Introduction to Engineering Optimization (ME6806)

Dr. Yogesh Kumar
Assistant Professor
Mechanical Engineering Department
National Institute of Technology Patna
Bihar - 800 005, India
yogesh.me@nitp.ac.in

Introduction

- This part of the course deals with techniques that are applicable to the solution of the constrained optimization problem:

$$\text{Find } \mathbf{X} \text{ which minimizes } f(\mathbf{X})$$

subject to

$$g_j(\mathbf{X}) \leq 0, \qquad j = 1,2,\ldots,m$$

$$h_k(\mathbf{X}) = 0, \qquad k = 1,2,\ldots,p$$

- There are many techniques available for the solution of a constrained nonlinear programming problem. All the methods can be classified into two broad categories: direct methods and indirect methods.

- In the direct methods, the constraints are handled in an explicit manner, whereas in most of the indirect methods, the constrained problem is solved as a sequence of unconstrained minimization problems.

Characteristic of a constrained problem

Direct Methods	Indirect Methods
Random search methods	Transformation of variables technique
Heuristic search methods	Sequential unconstrained minimization techniques
Complex method	Interior penalty function method
Objective and constraint approximation methods	Exterior penalty function method
Sequential linear programming method	Augmented Lagrange multiplier method
Sequential quadratic programming method	
Methods of feasible directions	
Zoutendijk's method	
Rosen's gradient projection method	
Generalized reduced gradient method	

Characteristic of a constrained problem

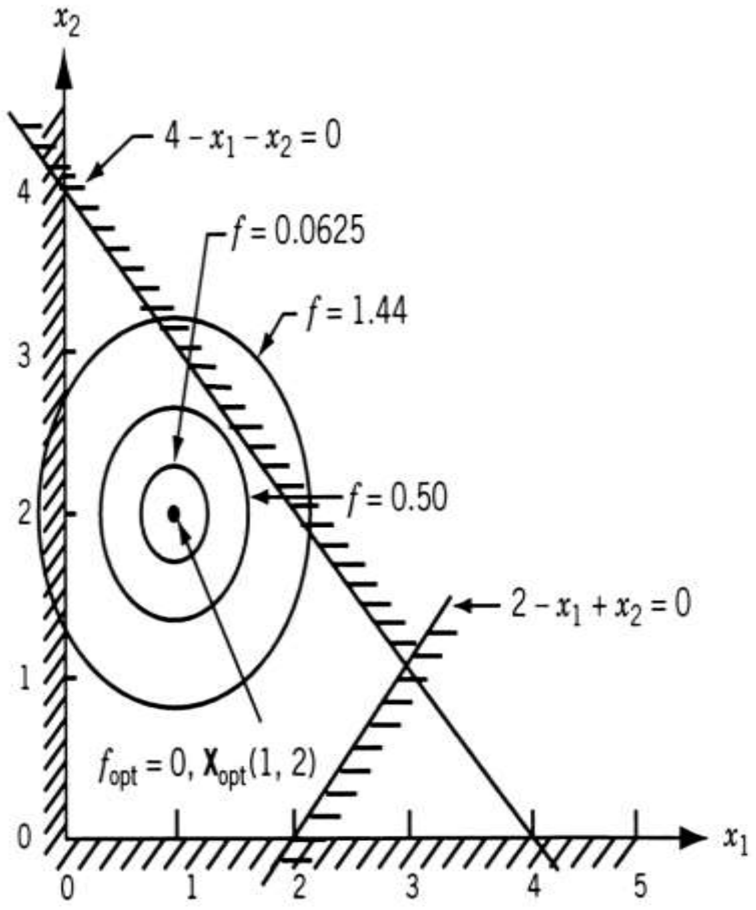
In the presence of the constraints, an optimization problem may have the following features:

- The constrainst may have no effect on the optimum point; that is, the constrained minimum is the same as the unconstrained minimum as shown in the figure. In this case, the minimum point **X*** can be found by making use of the necessary and sufficient conditions as follows:

$$\nabla f|_{\mathbf{X}^*} = \mathbf{0}$$

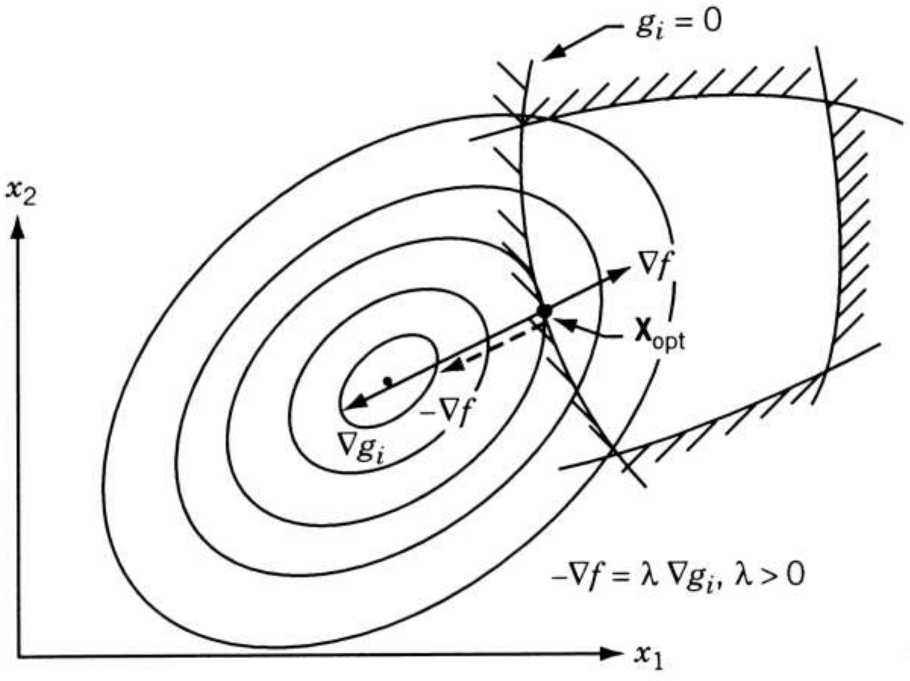
$$\mathbf{J}_{\mathbf{X}^*} = \left[\frac{\partial^2 f}{\partial x_i\, \partial x_j}\right]_{\mathbf{X}^*} = \text{positive definite}$$

- However to use these conditions one must be certain that the constraints are not going to have any effect on the minimum.



Characteristic of a constrained problem

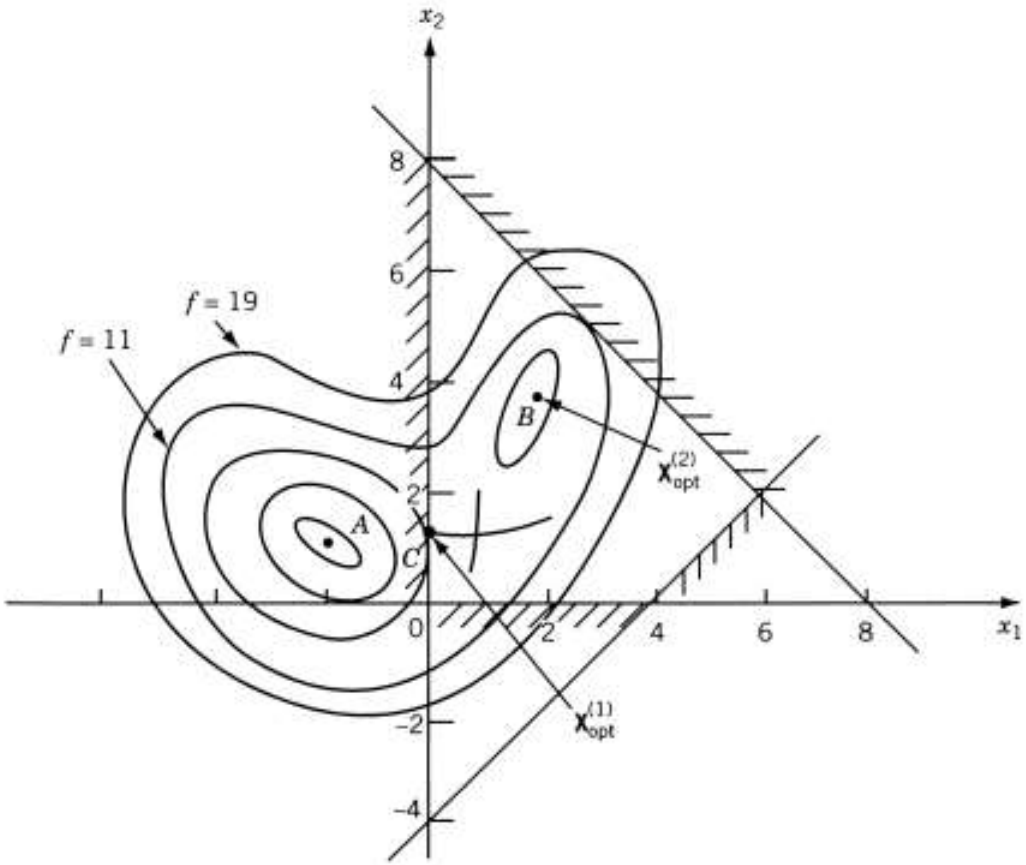
- The optimum, unique solution occurs on a constraint boundary as shown. In this case, the Kuhn-Tucker necessary conditions indicate that the negative of the gradient must be expressible as a positive linear combination of the gradients of the active constraints.



Constrained minimum occurring on a nonlinear constraint.

Characteristic of a constrained problem

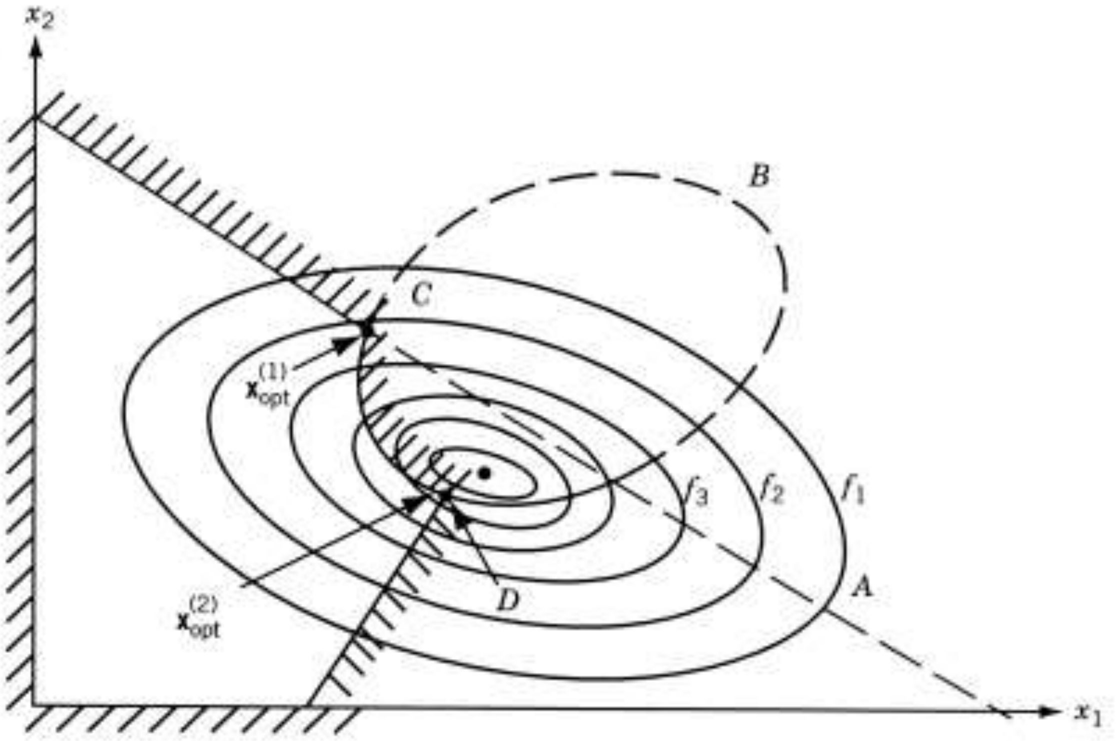
- If the objective function has two or more unconstrained local minima, the constrained problem may have multiple minima as shown in the figure.



Relative minima introduced by objective function.

Characteristic of a constrained problem

- In some cases, even if the objective function has a single unconstrained minimum, the constraints may introduce multiple local minima as shown in the figure.
- A constrained optimization technique must be able to locate the minimum in all the situations outlined above.



Relative minima introduced by constraints.

Direct Methods

RANDOM SEARCH METHODS

The random search methods described for unconstrained minimization can be used with minor modifications to solve a constrained optimization problem. The basic procedure can be described by the following steps:

1. Generate a trial design vector using one random number for each design variable.

2. Verify whether the constraints are satisfied at the trial design vector. Usually, the equality constraints are considered satisfactory whenever their magnitudes lie within a specified tolerance. If any constraint is violated, continue generating new trial vectors until a trial vector that satisfies all the constraints is found.

3. If all the constraints are satisfied, retain the current trial vector as the best design if it gives a reduced objective function value compared to the previous best available design. Otherwise, discard the current feasible trial vector and proceed to step 1 to generate a new trial design vector.

4. The best design available at the end of generating a specified maximum number of trial design vectors is taken as the solution of the constrained optimization problem.

Direct Methods

RANDOM SEARCH METHODS

- It can be seen that several modifications can be made to the basic procedure indicated above. For example, after finding a feasible trial design vector, a feasible direction can be generated (using random numbers) and a one-dimensional search can be conducted along the feasible direction to find an improved feasible design vector.
- Another procedure involves constructing an unconstrained function, F**(X)**, by adding penalty for violating any constraint as:

$$F(\mathbf{X}) = f(\mathbf{X}) + a \sum_{j=1}^{m} [G_j(\mathbf{X})]^2 + b \sum_{k=1}^{p} [H_k(\mathbf{X})]^2$$

where

$$[G_j(\mathbf{X})]^2 = [\max(0, g_j(\mathbf{X}))]^2$$

$$[H_k(\mathbf{X})]^2 = h_k^2(\mathbf{X})$$

indicate the squares of violations of inequality and equality constraints, respectively, and *a* and *b* are constants.

Direct Methods

RANDOM SEARCH METHODS

- The equation

$$F(\mathbf{X}) = f(\mathbf{X}) + a \sum_{j=1}^{m} [G_j(\mathbf{X})]^2 + b \sum_{k=1}^{p} [H_k(\mathbf{X})]^2$$

indicates that while minimizing the objective function f **(X)**, a positive penalty is added whenever a constraint is violated, the penalty being proportional to the square of the amount of violation. The values of the constants a and b can be adjusted to change the contributions of the penalty terms relative to the magnitude of the objective function.

- Note that the random search methods are not efficient compared to the other methods described in this chapter. However, they are very simple to program and are usually reliable in finding a nearly optimal solution with a sufficiently large number of trial vectors. Also, these methods can find near global optimal solution even when the feasible region is nonconvex.

Direct Methods

SEQUENTIAL LINEAR PROGRAMMING

- In the sequential linear programming (SLP) method, the solution of the original nonlinear programming problem is found by solving a series of linear programming problems.

- Each LP problem is generated by approximating the nonlinear objective and constraint functions using first-order Taylor series expansions about the current design vector $\mathbf{X}_i$.

- The resulting LP problem is solved using the simplex method to find the new design vector $\mathbf{X}_{i+1}$.

- If $\mathbf{X}_{i+1}$ does not satisfy the stated convergence criteria, the problem is relinearized about the point $\mathbf{X}_{i+1}$ and the procedure is continued until the optimum solution $\mathbf{X}^*$ is found.

Direct Methods

SEQUENTIAL LINEAR PROGRAMMING

- If the problem is a convex programming problem, the linearized constraints always lie entirely outside the deasible region. Hence the optimum solution of the approximating LP problem, which lies at a vertex of the new feasible region, will lie outside the original feasible region.

- However, by relinearizing the problem about the new point and repeating the process, we can achieve convergence to the solution of the original problem in a few iterations.

- The SLP method is also known as the *cutting plane method*.

Direct methods

SEQUENTIAL LINEAR PROGRAMMING

1. Start with an initial point $\mathbf{X}_1$ and set the iteration number as $i = 1$. The point $\mathbf{X}_1$ need not be feasible.

2. Linearize the objective and constraint functions about the point $\mathbf{X}_i$ as

$$f(\mathbf{X}) \approx f(\mathbf{X}_i) + \nabla f(\mathbf{X}_i)^T(\mathbf{X} - \mathbf{X}_i)$$

$$g_j(\mathbf{X}) \approx g_j(\mathbf{X}_i) + \nabla g_j(\mathbf{X}_i)^T(\mathbf{X} - \mathbf{X}_i)$$

$$h_k(\mathbf{X}) \approx h_k(\mathbf{X}_i) + \nabla h_k(\mathbf{X}_i)^T(\mathbf{X} - \mathbf{X}_i)$$

Direct methods

SEQUENTIAL LINEAR PROGRAMMING

3. Formulate the approximating linear programming problem as:

$$\text{Minimize } f(\mathbf{X}_i) + \nabla f_i^T(\mathbf{X} - \mathbf{X}_i)$$

subject to

$$g_j(\mathbf{X}_i) + \nabla g_j(\mathbf{X}_i)^T(\mathbf{X} - \mathbf{X}_i) \le 0, \qquad j = 1,2,\ldots,m$$

$$h_k(\mathbf{X}_i) + \nabla h_k(\mathbf{X}_i)^T(\mathbf{X} - \mathbf{X}_i) = 0, \qquad k = 1,2,\ldots,p$$

Notice that the LP problem in the above equation may sometimes have an unbounded solution. This can be avoided by formulating the first approximating LP problem by considering only the following constraints:

$$l_i \le x_i \le u_i \qquad i=1,2,\ldots,n$$

l_i and u_i represent the lower and upper bounds on x_i, respectively.

Direct methods

SEQUENTIAL LINEAR PROGRAMMING

4. Solve the approximating LP problem to obtain the solution vector $\mathbf{X_{i+1}}$

5. Evaluate the original constraints at $\mathbf{X_{i+1}}$; that is, find

$$g_j(\mathbf{X}_{i+1}), \quad j = 1,2,\ldots,m \quad \text{and} \quad h_k(\mathbf{X}_{i+1}), \quad k = 1,2,\ldots,p$$

If $g_j(\mathbf{X_{i+1}}) \leq \epsilon$ for $j = 1,2,\ldots,m$, and $|h_k(\mathbf{X_{i+1}})| > \epsilon$ for some k, find the most violated constraint, for example, as

$$g_k(\mathbf{X}_{i+1}) = \max_j [g_j(\mathbf{X}_{i+1})]$$

Direct methods

SEQUENTIAL LINEAR PROGRAMMING

Relinearize the constraint $g_k(\mathbf{X}) \leq 0$ about the point $\mathbf{X}_{i+1}$ as

$$g_k(\mathbf{X}) \simeq g_k(\mathbf{X}_{i+1}) + \nabla g_k(\mathbf{X}_{i+1})^T(\mathbf{X} - \mathbf{X}_{i+1}) \leq 0$$

and add this as the $(m+1)$th inequality constraint to the previous LP problem.

6. Set the new iteration number as $i=i+1$, the total number of constraints in the new approximating LP problem as $m+1$ inequalities and p equalities, and go to step 4.

Direct methods

The sequential linear programming method has several advantages:

1. It is an efficient technique for solving complex programming problems with nearly linear objective functions.

2. Each of the approximating problems will be a LP problem and hence can be solved quite efficiently. Moreover, any two consecutive approximating LP problems differ by only one constraint, and hence the dual simplex method can be used to solve the sequence of approximating LP problems much more efficiently.

3. The method can easily be extended to solve integer programming problems. In this case, one integer LP problem has to be solved in each stage.

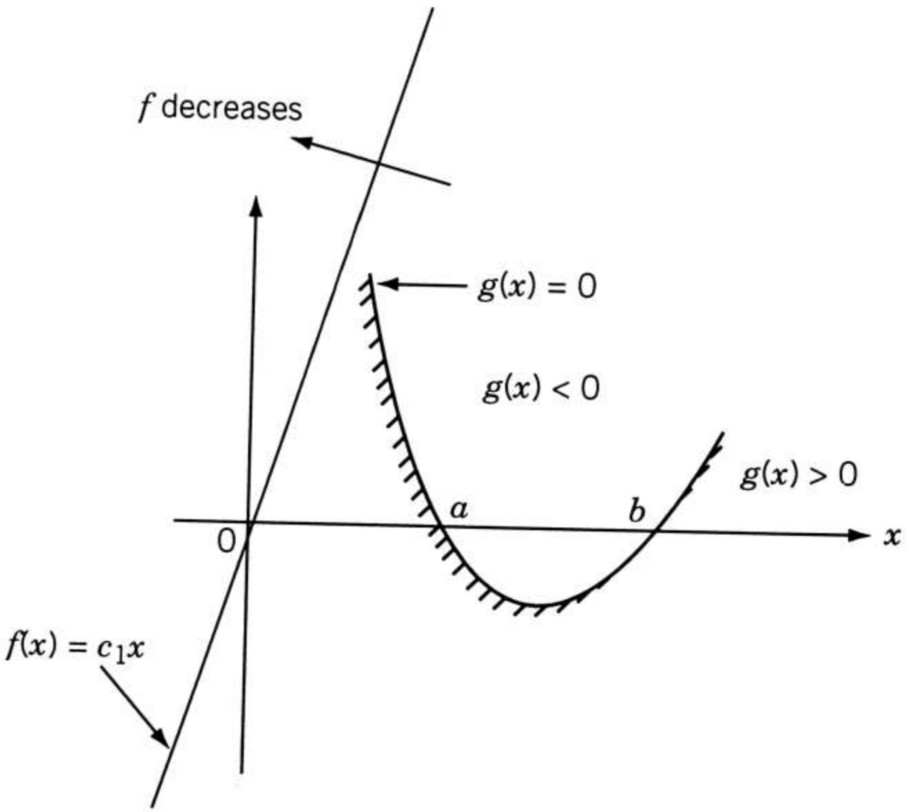
Direct Methods

Geometric Interpretation of the Method:

The SLP method can be illustrated with the help of a one-variable problem:

Minimize $f(x) = c_1 x$

subject to $g(x) \leq 0$



where c_1 is a constant and $g(x)$ is a nonlinear function of x.

Let the feasible region and the contour of the objective function be as shown in the figure:

Direct Methods

Geometric Interpretation of the Method:

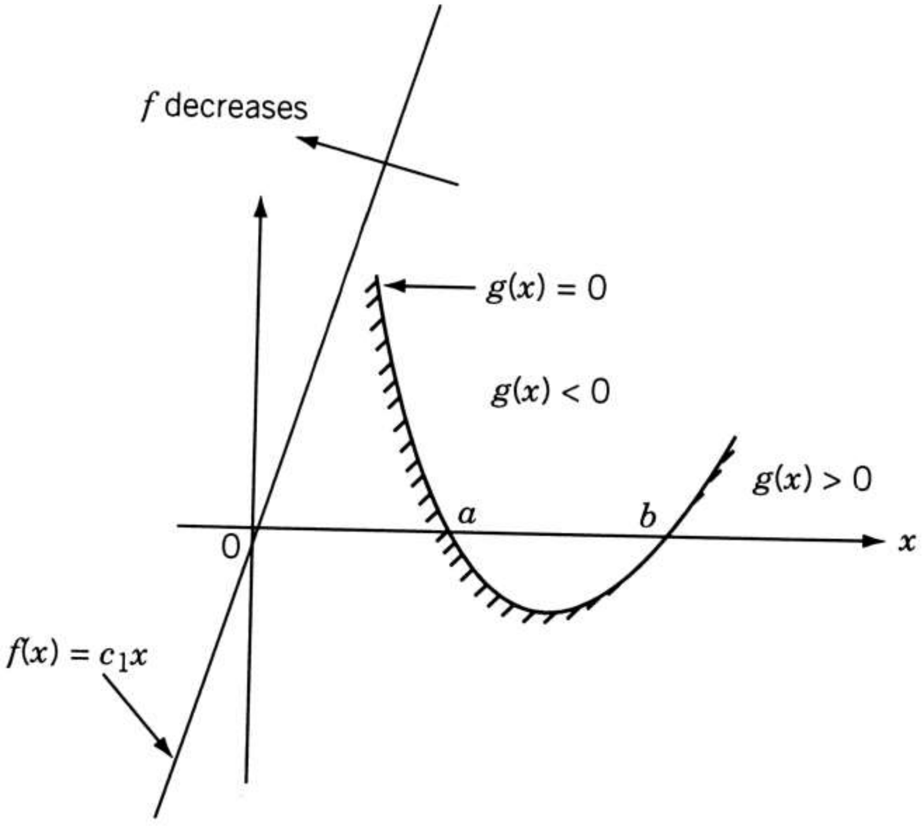
To avoid any possibility of an unbounded solution, let us first take the constraints on x as:

$$c \le x \le d$$

where c and d represent the lower and upper bounds on x. With these constraints, we formulate the LP problem:

$$\text{Minimize } f(x) = c_1 x$$

$$\text{subject to } c \le x \le d$$



Sequential Linear Programming Problem

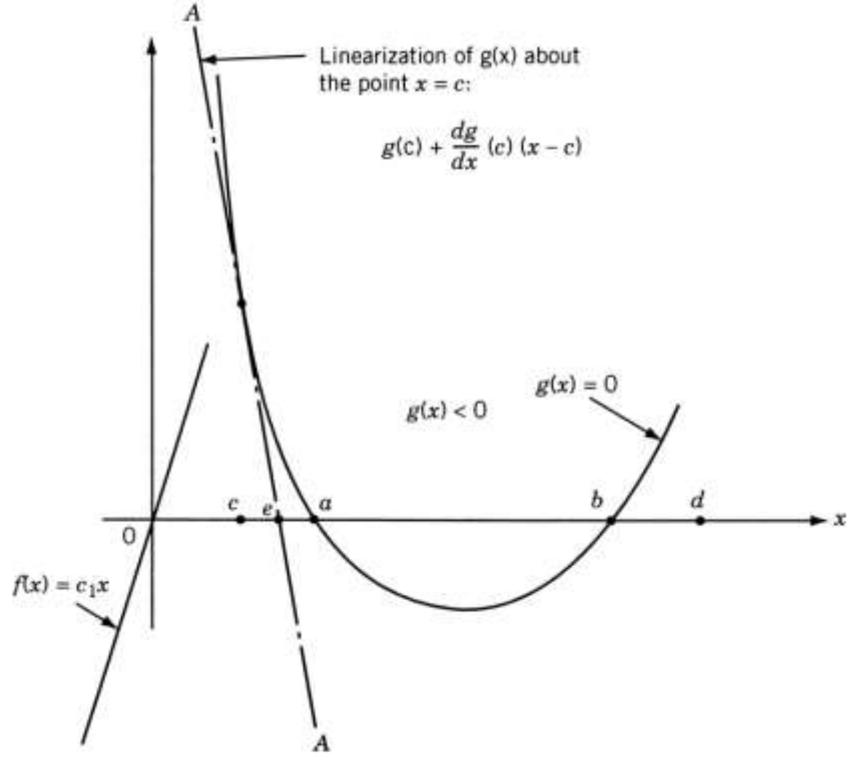
- The optimum solution of this approximating LP problem can be seen to be $x^* = c$. Next, we linearize the constraint $g(x)$ about point c and add it to the previous constraint set. Thus, the new LP problem becomes:

$$\text{Minimize} \quad f(x) = c_1 x$$

$$\text{subject to} \quad c \le x \le d$$

$$g(c) + \frac{dg}{dx}(c)(x - c) \le 0$$

The feasible region of x, according to the above two constraints, is given by $e \le x \le d$ as shown in the figure. The optimum solution of the approximating LP problem given by the above equations can be seen to be $x^* = e$. Next, we linearize the constraint $g(x) \le 0$ about the current solution $x = e^*$ and add it to the previous constraint set to obtain the next approximating LP problem as:



Sequential Linear Programming Problem

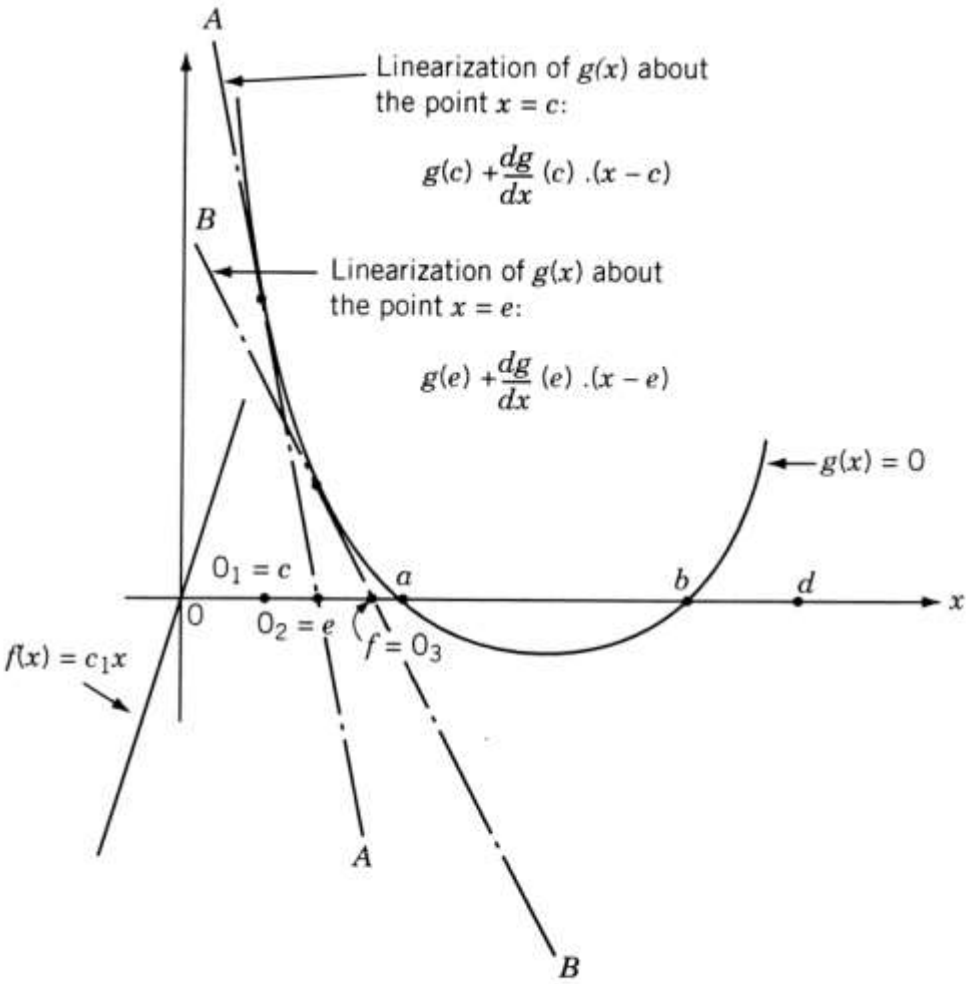
Minimize $f(x) = c_1 x$

subject to $c \le x \le d$

$$g(c) + \frac{dg}{dx}(c)(x-c) \le 0$$

$$g(e) + \frac{dg}{dx}(e)(x-e) \le 0$$

The permissable range of x, according to the above constraints can be seen to be $f \le x \le d$ from the figure. The optimum solution of the LP problem of the above equations can be obtained as $x^* = f$.



Sequential Linear Programming

- We then linearize $g(x) \leq 0$ about the present point $x^*=f$ and add it to the previous constraint set to define a new approximating LP problem. This procedure has to be continued until the optimum solution is found to the desired level of accuracy.

- As can be seen from the figures, the optimum of all the approximating LP problems (e.g., points $c,e,f,\ldots$) lie outside the feasible region and converge toward the optimum point, $x = a$.

- The process is assumed to have converged whenever the solution of an approximating problem satisfies the original constraint within some specified tolerance level as

$$g(x_k^*) \leq \epsilon$$

where ϵ is a small positive number and x_k^* is the optimum solution of the kth approximating LP problem.

Sequential Linear Programming

- It can be seen that the lines (hyperplanes in a general problem) defined by $g(x_k^*) + dg/dx\,(x_k^*)(x - x_k^*)$ cut off a portion of the existing feasible region. Hence this method is called the *cutting plane method.*

Example: Minimize $f(x_1,x_2)=x_1 - x_2$

Subject to $g(x_1,x_2)=3x_1^2 - 2x_1 x_2 + x_2^2 - 1 \le 0$

using the cutting plane method. Take the convergence limit in step 5 as $\epsilon = 0.02.$

Note: This example was originally given by Kelly. Since the constraint boundary represents an ellipse, the problem is a convex programming problem. From graphical representation, the optimum solution of the problem can be identified as $x_1^* = 0$, $x_2^* = 1$, and $f_{min} = -1$

Sequential Linear Programming

Steps 1, 2, 3: Although we can start the solution from any initial point $\mathbf{X}_1$, to avoid the possible unbounded solution, we first take the bounds on x_1 and x_2 as

$$-2 \le x_1 \le 2$$

$$-2 \le x_2 \le 2$$

And solve the following LP problem:

$$\text{Minimize } f = x_1 - x_2$$

subject to

$$-2 \le x_1 \le 2$$
$$-2 \le x_2 \le 2$$

Sequential Linear Programming

The solution of the problem can be obtained as:

$$\mathbf{X} = \begin{bmatrix} \mathbf{-2} \\ \mathbf{2} \end{bmatrix} \quad \text{with} \quad f(\mathbf{X}) = -4$$

Step 4: Since we have solved one LP problem, we can take

$$\mathbf{X}_{i+1} = \mathbf{X}_2 = \begin{Bmatrix} -2 \\ 2 \end{Bmatrix}$$

Step 5: Since $g_1(\mathbf{X}_2) = 23 > \epsilon$, we linearize $g_1(\mathbf{X})$ about point $\mathbf{X}_2$ as

$$g_1(\mathbf{X}) \simeq g_1(\mathbf{X}_2) + \nabla g_1(\mathbf{X}_2)^T(\mathbf{X} - \mathbf{X}_2) \leq 0$$

Sequential Linear Programming

As

$$g_1(\mathbf{X}_2) = 23, \qquad \left.\frac{\partial g_1}{\partial x_1}\right|_{\mathbf{X}_2} = (6x_1 - 2x_2)|_{\mathbf{X}_2} = -16$$

$$\left.\frac{\partial g_1}{\partial x_2}\right|_{\mathbf{X}_2} = (-2x_1 + 2x_2)|_{\mathbf{X}_2} = 8$$

The equation

$$g_1(\mathbf{X}) \simeq g_1(\mathbf{X}_2) + \nabla g_1(\mathbf{X}_2)^T(\mathbf{X} - \mathbf{X}_2) \leq 0$$

becomes

$$g_1(\mathbf{X}) \simeq -16x_1 + 8x_2 - 25 \leq 0$$

Sequential Linear Programming

- By adding this constraint to the previous LP problem, the new LP problem becomes:

$$\text{Minimize } f = x_1 - x_2$$

subject to

$$-2 \le x_1 \le 2$$

$$-2 \le x_2 \le 2$$

$$-16x_1 + 8x_2 - 25 \le 0$$

- *Step 6:* Set the iteration number as $i = 2$ and go to step 4.
- Step 7: Solve the approximating LP problem stated in the above equation and obtain the solution

$$\mathbf{X}_3 = \begin{Bmatrix} -0.5625 \\ 2.0 \end{Bmatrix} \quad \text{with} \quad f_3 = f(\mathbf{X}_3) = -2.5625$$

Sequential Linear Programming

- This procedure is continued until the specified convergence criterion, g_1 $(\mathbf{X_i}) \leq \epsilon$, in step 5 is satisfied. The computational results are summarized in the table.

Iteration Number, i	New Linearized Constraint Considered	Solution of the Approximating LP Problem $\mathbf{X}_{i+1}$	$f(\mathbf{X}_{i+1})$	$g_1(\mathbf{X}_{i+1})$
1	$-2 \leq x_1 \leq 2$ and $-2 \leq x_2 \leq 2$	(−2.0,2.0)	−4.00000	23.00000
2	$-16.0x_1 + 8.0x_2 - 25.0 \leq 0$	(−0.56250,2.00000)	−2.56250	6.19922
3	$-7.375x_1 + 5.125x_2 - 8.19922 \leq 0$	(0.27870,2.00000)	−1.72193	2.11978
4	$-2.33157x_1 + 3.44386x_2 - 4.11958 \leq 0$	(−0.52970,0.83759)	−1.36730	1.43067
5	$-4.85341x_1 + 2.73459x_2 - 3.43067 \leq 0$	(−0.05314,1.16024)	−1.21338	0.47793
6	$-2.63930x_1 + 2.42675x_2 - 2.47792 \leq 0$	(0.42655,1.48490)	−1.05845	0.48419
7	$-0.41071x_1 + 2.11690x_2 - 2.48420 \leq 0$	(0.17058,1.20660)	−1.03603	0.13154
8	$-1.38975x_1 + 2.07205x_2 - 2.13155 \leq 0$	(0.01829,1.04098)	−1.02269	0.04656
9	$-1.97223x_1 + 2.04538x_2 - 2.04657 \leq 0$	(−0.16626,0.84027)	−1.00653	0.06838
10	$-2.67809x_1 + 2.01305x_2 - 2.06838 \leq 0$	(−0.07348,0.92972)	−1.00321	0.01723

Sequential Quadratic Programming

- The sequential quadratic programming is one of the most recently developed and perhaps one of the best methods of optimization.

- The method has a theoretical basis that is related to

1. The solution of a set of nonlinear equations using Newton's method.

2. The derivation of simultaneous nonlinear equations using Kuhn-Tucker conditions to the Lagrangian of the constrained optimization problem.

Sequential Quadratic Programming

- Consider a nonlinear optimization problem with only equality constraints as:

$$\text{Find } \mathbf{X} \text{ which minimizes } f(\mathbf{X})$$

subject to

$$h_k(\mathbf{X}) = 0, \qquad k = 1,2,\ldots,p$$

- The extension to include inequality constraints will be considered at a later stage. The Lagrange function, L **($\mathbf{X}$,λ),** corresponding to the problem of the above equation is given by:

$$L = f(\mathbf{X}) + \sum_{k=1}^{p} \lambda_k h_k(\mathbf{X})$$

where λ_k is the Lagrange multiplier for the k^{th} equality constraint.

Sequential Quadratic Programming

- The Kuhn-Tucker necessary consitions can be stated as:

$$\nabla L = \mathbf{0} \quad \text{or} \quad \nabla f + \sum_{k=1}^{p} \lambda_k \nabla h_k = \mathbf{0} \quad \text{or} \quad \nabla f + [A]^T \boldsymbol{\lambda} = \mathbf{0}$$

$$h_k(\mathbf{X}) = 0, \qquad k = 1,2,\ldots,p$$

where [A] is an n x p matrix whose k^{th} column denotes the gradient of the function h_k. The above equations represent a set of $n+p$ nonlinear equations in $n+p$ unknowns (x_i, $i=1,\ldots,n$ and λ_k, $k=1,\ldots,p$). These nonlinear equations can be solved using Newton's method. For convenience, we rewrite the above equations as:

$$\mathbf{F}(\mathbf{Y}) = \mathbf{0}$$

where

$$\mathbf{F} = \begin{Bmatrix} \nabla L \\ \mathbf{h} \end{Bmatrix}_{(n+p)\times 1}, \qquad \mathbf{Y} = \begin{Bmatrix} \mathbf{X} \\ \boldsymbol{\lambda} \end{Bmatrix}_{(n+p)\times 1}, \qquad \mathbf{0} = \begin{Bmatrix} \mathbf{0} \\ \mathbf{0} \end{Bmatrix}_{(n+p)\times 1}$$

Sequential Quadratic Programming

- According to the Newton's method, the solution of the above equation can be found as:

$$\mathbf{Y}_{j+1} = \mathbf{Y}_j + \Delta\mathbf{Y}_j$$

with

$$[\nabla F]_j^T \Delta\mathbf{Y}_j = -\mathbf{F}(\mathbf{Y}_j)$$

where $\mathbf{Y}_j$ is the solution at the start of the jth equation and $\Delta\mathbf{Y}_j$ is the change in $\mathbf{Y}_j$ necessary to generate the improved solution, $\mathbf{Y}_{j+1}$, and $[\nabla F]_j = [\nabla F(\mathbf{Y}_j)]_j$ is the $(n+p)$ x $(n+p)$ Jacobian matrix of the nonlinear equations whose ith column denotes the gradient of the function $F_i(\mathbf{Y})$ with respect to the vector **Y.**

Sequential Quadratic Programming

- By substituting

$$\mathbf{F}(\mathbf{Y}) = \mathbf{0}$$

where

$$\mathbf{F} = \begin{Bmatrix} \nabla L \\ \mathbf{h} \end{Bmatrix}_{(n+p)\times 1}, \quad \mathbf{Y} = \begin{Bmatrix} \mathbf{X} \\ \boldsymbol{\lambda} \end{Bmatrix}_{(n+p)\times 1}, \quad \mathbf{0} = \begin{Bmatrix} \mathbf{0} \\ \mathbf{0} \end{Bmatrix}_{(n+p)\times 1}$$

into

$$[\nabla F]_j^T \Delta \mathbf{Y}_j = -\mathbf{F}(\mathbf{Y}_j)$$

we obtain:

$$\begin{bmatrix} [\nabla^2 L] & [H] \\ [H]^T & [0] \end{bmatrix}_j \begin{Bmatrix} \Delta \mathbf{X} \\ \Delta \boldsymbol{\lambda} \end{Bmatrix}_j = -\begin{Bmatrix} \nabla L \\ \mathbf{h} \end{Bmatrix}_j$$

$$\Delta \mathbf{X}_j = \mathbf{X}_{j+1} - \mathbf{X}_j$$

$$\Delta \boldsymbol{\lambda}_j = \boldsymbol{\lambda}_{j+1} - \boldsymbol{\lambda}_j$$

Sequential Quadratic Programming Method

where

$$[\nabla^2 L]_{n \times n}$$

denotes the Hessian matrix of the Lagrange function. The first set of equations in

$$\begin{bmatrix} [\nabla^2 L] & [H] \\ [H]^T & [0] \end{bmatrix}_j \begin{Bmatrix} \Delta \mathbf{X} \\ \Delta \boldsymbol{\lambda} \end{Bmatrix}_j = -\begin{Bmatrix} \nabla L \\ \mathbf{h} \end{Bmatrix}_j$$

can be written separately as:

$$[\nabla^2 L]_j \Delta \mathbf{X}_j + [H]_j \Delta \boldsymbol{\lambda}_j = -\nabla L_j$$

Sequential Quadratic Programming Method

- The equation

$$[\nabla^2 L]_j \Delta\mathbf{X}_j + [H]_j \Delta\boldsymbol{\lambda}_j = -\nabla L_j$$

and the second set of equations in the equation

$$\begin{bmatrix} [\nabla^2 L] & [H] \\ [H]^T & [0] \end{bmatrix}_j \begin{Bmatrix} \Delta\mathbf{X} \\ \Delta\boldsymbol{\lambda} \end{Bmatrix}_j = -\begin{Bmatrix} \nabla L \\ \mathbf{h} \end{Bmatrix}_j$$

can now be combined as:

$$\begin{bmatrix} [\nabla^2 L] & [H] \\ [H]^T & [0] \end{bmatrix}_j \begin{Bmatrix} \Delta\mathbf{X}_j \\ \boldsymbol{\lambda}_{j+1} \end{Bmatrix} = -\begin{Bmatrix} \nabla f_j \\ \mathbf{h}_j \end{Bmatrix}$$

The above equation can be solved to find the change in the design vector $\Delta\mathbf{X}_\mathbf{j}$ and the new values of the Lagrange multipliers, λ_{j+1}.The iterative process indicated by the above equation can be continued until convergence is achieved.

Sequential Quadratic Programming Method

- Now consider the following quadratic programming problem:
- Find $\Delta\mathbf{X}$ that minimizes the quadratic objective function

$$Q = \nabla f^T \Delta\mathbf{X} + \frac{1}{2}\Delta\mathbf{X}^T[\nabla^2 L]\Delta\mathbf{X}$$

subject to the linear equality constraints

$$h_k + \nabla h_k^T \Delta\mathbf{X} = 0, \qquad k = 1,2,\ldots,p \quad \text{or} \quad \mathbf{h} + [H]^T \Delta\mathbf{X} = \mathbf{0}$$

- The Lagrange function L corresponding to the above problem is given by:

$$\tilde{L} = \nabla f^T \Delta\mathbf{X} + \frac{1}{2}\Delta\mathbf{X}^T[\nabla^2 L]\Delta\mathbf{X} + \sum_{k=1}^{p} \lambda_k (h_k + \nabla h_k^T \Delta\mathbf{X})$$

where λ_k is the Lagrange multiplier associated with the k^{th} equality constraint.

Sequential Quadratic Programming Method

- The Kuhn-Tucker necessary conditions can be stated as:

$$\nabla f + [\nabla^2 L]\Delta\mathbf{X} + [H]\boldsymbol{\lambda} = \mathbf{0}$$

$$h_k + \nabla h_k^T \Delta\mathbf{X} = 0, \qquad k = 1,2,\ldots,p$$

- The above equations can be identified to be same as

$$\begin{bmatrix} [\nabla^2 L] & [H] \\ [H]^T & [0] \end{bmatrix}_j \begin{Bmatrix} \Delta\mathbf{X}_j \\ \boldsymbol{\lambda}_{j+1} \end{Bmatrix} = -\begin{Bmatrix} \nabla f_j \\ \mathbf{h}_j \end{Bmatrix}$$

in matrix form.

Sequential Quadratic Programming Method

- This shows that the original problem of the equation

$$\text{Find } \mathbf{X} \text{ which minimizes } f(\mathbf{X})$$

subject to

$$h_k(\mathbf{X}) = 0, \qquad k = 1,2,\ldots,p$$

can be solved iteratively by solving the quadratic programming problem defined by the equation

$$Q = \nabla f^T \Delta\mathbf{X} + \tfrac{1}{2}\Delta\mathbf{X}^T[\nabla^2 L]\Delta\mathbf{X}$$

- In fact, when inequality constraints are added to the original problem, the quadratic programming problem of the above equation becomes:

Sequential Quadratic Programming Method

Find $\mathbf{X}$ which minimizes $Q = \nabla f^T \Delta\mathbf{X} + \frac{1}{2}\Delta\mathbf{X}^T[\nabla^2 L]\Delta\mathbf{X}$

subject to

$$g_j + \nabla g_j^T \Delta\mathbf{X} \le 0, \quad j = 1,2,\ldots,m$$

$$h_k + \nabla h_k^T \Delta\mathbf{X} = 0, \quad k = 1,2,\ldots,p$$

with the Lagrange function given by:

$$\tilde{L} = f(\mathbf{X}) + \sum_{j=1}^{m} \lambda_j g_j(\mathbf{X}) + \sum_{k=1}^{p} \lambda_{m+k} h_k(\mathbf{X})$$

- Since the minimum of the Augmented Lagrange function is involved, the sequential quadratic programming method is also known as the projected Lagrangian method.

Sequential Quadratic Programming Method

Solution Procedure

As in the case of Newton's method of unconstrained minimization, the solution vector $\Delta\mathbf{X}$ in the equation

$$\text{Find } \mathbf{X} \text{ which minimizes } Q = \nabla f^T \Delta\mathbf{X} + \tfrac{1}{2}\Delta\mathbf{X}^T[\nabla^2 L]\,\Delta\mathbf{X}$$

subject to

$$g_j + \nabla g_j^T \Delta\mathbf{X} \le 0, \qquad j = 1,2,\ldots,m$$
$$h_k + \nabla h_k^T \Delta\mathbf{X} = 0, \qquad k = 1,2,\ldots,p$$

is treated as the search direction **S**, and the quadratic programming subproblem (in terms of the design vector **S**) is restated as:

$$\text{Find } \mathbf{S} \text{ which minimizes } Q(\mathbf{S}) = \nabla f(\mathbf{X})^T\mathbf{S} + \tfrac{1}{2}\mathbf{S}^T[H]\mathbf{S}$$

subject to

$$\beta_j g_j(\mathbf{X}) + \nabla g_j(\mathbf{X})^T\mathbf{S} \le 0, \qquad j = 1,2,\ldots,m$$
$$\bar{\beta} h_k(\mathbf{X}) + \nabla h_k(\mathbf{X})^T\mathbf{S} = 0, \qquad k = 1,2,\ldots,p$$

Sequential Quadratic Programming Method

where [H] is a positive definite matrix that is taken initially as the identity matrix and is updated in subsequent iterations so as to converge to the Hessian matrix of the Lagrange function of the equation:

$$\tilde{L} = f(\mathbf{X}) + \sum_{j=1}^{m} \lambda_j g_j(\mathbf{X}) + \sum_{k=1}^{p} \lambda_{m+k} h_k(\mathbf{X})$$

and

$$\beta_j \text{ and } \overline{\beta}$$

are constants used to ensure that the linearized constraints do not cut off the feasible space completely. Typical values of these constants are given by:

$$\overline{\beta} \approx 0.9; \quad \beta_j = \begin{cases} 1 & \text{if } g_j(\mathbf{X}) \le 0 \\ \overline{\beta} & \text{if } g_j(\mathbf{X}) \ge 0 \end{cases}$$

Sequential Quadratic Programming Method

- The subproblem of the equation

$$\text{Find } \mathbf{S} \text{ which minimizes } Q(\mathbf{S}) = \nabla f(\mathbf{X})^T\mathbf{S} + \tfrac{1}{2}\mathbf{S}^T[H]\mathbf{S}$$

subject to

$$\beta_j g_j(\mathbf{X}) + \nabla g_j(\mathbf{X})^T\mathbf{S} \le 0, \qquad j = 1,2,\ldots,m$$

$$\overline{\beta} h_k(\mathbf{X}) + \nabla h_k(\mathbf{X})^T\mathbf{S} = 0, \qquad k = 1,2,\ldots,p$$

is a quadratic programming problem and hence methods for minimizing quadratic functions can be used for their solution. Alternatively, the problem can be solved by any of the methods described in this lecture since the gradients of the function involved can be evaluated easily. Since the Lagrange multipliers associated with the solution of the above problem, are needed, they can be evaluated:

Sequential Quadratic Programming Method

$$\underset{n\times p}{\mathbf{G}}\ \underset{p\times 1}{\boldsymbol{\lambda}} = \underset{n\times 1}{\mathbf{F}}$$

where

$$\mathbf{G} = \begin{bmatrix} \frac{\partial g_{j1}}{\partial x_1} & \frac{\partial g_{j2}}{\partial x_1} & \cdots & \frac{\partial g_{jp}}{\partial x_1} \\ \frac{\partial g_{j1}}{\partial x_2} & \frac{\partial g_{j2}}{\partial x_2} & \cdots & \frac{\partial g_{jp}}{\partial x_2} \\ \vdots & & & \\ \frac{\partial g_{j1}}{\partial x_n} & \frac{\partial g_{j2}}{\partial x_n} & \cdots & \frac{\partial g_{jp}}{\partial x_n} \end{bmatrix}_{\mathbf{X}}$$

$$\boldsymbol{\lambda} = \begin{Bmatrix} \lambda_{j1} \\ \lambda_{j2} \\ \vdots \\ \lambda_{jp} \end{Bmatrix} \quad \text{and} \quad \mathbf{F} = \begin{Bmatrix} -\frac{\partial f}{\partial x_1} \\ -\frac{\partial f}{\partial x_2} \\ \vdots \\ -\frac{\partial f}{\partial x_n} \end{Bmatrix}_{\mathbf{X}}$$

$$\boldsymbol{\lambda} = (\mathbf{G}^T\mathbf{G})^{-1}\mathbf{G}^T\mathbf{F}$$

Sequential Quadratic Programming Method

- Once the search direction, **S**, is found by solving the problem in the equation

$$\text{Find } \mathbf{S} \text{ which minimizes } Q(\mathbf{S}) = \nabla f(\mathbf{X})^T \mathbf{S} + \tfrac{1}{2}\mathbf{S}^T[H]\mathbf{S}$$

subject to

$$\beta_j g_j(\mathbf{X}) + \nabla g_j(\mathbf{X})^T \mathbf{S} \le 0, \quad j = 1,2,\ldots,m$$
$$\overline{\beta} h_k(\mathbf{X}) + \nabla h_k(\mathbf{X})^T \mathbf{S} = 0, \quad k = 1,2,\ldots,p$$

the design vector is updated as:

$$\mathbf{X}_{j+1} = \mathbf{X}_j + \alpha^* \mathbf{S}$$

where α^* is the optimal step length along the direction **S** found by minimizing the function (using an exterior penalty function approach):

$$\phi = f(\mathbf{X}) + \sum_{j=1}^{m} \lambda_j (\max[0, g_j(\mathbf{X})]) + \sum_{k=1}^{p} \lambda_{m+k} |h_k(\mathbf{X})|$$

with

$$\lambda_j = \begin{cases} |\lambda_j|, & j = 1,2,\ldots,m+p \text{ in first iteration} \\ \max\{|\lambda_j|, \tfrac{1}{2}(\tilde{\lambda}_j, |\lambda_j|)\} & \text{in subsequent iterations} \end{cases}$$

Sequential Quadratic Programming Method

and $\tilde{\lambda}_j = \lambda_j$ of the previous iteration.

- The one-dimensional step length α^* can be found by any of the methods discussed before for one-dimensional minimization.
- Once $\mathbf{X}_{j+1}$ is found from the equation $\mathbf{X}_{j+1} = \mathbf{X}_j + \alpha^* \mathbf{S}$, for the next iteration of the Hessian matrix [H] is updated to improve the quadratic approximation in the equation

$$\text{Find } \mathbf{S} \text{ which minimizes } Q(\mathbf{S}) = \nabla f(\mathbf{X})^T \mathbf{S} + \tfrac{1}{2} \mathbf{S}^T [H] \mathbf{S}$$

subject to

$$\beta_j g_j(\mathbf{X}) + \nabla g_j(\mathbf{X})^T \mathbf{S} \le 0, \qquad j = 1,2,\ldots,m$$

$$\overline{\beta} h_k(\mathbf{X}) + \nabla h_k(\mathbf{X})^T \mathbf{S} = 0, \qquad k = 1,2,\ldots,p$$

Sequential Quadratic Programming Method

- Usually, a modified BFGS formula, given below is used for this purpose:

$$[H_{i+1}] = [H_i] - \frac{[H_i]\mathbf{P}_i\mathbf{P}_i^T[H_i]}{\mathbf{P}_i^T[H_i]\mathbf{P}_i} + \frac{\boldsymbol{\gamma}\boldsymbol{\gamma}^T}{\mathbf{P}_i^T\mathbf{P}_i}$$

$$\mathbf{P}_i = \mathbf{X}_{i+1} - \mathbf{X}_i$$

$$\boldsymbol{\gamma} = \theta\mathbf{Q}_i + (1 - \theta)[H_i]\mathbf{P}_i$$

$$\mathbf{Q}_i = \nabla_x\tilde{L}(\mathbf{X}_{i+1},\boldsymbol{\lambda}_{i+1}) - \nabla_x\tilde{L}(\mathbf{X}_i,\boldsymbol{\lambda}_i)$$

$$\theta = \begin{cases} 1.0 & \text{if} \quad \mathbf{P}_i^T\mathbf{Q}_i \geq 0.2\mathbf{P}_i^T[H_i]\mathbf{P}_i \\ \dfrac{0.8\mathbf{P}_i^T[H_i]\mathbf{P}_i}{\mathbf{P}_i^T[H_i]\mathbf{P}_i - \mathbf{P}_i^T\mathbf{Q}_i} & \text{if} \quad \mathbf{P}_i^T\mathbf{Q}_i < 0.2\mathbf{P}_i^T[H_i]\mathbf{P}_i \end{cases}$$

where $\tilde{L}$ is given by

$$\tilde{L} = f(\mathbf{X}) + \sum_{j=1}^{m} \lambda_j g_j(\mathbf{X}) + \sum_{k=1}^{p} \lambda_{m+k} h_k(\mathbf{X})$$

and the constants 0.2 and 0.8 in the above equation can be changed, based on numerical experience.

Sequential Quadratic Programming Method

Example 1: Find the solution of the problem

$$\text{Minimize } f(\mathbf{X}) = 0.1x_1 + 0.05773x_2$$

subject to

$$g_1(\mathbf{X}) = \frac{0.6}{x_1} + \frac{0.3464}{x_2} - 0.1 \leq 0$$

$$g_2(\mathbf{X}) = 6 - x_1 \leq 0$$

$$g_3(\mathbf{X}) = 7 - x_2 \leq 0$$

using the sequential quadratic programming technique

Sequential Quadratic Programming Method - Example 1

Solution: Let the starting point be:

$$\mathbf{X}_1 = (11.8765 \quad 7.0)^T$$

With $g_1(\mathbf{X}_1) = g_3(\mathbf{X}_1) = 0$, $g_2(\mathbf{X}_1) = -5.8765$, and $f(\mathbf{X}_1) = 1.5917$. The gradients of the objective and constraint functions at $\mathbf{X}_1$ are given by:

$$\nabla f(\mathbf{X}_1) = \begin{Bmatrix} 0.1 \\ 0.05773 \end{Bmatrix}, \qquad \nabla g_1(\mathbf{X}_1) = \begin{Bmatrix} -\dfrac{0.6}{x_1^2} \\ -\dfrac{0.3464}{x_2^2} \end{Bmatrix}_{\mathbf{X}_1} = \begin{Bmatrix} -0.004254 \\ -0.007069 \end{Bmatrix}$$

$$\nabla g_2(\mathbf{X}_1) = \begin{Bmatrix} -1 \\ 0 \end{Bmatrix}, \qquad \nabla g_3(\mathbf{X}_1) = \begin{Bmatrix} 0 \\ -1 \end{Bmatrix}$$

Sequential Quadratic Programming Method - Example 1

Solution:

We assume the matrix $[H_1]$ to be the identity matrix and hence the objective function of the equation

$$\text{Find } \mathbf{S} \text{ which minimizes } Q(\mathbf{S}) = \nabla f(\mathbf{X})^T\mathbf{S} + \tfrac{1}{2}\mathbf{S}^T[H]\mathbf{S}$$

subject to

$$\beta_j g_j(\mathbf{X}) + \nabla g_j(\mathbf{X})^T\mathbf{S} \le 0, \qquad j = 1,2,\ldots,m$$

$$\overline{\beta} h_k(\mathbf{X}) + \nabla h_k(\mathbf{X})^T\mathbf{S} = 0, \qquad k = 1,2,\ldots,p$$

becomes

$$Q(\mathbf{S}) = 0.1s_1 + 0.05773s_2 + 0.5s_1^2 + 0.5s_2^2$$

Sequential Quadratic Programming Method - Example 1

Solution:

Equation

$$\bar{\beta} \approx 0.9; \qquad \beta_j = \begin{cases} 1 & \text{if} \quad g_j(\mathbf{X}) \leq 0 \\ \bar{\beta} & \text{if} \quad g_j(\mathbf{X}) \geq 0 \end{cases}$$

gives $\beta_1=\beta_3=0$ since $g_1= g_3=0$ and $\beta_2=1.0$ since $g_2<0$, and hence the constraints of the equation

$$\text{Find } \mathbf{S} \text{ which minimizes } Q(\mathbf{S}) = \nabla f(\mathbf{X})^T\mathbf{S} + \tfrac{1}{2}\mathbf{S}^T[H]\mathbf{S}$$

subject to

$$\beta_j g_j(\mathbf{X}) + \nabla g_j(\mathbf{X})^T\mathbf{S} \leq 0, \qquad j = 1,2,\ldots,m$$

$$\bar{\beta} h_k(\mathbf{X}) + \nabla h_k(\mathbf{X})^T\mathbf{S} = 0, \qquad k = 1,2,\ldots,p$$

Sequential Quadratic Programming Method - Example 1

Solution:

can be expressed as:

$$\tilde{g}_1 = -0.004254s_1 - 0.007069s_2 \leq 0$$

$$\tilde{g}_2 = -5.8765 - s_1 \leq 0$$

$$\tilde{g}_3 = -s_2 \leq 0$$

We solve this quadratic programming problem directly with the use of the Kuhn-Tucker conditions which are given by:

$$\frac{\partial Q}{\partial s_1} + \sum_{j=1}^{3} \lambda_j \frac{\partial \tilde{g}_j}{\partial s_1} = 0$$

$$\frac{\partial Q}{\partial s_2} + \sum_{j=1}^{3} \lambda_j \frac{\partial \tilde{g}_j}{\partial s_2} = 0$$

$$\lambda_j \tilde{g}_j = 0, \quad j = 1,2,3$$

$$\tilde{g}_j \leq 0, \quad j = 1,2,3$$

$$\lambda_j \geq 0, \quad j = 1,2,3$$

Sequential Quadratic Programming Method - Example 1

- The equations

$$\frac{\partial Q}{\partial s_1} + \sum_{j=1}^{3} \lambda_j \frac{\partial \tilde{g}_j}{\partial s_1} = 0$$

$$\frac{\partial Q}{\partial s_2} + \sum_{j=1}^{3} \lambda_j \frac{\partial \tilde{g}_j}{\partial s_2} = 0$$

can be expressed in this case as:

$$0.1 + s_1 - 0.004254\lambda_1 - \lambda_2 = 0$$

$$0.05773 + s_2 - 0.007069\lambda_1 - \lambda_3 = 0$$

- By considering all possibilities of active constraints, we find that the optimum solution of the quadratic programming problem is given by

$$s_1^* = -0.04791, \quad s_2^* = 0.02883, \quad \lambda_1^* = 12.2450, \quad \lambda_2^* = 0, \quad \lambda_3^* = 0$$

Sequential Quadratic Programming Method - Example 1

- The new design vector **X** can be expressed as:

$$\mathbf{X} = \mathbf{X}_1 + \alpha\mathbf{S} = \begin{Bmatrix} 11.8765 - 0.04791\alpha \\ 7.0 + 0.02883\alpha \end{Bmatrix}$$

where α can be found by minimizing the function ϕ in equation

$$\phi = f(\mathbf{X}) + \sum_{j=1}^{m} \lambda_j (\max[0, g_j(\mathbf{X})]) + \sum_{k=1}^{p} \lambda_{m+k} |h_k(\mathbf{X})|$$

with

$$\lambda_j = \begin{cases} |\lambda_j|, \quad j = 1,2,\ldots,m+p \text{ in first iteration} \\ \max\ \{|\lambda_j|, \frac{1}{2}(\tilde{\lambda}_j, |\lambda_j|)\} \text{ in subsequent iterations} \end{cases}$$

Sequential Quadratic Programming Method - Example 1

$$\phi = 0.1(11.8765 - 0.04791\alpha) + 0.05773(7.0 + 0.02883\alpha) + 12.2450\left(\frac{0.6}{11.8765 - 0.04791\alpha} + \frac{0.3464}{7.0 + 0.02883\alpha} - 0.1\right)$$

- By using the quadratic interpolation technique (unrestricted search method can also be used for simplicity), we find that ϕ attains its minimum value of 1.48 at $\alpha^*=64.93$, which corresponds to the new design vector

$$\mathbf{X}_2 = \begin{Bmatrix} 8.7657 \\ 8.8719 \end{Bmatrix}$$

with $f\,(\mathbf{X}_2)$=1.38874 and $g_1\,(\mathbf{X}_2)$=0.0074932 (violated slightly)

Sequential Quadratic Programming Method - Example 1

- Next we update the matrix [H] using

$$[H_{i+1}] = [H_i] - \frac{[H_i]\mathbf{P}_i\mathbf{P}_i^T[H_i]}{\mathbf{P}_i^T[H_i]\mathbf{P}_i} + \frac{\gamma\gamma^T}{\mathbf{P}_i^T\mathbf{P}_i}$$

with

$$\tilde{L} = 0.1x_1 + 0.05773x_2 + 12.2450\left(\frac{0.6}{x_1} + \frac{0.3464}{x_2} - 0.1\right)$$

$$\nabla_x\tilde{L} = \begin{Bmatrix} \dfrac{\partial\tilde{L}}{\partial x_1} \\ \dfrac{\partial\tilde{L}}{\partial x_2} \end{Bmatrix} \quad \text{with} \quad \frac{\partial\tilde{L}}{\partial x_1} = 0.1 - \frac{7.3470}{x_1^2}$$

$$\text{and} \quad \frac{\partial\tilde{L}}{\partial x_2} = 0.05773 - \frac{4.2417}{x_2^2}$$

$$\mathbf{P}_1 = \mathbf{X}_2 - \mathbf{X}_1 = \begin{Bmatrix} -3.1108 \\ 1.8719 \end{Bmatrix}$$

$$\mathbf{Q}_1 = \nabla_x\tilde{L}(\mathbf{X}_2) - \nabla_x\tilde{L}(\mathbf{X}_1) = \begin{Bmatrix} 0.00438 \\ 0.00384 \end{Bmatrix} - \begin{Bmatrix} 0.04791 \\ -0.02883 \end{Bmatrix} = \begin{Bmatrix} -0.04353 \\ 0.03267 \end{Bmatrix}$$

$$\mathbf{P}_1^T[H_1]\mathbf{P}_1 = 13.1811, \quad \mathbf{P}_1^T\mathbf{Q}_1 = 0.19656$$

Sequential Quadratic Programming Method - Example 1

This indicates that $\mathbf{P}_1^T\mathbf{Q}_1 < 0.2\mathbf{P}_1^T[H_1]\mathbf{P}_1$, and hence θ is computed using Eq. (7.147) as

$$\theta = \frac{(0.8)(13.1811)}{13.1811 - 0.19656} = 0.81211$$

$$\boldsymbol{\gamma} = \theta\mathbf{Q}_1 + (1 - \theta)[H_1]\mathbf{P}_1 = \begin{Bmatrix} 0.54914 \\ -0.32518 \end{Bmatrix}$$

Hence

$$[H_2] = \begin{bmatrix} 0.2887 & 0.4283 \\ 0.4283 & 0.7422 \end{bmatrix}$$

Sequential Quadratic Programming Method - Example 1

- We can now start another iteration by defining a new quadratic programming problem using

$$\text{Find } \mathbf{S} \text{ which minimizes } Q(\mathbf{S}) = \nabla f(\mathbf{X})^T\mathbf{S} + \tfrac{1}{2}\mathbf{S}^T[H]\mathbf{S}$$

subject to

$$\beta_j g_j(\mathbf{X}) + \nabla g_j(\mathbf{X})^T\mathbf{S} \le 0, \qquad j = 1,2,\ldots,m$$

$$\overline{\beta} h_k(\mathbf{X}) + \nabla h_k(\mathbf{X})^T\mathbf{S} = 0, \qquad k = 1,2,\ldots,p$$

and continue the procedure until the optimum solution is found. Note that the objective function reduced from a value of 1.5917 to 1.38874 in one iteration when **X** changed from $\mathbf{X}_1$ to $\mathbf{X}_2$.

Introduction to Engineering Optimization (ME6806)

Dr. Yogesh Kumar
Assistant Professor
Mechanical Engineering Department
National Institute of Technology Patna
Bihar - 800 005, India
yogesh.me@nitp.ac.in

Module 5

Constrained Optimization Algorithms: Transformation Methods

TRANSFORMATION TECHNIQUES

- If the constraints g_j **(X)** are explicit functions of the variables x_i and have certain simple forms, it may be possible to make a transformation of the independent variables such that the constraints are satisfied automatically.

- Thus, it may be possible to convert a constrained optimization problem into an unconstrained one by making change of variables. Some typical transformations are indicated below:

If lower and upper bounds are satisfied as

$$l_i \le x_i \le u_i$$

these can be satisfied by transforming the variable x_i as:

$$x_i = l_i + (u_i - l_i) \sin^2 y_i$$

where y_i is the new variable, which can take any value.

TRANSFORMATION TECHNIQUES

2. If a variable x_i is restricted to lie in the interval (0,1), we can use the transformation:

$$x_i = \sin^2 y_i, \qquad x_i = \cos^2 y_i$$

$$x_i = \frac{e^{y_i}}{e^{y_i} + e^{-y_i}} \quad \text{or} \quad x_i = \frac{y_i^2}{1 + y_i^2}$$

3. If the variable x_i is constrained to take only positive values, the transformation can be:

$$x_i = \text{abs}(y_i), \quad x_i = y_i^2 \quad \text{or} \quad x_i = e^{y_i}$$

TRANSFORMATION TECHNIQUES

4. If the variable is restricted to take values lying only in between -1 and 1, the transformation can be

$$x_i = \sin y_i, \quad x_i = \cos y_i, \quad \text{or} \quad x_i = \frac{2y_i}{1 + y_i^2}$$

Note the following aspects of the transformation techniques:

1. The constraints $g_j(\mathbf{X})$ have to be very simple functions of x_i.
2. For certain constraints, it may not be possible to find the necessary transformation.
3. If it is not possible to eliminate all the constraints by making change of variables, it may be better not to use the transformation at all. The partial transformation may sometimes produce a distorted objective function which might be more difficult to minimize than the original function.

TRANSFORMATION TECHNIQUES

Example: Find the dimensions of a rectangular prism type box that has the largest volume when the sum of its length, width and height is limited to a maximum value of 60 in, and its length is restricted to a maximum value of 36 in.

Solution: Let x_1, x_2, and x_3 denote the length, width, and height of the box, respectively. The problem can be stated as follows:

$$\text{Maximize } f(x_1, x_2, x_3) = x_1 x_2 x_3 \qquad (E_1)$$

subject to

$$x_1 + x_2 + x_3 \leq 60 \qquad (E_2)$$

$$x_1 \leq 36 \qquad (E_3)$$

$$x_i \geq 0, \quad i = 1,2,3 \qquad (E_4)$$

TRANSFORMATION TECHNIQUES - Example

Solution: By introducing the new variables as:

$$y_1 = x_1, \quad y_2 = x_2, \quad y_3 = x_1 + x_2 + x_3 \qquad (E_5)$$

or

$$x_1 = y_1, \quad x_2 = y_2, \quad x_3 = y_3 - y_1 - y_2 \qquad (E_6)$$

The constraints of the equations (E2) to (E4) can be restated as:

$$0 \le y_1 \le 36, \quad 0 \le y_2 \le 60, \quad 0 \le y_3 \le 60$$

where the upper bound, for example, on y_2 is obtained by setting $x_1=x_3=0$ in the equation E_2. The constraints of the equation E7 will be satisfied automatically if we define new variables z_i, $i=1,2,3$, as

$$y_1 = 36 \sin^2 z_1, \quad y_2 = 60 \sin^2 z_2, \quad y_3 = 60 \sin^2 z_3$$

TRANSFORMATION TECHNIQUES - Example

- Thus the problem can be stated as an unconstrained problem as follows:

$$\text{Maximize } f(z_1, z_2, z_3)$$

$$= y_1 y_2 (y_3 - y_1 - y_2) \tag{E$_9$}$$

$$= 2160 \sin^2 z_1 \sin^2 z_2 (60 \sin^2 z_3 - 36 \sin^2 z_1 - 60 \sin^2 z_2)$$

- The necessary conditions of optimality yield the relations:

$$\frac{\partial f}{\partial z_1} = 259{,}200 \sin z_1 \cos z_1 \sin^2 z_2 \left(\sin^2 z_3 - \tfrac{6}{5} \sin^2 z_1 - \sin^2 z_2\right) = 0 \tag{E$_{10}$}$$

$$\frac{\partial f}{\partial z_2} = 518{,}400 \sin^2 z_1 \sin z_2 \cos z_2 \left(\tfrac{1}{2} \sin^2 z_3 - \tfrac{3}{10} \sin^2 z_1 - \sin^2 z_2\right) = 0 \tag{E$_{11}$}$$

$$\frac{\partial f}{\partial z_3} = 259{,}200 \sin^2 z_1 \sin^2 z_2 \sin z_3 \cos z_3 = 0 \tag{E$_{12}$}$$

TRANSFORMATION TECHNIQUES - Example

- Equation E_{12} gives the nontrivial solution as cos z_3=0 or $\sin^2 z_3$=1. Hence the equations E_{10} and E_{11} yield $\sin^2 z_1$=5/9 and $\sin^2 z_2$=1/3.

- Thus, the optimal solution is given by $x_1^*=20$ *in.*, $x_2^*=20$ *in.*, $x_3^*=20$ *in.*, and the maximum volume =8000 in^3.

Thanks