0.5\,, \qquad x_2 = 1.0, \qquad x_3 = 1.5$$

$$f(1.5) = 38.25$$

$f(x_1) > f(x_2) > f(x_3)$ and minimum does not lie between (0.5,1.5)

Repeat the process till $f(2.5) = 27.85$

$$f(3.0) = 27.00$$

$$f(3.5) = 27.68$$

$$f(x_1) > f(x_2) < f(x_3)$$

Solution lies between (2.5, 3.5).

The accuracy solution $2(a-b)/n = 2(5-0)/10 = 1.0$

If more accurate solution is required, divide into more number of parts by increasing *n*.

(b) Bounding phase method

Step 1

Choose an initial guess $x^{(0)}$ and an increment Δ. Set k =0

Step 2

If $f(x^0 - |\Delta x|) \geq f(x^0) \geq f(x^0 + |\Delta x|)$, then Δ $is + ve$

$f(x^0 - |\Delta x|) \leq f(x^0) \leq f(x^0 + |\Delta x|)$, then Δ $is - ve$

Else goto Step 1

Step 3

Set $x^{(k+1)} = x^{(k)} + 2^k \Delta$.

Step 4

If $f(x^{(k+1)}) < f(x^{(k)})$, set k = k+1 and goto Step 3

Else, the minimum lies in the interval $(x^{(k-1)}, x^{(k+1)})$ and **terminate**

If Δ is large, accuracy is poor.

Example 7

Find minimum of $f(x) = x^2 + \frac{54}{x}$ using bounding phase method

1. Choose an initial guess $x(0) = 0.6$ and increment $\Delta = 0.5$ set k = 0.
2. Calculate three function values

$$f(x^{(0)} - |\Delta|) = f(0.6 - 0.5) = 540.010$$

$$f(x^{(0)}) = f(0.6) = 90.360$$

$$f(x^{(0)} + |\Delta|) = f(0.6 + 0.5) = 50.301$$

We observe that

$$f(0.1) > f(0.6) > f(1.1) \qquad \therefore set \Delta = +0.5$$

3. Next guess: $x^{(1)} = x^{(0)} + 2^0 \Delta = 1.1$
4. $f(x^{(1)}) = 50.301 < f(x^{(0)})$ ∴ set k = 1 and goto Step 3

3. Next guess: $x^{(2)} = x^{(1)} + 2^1\Delta = 1.1 + 2(0.5) = 2.1$

4. $f(x^{(2)}) = 30.124 < f(x^1)$ **∴** set k = 2 and goto Step 3

3. Next guess: $x^{(3)} = x^{(2)} + 2^2\Delta = 2.1 + 4(0.5) = 4.1$

4. $f(x^{(3)}) = 29.981 < f(x^{(2)})$ **∴** set k = 3 and goto Step3.

3. Next guess:

4. $f(x^{(4)}) = 72.277 > f(x^{(3)}).$ Thus terminate with interval (2.1, 8.1)

with $\Delta = 0.5,$ the obtained bracketing is poor. Functions evaluated are 7.

$\Delta = 0.001,$ the obtained interval is (1.623, 4.695). Functions evaluated are 15.

4.3. Region elimination methods:

Once the minimum point is bracketed, a more sophisticated algorithm is used to improve the accuracy of the solution. Region elimination methods are used for this purpose. The fundamental rule for region elimination method is as follows:

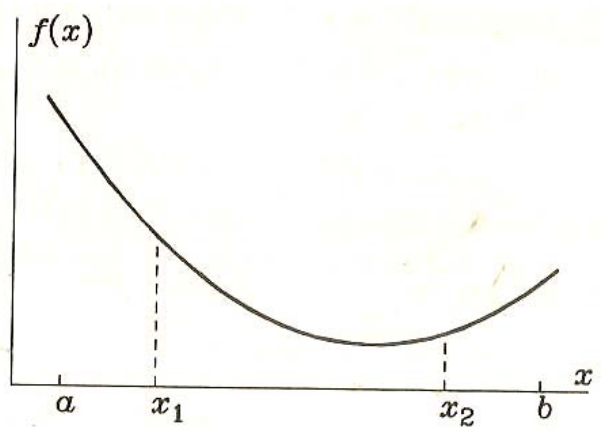

Fig.9. A typical single-variable unimodal function with function values at two distinct points

Consider a unimodal function drawn in Fig.9. The two points x_1 and x_2 lie in the interval (a,b) and satisfy $x_1 < x_2$. For minimization, the following conditions apply

- If $f(x_1) > f(x_2)$ then the minimum does not lie in (a, x_1)
- If $f(x_1) < f(x_2)$ then the minimum does not lie in (x_2, b)
- If $f(x_1) = f(x_2)$ then the minimum does not lie in (a, x_1) and (x_2, b)

(a) Interval halving method:

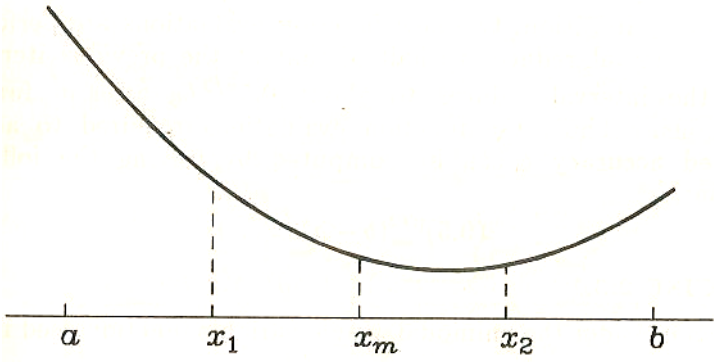

Fig.10 Three points x_1, x_m, and x_2 used in the interval halving

Fig.10 shows a region in the interval (a, b). Three points divide the search space into four regions. The fundamental rule for region elimination is used to eliminate a portion of search space based on function values at three chosen points.

x_1, x_m, x_2 are three search points. Two of the function values are compared at a time and some region is eliminated. The three possibilities are

(i) If $f(x_1) < f(x_m)$ minimum cannot lie beyond ***x_m*** and reduce the interval from *(a, b)* to *(a, **x_m**)* search space is reduced by 50 percent.

(ii) If $f(x_1) > f(x_m)$, minimum cannot lie in the interval *(a, **x_1**).*The point x_1 is ¼ in search space, hence reduction is 25 percent.

Then compare and to eliminate further 25 percent of search space. Continue the process till small enough interval is found.

Algorithm:

1. Choose lower bound *a* and upper bound *b* and a small value $\in$ for desired accuracy. $x_m = (a+b)/2, L_0 = L = b-a.$ Compute $f(x_m)$
2. Set $x_1 = a + \frac{L}{4},\ x_2 = b - \frac{L}{4}.$ Compute $f(x_1)$ and $f(x_2)$
3. If $f(x_1) < f(x_m),$ set $b = x_m,\ x_m = x_1.$Goto Step 5, else goto Step 4
4. If $f(x_2) < f(x_m),$ set $a = x_m,\ x_m = x_2.$ Goto Step 5, else$a = x_1,\ b = x_2;$ goto Step 5
5. Calculate L = $b-a$. If $|L| < \in,$ terminate. Else goto step 2.

(a) Fibonacci search method:

The search interval is reduced according to Fibonacci numbers.

$F_n = F_{n-1} + F_{n-2}$ where n=2,3,4,….. and $F_0 = F_1 = 1$

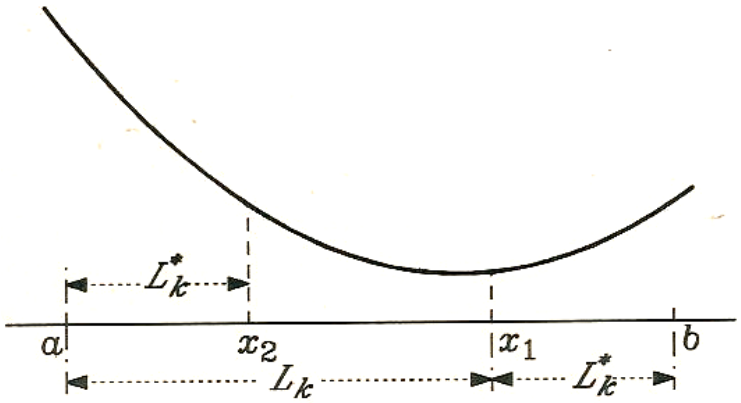

Fig.11. Fibonacci search points(x_1 and x_2)

Algorithm:

1. Choose a lower bound *a* and an upper bound *b*. Set L = b-a. Assume the desired number of function evaluations to be n. Set k=2
2. Compute $L_k^* = (F_{n-k+1}/F_{n+1})L$ Set $x_1 = a + L_k^*$ and $x_2 = b - L_k^*$

3. Compute one of $f(x_1)$ or $f(x_2)$, which was not evaluated earlier. Use the fundamental region elimination rule to eliminate a region. Set new **a** and **b.**

4. Is k = n? If no, set k = k+1 and goto Step 2. Else terminate.

(c) Golden section search method

Difficulties with Fibonacci search method:

(i) Fibonacci numbers have to be calculated and stored.

(ii) At every iteration the proportion of the eliminated region is not the same.

In golden section search method, the search space (a, b) is first linearly mapped to a unit interval search space (0,1). Two points at τ from either end of search space are chosen so that at every iteration the eliminated region is $(1-\tau)$ to that in the previous iteration (Fig.12). This can be achieved by equating $(1-\tau)=(\tau \times \tau)$ This yields the golden number $\tau = 0.618$

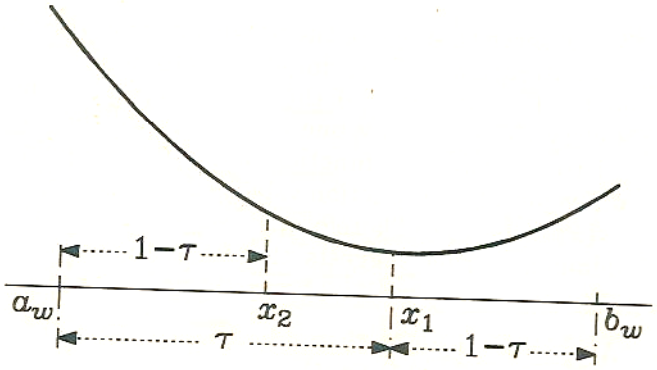

Fig.12. The points (*x1* and *x2*) used in the golden section

Algorithm:

1. Choose a lower bound a and an upper bound b. Choose a small value $\in$.Normalize the variable x by using the equation $w = (x-a)/(b-a)$. Thus $a_w = 0, b_w = 1, \quad and \quad L_w = 1.$, Set k=1.
2. Set $w_1 = a_w + (0.618)L_w$ and $w_2 = b_w - (0.618)L_w$.Compute $f(w_1) \quad o \quad r \; f(w_2),$ depending on whichever of the two was not evaluated earlier. Use the fundamental region elimination rule to eliminate a region. Set new a_w and b_w
3. Is $|L_w| < \in ?$ If no, set k=k+1, go to Step 2. Else **terminate.**

4.4 Gradient based methods

Despite the difficulty of finding the derivatives, they are popular because of their effectiveness.

(a) Newton- Raphson method

Considering Taylor series expansion $x^{(n+1)} = x^{(n)} - \frac{f'(x^{(n)})}{f''(x^{(n)})}$

Algorithm:

1. Choose initial guess $x^{(1)}$ and small value $\in$. Set k = 1.Compute $f'(x^{(1)})$.
2. Compute $f''(x^{(k)})$.
3. Calculate $x^{(k+1)} = x^{(k)} - \left(\frac{f'(x^{(k)})}{f''(x^{(k)})} \right)$ Compute $f'(x^{(k+1)})$.
4. If $\left| f'(x^{(k+1)}) \right| < \in$, Terminate. Else k = k+1 and go to Step 2.

Convergence will depend on initial guess value and nature of the objective function.

Example 8

Minimize $f(x) = x^2 + \frac{54}{x}$ using Newton-Raphson method.

1. Choose initial guess $x^{(1)} = 1.$ and $\in = 10^{-3},\ k = 1$

$$f'(x^{(n)}) = \frac{f(x^{(n)} + \Delta x^{(n)}) - f(x^{(n)} - \Delta x^{(n)})}{2\Delta x^{(n)}}$$

$$f''(x^{(n)}) = \frac{f(x^{(n)} + \Delta x^{(n)}) - 2f(x^{(n)}) + f(x^{(n)} - \Delta x^{(n)})}{(\Delta x^{(n)})^2}$$

$\Delta x^{(n)}$ is parameter, usually taken a small value (0.001).

$$f'(x^{(n)}) = -52.005$$

$$f''(x^{(n)}) = 110.011$$

2.

$$x(2) = x(1) - \frac{f'(x^{(1)})}{f''(x^{(1)})}$$

$$= 1 - \left(\frac{-52.005}{110.011} \right) = 1.473$$

The iterative values of x, for various k's are tabulated below.

k	$x^{(k)}$	$f'(x^{(k)})$	$f''(x^{(k)})$
1	1	-52.005	110.011
2	1.473	-21.944	35.796
3	2.086	-8.239	13.899
4	2.679	-2.167	
-			
7	3.0001	$-4(10)^{-8}$	

f ' is small enough to terminate.