CHAPTER 6. DESIGN OPTIMIZATION

6.1 General Model of Design under Uncertainty

The design under uncertainty can generally be defined as:

$$\begin{aligned} &\text{Minimize} && \text{Cost}(\mathbf{d}) \quad \text{or} \quad \text{Risk}(\mathbf{d}) \\ &\text{subject to} && P\left\{G_i\left\{\mathbf{X};\mathbf{d}(\mathbf{X})\right\}>0\right\} \le P_{f_i},\ i=1,\cdots,nc \\ & && \mathbf{d}_\text{L} \le \mathbf{d} \le \mathbf{d}_\text{U},\quad \mathbf{d}\in R^{nd} \text{ and } \mathbf{X}\in R^{nr} \end{aligned} \tag{67}$$

where nc is the number of probabilistic constraints; nd is the number of design parameters; nr is the number of random variables; $\mathbf{d}=[d_i]^T=\mu(\mathbf{X})$ is the design vector; $\mathbf{X}=[X_i]^T$ is the random vector; and the probabilistic constraints are described by the performance function $G_i\left\{\mathbf{X};\mathbf{d}(\mathbf{X})\right\}$, their probabilistic models, and the probability of failure. The probability of failure is defined as $P_f \equiv \Phi(-\beta_t)$ with a target reliability index β_t where the failure is defined as $G_i\left\{\mathbf{X};\mathbf{d}(\mathbf{X})\right\}>0$.

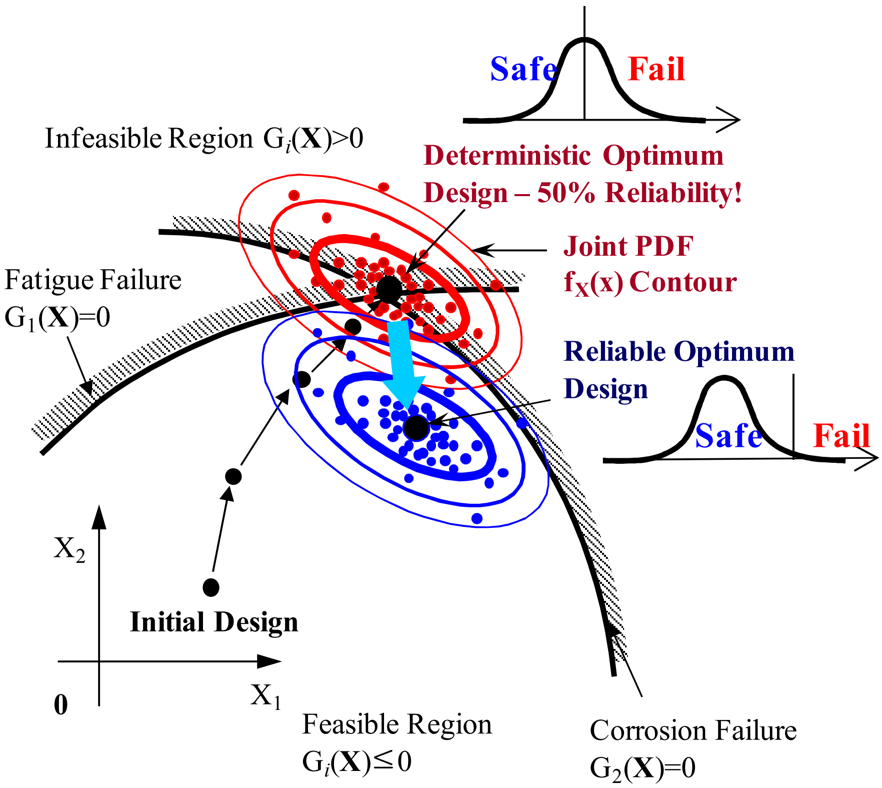


Figure 6.1: Design under Uncertainty

6.2 A General Formulation of Design Optimization

In general, design optimization can be formulated as

$$\begin{array}{lll} \text{Minimize (or Maximize)} & f(\mathbf{x}) & \\ \text{Subject to} & h_i(\mathbf{x}) = 0, & i = 1, \cdots, p \\ & g_j(\mathbf{x}) \le 0, & j = 1, \cdots, m \\ & \mathbf{x}_L \le \mathbf{x} \le \mathbf{x}_U, & \mathbf{x} \in R^n \end{array} \tag{68}$$

where m: no of inequality constraints, feasible where $g_j(\mathbf{x}) \le 0$

p: no of equality constraints, feasible where $h_i(\mathbf{x}) = 0$

6.3 Optimality Condition

Refer to Section 6.1 (Arora, 2004): First-order necessary KKT condition.

- Lagrangian function:

$$L = f(\mathbf{x}) + \sum_{i=1}^{p} v_i h_i(\mathbf{x}) + \sum_{j=1}^{m} u_j g_j(\mathbf{x}) \tag{69}$$

- Gradient conditions

$$\frac{\partial L}{\partial v_i} = 0 \quad \Rightarrow \quad h_i(\mathbf{x}^*) = 0; \quad i = 1 \sim p \tag{70}$$

$$\frac{\partial L}{\partial x_k} = 0 \quad \Rightarrow \quad \frac{\partial f}{\partial x_k} + \sum_{i=1}^{p} v_i^* \frac{\partial h_i}{\partial x_k} + \sum_{j=1}^{m} u_j^* \frac{\partial g_j}{\partial x_k} = 0; \quad k = 1 \sim nd \tag{71}$$

- Feasibility check

$$g_j(\mathbf{x}^*) \le 0; \quad j = 1 \sim m \tag{72}$$

- Switching conditions

$$u_j^* g_j(\mathbf{x}^*) = 0; \quad j = 1 \sim m \tag{73}$$

- Nonnegativity of Lagrange multipliers for inequalities

$$u_j^* \ge 0; \quad j = 1 \sim m \tag{74}$$

- Regularity check
  Gradients of active constraints must be linearly independent. In such a case, the Lagrangian multipliers for the constraints are unique.

Exercise: Check for KKT necessary conditions

Minimize $f(x,y)=(x-10)^2+(y-8)^2$ subject to $g_1 = x+y-12 \le 0, g_2 = x-8 \le 0$

Refer to Example 5.1 (Arora, 2004).
Arora, J.S. Introduction to Optimum Design, Second Edition, Elsevier, 2004

The ordinary optimization task is where many constraints are imposed. In the process of finding a usable-feasible search direction, we are able to detect if the KKT conditions are satisfied. If they are, the optimization process must be terminated.

6.4 Concept of Numerical Algorithms in Design Optimization

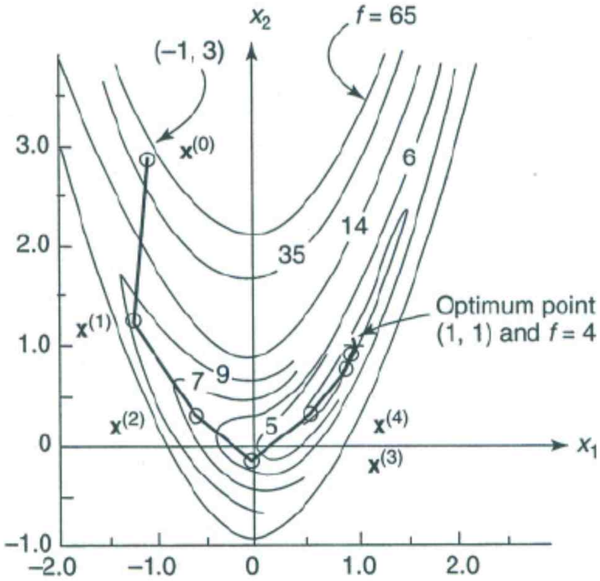


Figure 6.2: Conceptual steps of unconstrained optimization algorithm

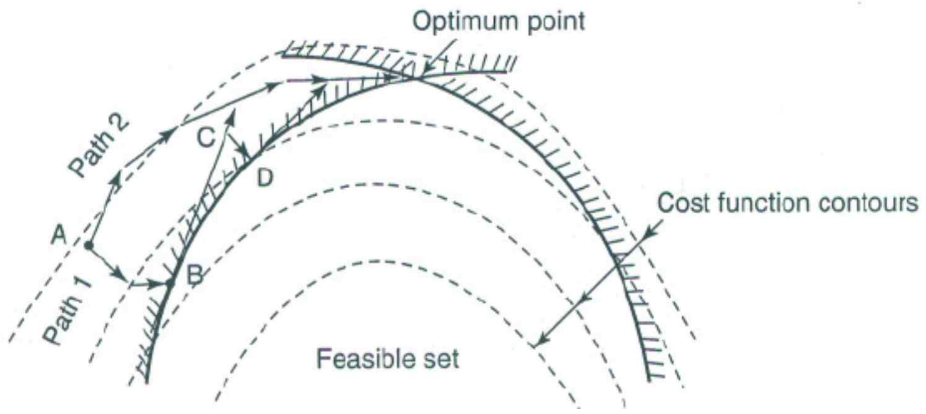


Figure 6.3: Conceptual steps of constrained optimization algorithm

Iterative numerical search methods are employed for the optimization. Two basic calculations are involved in the numerical search methods for optimum design: (1) calculation of a search direction and (2) calculation of a step size in the search direction. It can be generally expressed as

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)} \quad \text{where} \;\; \Delta\mathbf{x}^{(k)} = \alpha_k \mathbf{d}^{(k)} \tag{75}$$

So, finding α_k is a line search and $\mathbf{d}^{(k)}$ is the direction search.

6.4.1 Line Search

The cost function $f(\mathbf{x})$ is given as

$$f(\mathbf{x}^{(k+1)}) = f(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)}) = f(\alpha) \tag{76}$$

It is important to understand this reduction of a function of n variables to a function of one variable. The descent condition for the cost function can be expressed as the inequality:

$$f(\alpha) < f(0) \tag{77}$$

To satisfy the inequality (77), the curve $f(\alpha)$ must have a negative slope when α=0.

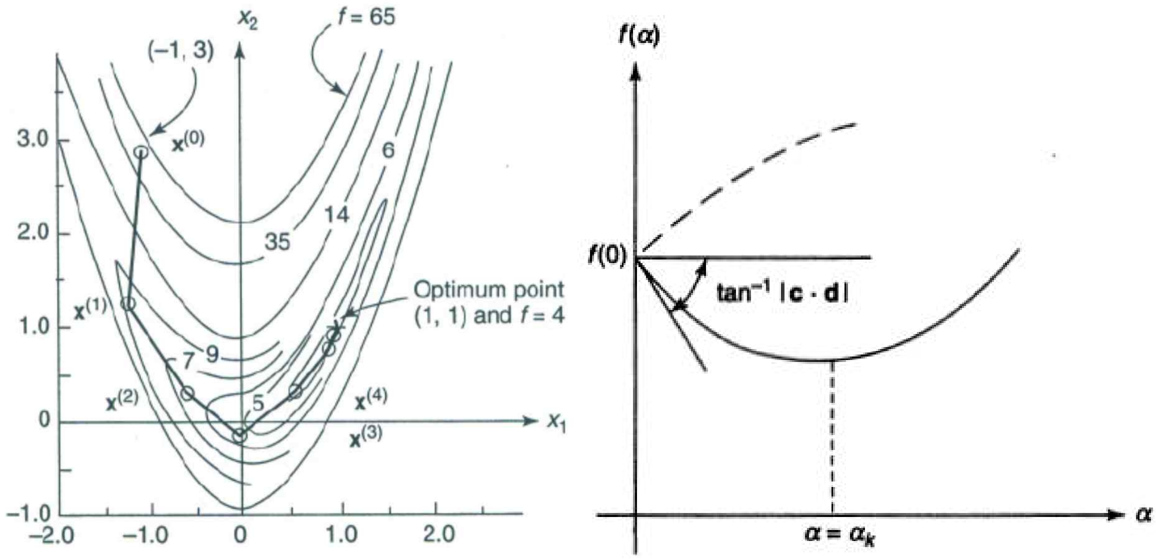


Figure 6.4: Descent condition for the cost function

Let $\nabla f(\mathbf{x})$ be $c(\mathbf{x})$. In fact, the slope of the curve $f(\alpha)$ at α=0 is calculated as $f'(0) = \mathbf{c}^{(k)} \cdot \mathbf{d}^{(k)} < 0$. If $\mathbf{d}^{(k)}$ is a descent direction, then α must always be a positive scalar. Thus, the one-dimensional minimization problem is to find $\alpha_k = \alpha$ such that $f(\alpha)$ is minimized.

The necessary condition for the optimal step size is $df(\alpha)/d\alpha = 0$, and the sufficient condition is $d^2 f(\alpha)/d\alpha^2 > 0$. Note that differentiation of $f(\mathbf{x}^{(k+1)})$ with respect to α gives

$$\left.\frac{df(\mathbf{x})}{d\alpha}\right|_{\mathbf{x}^{(k+1)}} = \frac{df^T(\mathbf{x}^{(k+1)})}{d\mathbf{x}} \frac{d(\mathbf{x}^{(k+1)})}{d\alpha} = \nabla f(\mathbf{x}^{(k+1)}) \cdot \mathbf{d}^{(k)} = \mathbf{c}^{(k+1)} \cdot \mathbf{d}^{(k)} = 0 \tag{78}$$

Analytical Step Size Determination

Let a direction of change for the function

$$f(\mathbf{x}) = 3x_1^2 + 2x_1x_2 + 2x_2^2 + 7 \tag{a}$$

at the point (1, 2) be given as (−1, −1). Compute the step size α_k to minimize $f(\mathbf{x})$ in the given direction.

Solution. For the given point $\mathbf{x}^{(k)} = (1, 2)$, $f(\mathbf{x}^{(k)}) = 22$, and $\mathbf{d}^{(k)} = (-1, -1)$. We first check to see if $\mathbf{d}^{(k)}$ is a direction of descent using Inequality (8.8). The gradient of the function at (1, 2) is given as $\mathbf{c}^{(k)} = (10, 10)$ and $\mathbf{c}^{(k)} \cdot \mathbf{d}^{(k)} = 10(-1) + 10(-1) = -20 < 0$. Therefore, (−1, −1) is a direction of descent. The new point $\mathbf{x}^{(k+1)}$ using Eq. (8.9a) is given as

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^{(k+1)} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \alpha \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad \text{or} \quad x_1^{(k+1)} = 1 - \alpha; \quad x_2^{(k+1)} = 2 - \alpha \tag{b}$$

Substituting these equations into the cost function of Eq. (a), we get

$$f(\mathbf{x}^{(k+1)}) = 3(1-\alpha)^2 + 2(1-\alpha)(2-\alpha) + 2(2-\alpha)^2 + 7 = 7\alpha^2 - 20\alpha + 22 = f(\alpha) \tag{c}$$

Therefore, along the given direction $(-1, -1)$, $f(\mathbf{x})$ becomes a function of the single variable α. Note from Eq. (c) that $f(0) = 22$, which is the cost function value at the current point, and that $f'(0) = -20 < 0$, which is the slope of $f(\alpha)$ at $\alpha = 0$ (also recall that $f'(0) = \mathbf{c}^{(k)} \cdot \mathbf{d}^{(k)}$). Now using the necessary and sufficient conditions of optimality for $f(\alpha)$, we obtain

$$\frac{df}{d\alpha} = 14\alpha_k - 20 = 0; \qquad \alpha_k = \frac{10}{7}; \qquad \frac{d^2 f}{d\alpha^2} = 14 > 0 \tag{d}$$

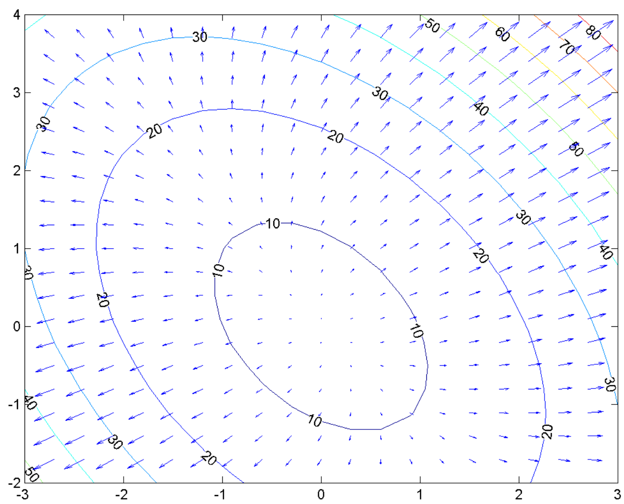
Therefore, $\alpha_k = \frac{10}{7}$ minimizes $f(\mathbf{x})$ in the direction $(-1, -1)$. The new point is

$$\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}^{(k+1)} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} + \left(\frac{10}{7}\right) \begin{bmatrix} -1 \\ -1 \end{bmatrix} = \begin{bmatrix} -\frac{3}{7} \\ \frac{4}{7} \end{bmatrix} \tag{e}$$

Substituting the new design $(-\frac{3}{7}, \frac{4}{7})$ into the cost function $f(\mathbf{x})$ we find the new value of the cost function as $\frac{54}{7}$. This is a substantial reduction from the cost function value of 22 at the previous point. Note that Eq. (d) for calculation of step size α can also be obtained by directly using the condition given in Eq. (8.11). Using Eq. (b), the gradient of f at the new design point in terms of α is given as

$$\mathbf{c}^{(k+1)} = (6x_1 + 2x_2, 2x_1 + 4x_2) = (10 - 8\alpha, 10 - 6\alpha) \tag{f}$$

Using the condition of Eq. (8.11), we get $14\alpha - 20 = 0$ which is same as Eq. (d).



```
>> [X,Y] = meshgrid(-3:.3:3,-2:.3:4);
>> f=3*X.^2+2*X.*Y+2*Y.^2+7;
>> [C,h]=contour(X,Y,f); clabel(C,h); hold on
>> [U,V] = gradient(f,2,2); quiver(X,Y,U,V)
```

Line Search Methods

1. Equal interval search
2. Golden section search
3. Quadratic interpolation method

With the assumption that the function $f(\alpha)$ is sufficiently smooth and unimodal, $f(\alpha)$ is approximated using a quadratic function with respect to α as

$$f(\alpha) \approx q(\alpha) = a_0 + a_1\alpha + a_2\alpha^2 \tag{79}$$

The minimum point $\bar{\alpha}$ of the quadratic curve is calculated by solving the necessary condition $dq/d\alpha = 0$.

One-dimensional Minimization with Quadratic Interpolation

Find the minimum point of $f(\alpha) = 2 - 4\alpha + e^{\alpha}$ of Example 8.3 by polynomial interpolation. Use the golden section search with $\delta = 0.5$ to bracket the minimum point initially.

Iteration 1. From Example 8.3 the following information is known.

$\alpha_l = 0.5$, $\alpha_i = 1.309017$, $\alpha_u = 2.618034$

$f(\alpha_l) = 1.648721$, $f(\alpha_i) = 0.466464$, $f(\alpha_u) = 5.236610$

$$a_2 = \frac{1}{1.30902}\left(\frac{3.5879}{2.1180} - \frac{-1.1823}{0.80902}\right) = 2.410$$

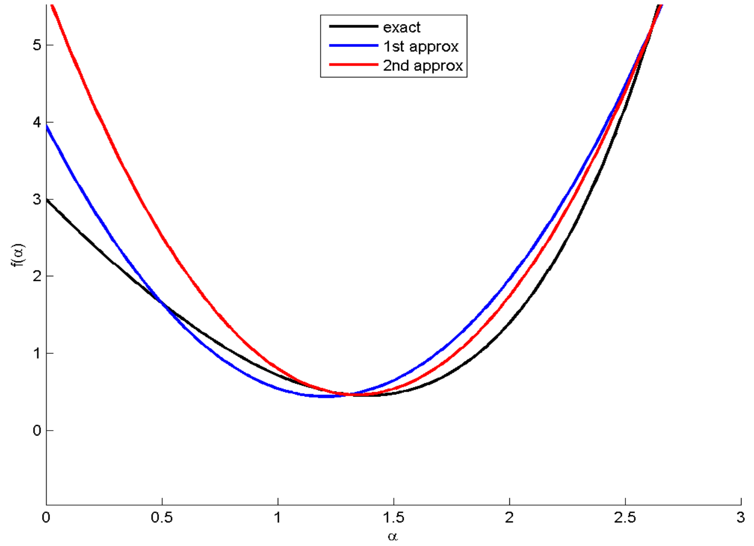
$$a_1 = \frac{-1.1823}{0.80902} - (2.41)(1.80902) = -5.821$$

$$a_0 = 1.648271 - (-5.821)(0.50) - 2.41(0.25) = 3.957$$

Therefore, $\bar{\alpha} = 1.2077$ from Eq. (9.3), and $f(\bar{\alpha}) = 0.5149$. Note that $\bar{\alpha} < \alpha_i$ and $f(\alpha_i) < f(\bar{\alpha})$. Thus, new limits of the reduced interval of uncertainty are $\alpha_l' = \bar{\alpha} = 1.2077$, $\alpha_u' = \alpha_u = 2.618034$, and $\alpha_i' = \alpha_i = 1.309017$.

$\alpha_l = 1.2077$, $\alpha_i = 1.309017$, $\alpha_u = 2.618034$

The coefficients a_0, a_1, and a_2 are calculated as before, $a_0 = 5.7129$, $a_1 = -7.8339$, and $a_2 = 2.9228$. Thus, $\bar{\alpha} = 1.34014$ and $f(\bar{\alpha}) = 0.4590$.



Homework 21: *Optimization Reading 1*

Chapters 4.3-4.5
Chapters 5.1-5.2
Chapters 8.1, 8.2

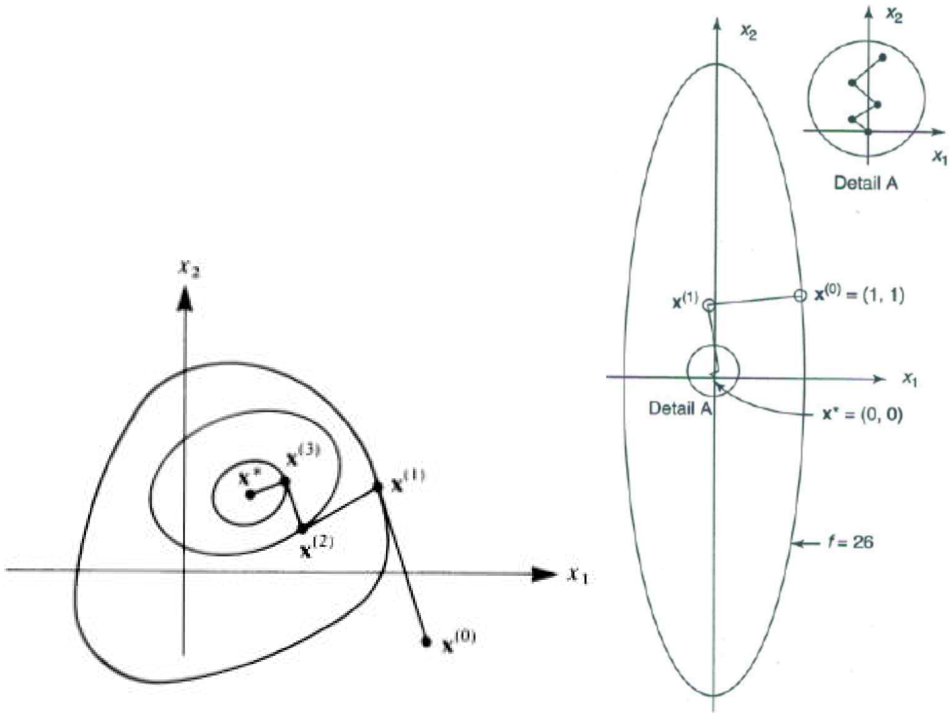
6.4.2 Direction Search

The basic requirement for **d** is that the cost function be reduced if we make a small move along **d**; that is, the descent condition ($f'(0) = \mathbf{c}^{(k)} \cdot \mathbf{d}^{(k)} < 0$) be satisfied. This is called the descent direction.

<u>Search Direction Methods</u>

1. Steepest descent method
2. Conjugate gradient method
3. Newton's Method
4. Quasi-Newton's Method
5. Sequential linear programming (SLP)
6. Sequential quadratic programming (SQP)

The first four are used for an unconstrained optimization problem whereas the last two are often used for a constrained optimization problem.



(a) Steepest descent method (b) Conjugate gradient method

Figure 6.5: Search direction methods using gradient method

a. Steepest decent method

Step 1. Estimate a starting design $\mathbf{x}^{(0)}$ and set the iteration counter $k = 0$. Select a convergence parameter $\varepsilon > 0$.

Step 2. Calculate the gradient of $f(\mathbf{x})$ at the point $\mathbf{x}^{(k)}$ as $\mathbf{c}^{(k)} = \nabla f(\mathbf{x}^{(k)})$.

Step 3. Calculate $\|\mathbf{c}^{(k)}\|$. If $\|\mathbf{c}^{(k)}\| < \varepsilon$, then stop the iterative process because $\mathbf{x}^* = \mathbf{x}^{(k)}$ is a minimum point. Otherwise, continue.

Step 4. Let the search direction at the current point $\mathbf{x}^{(k)}$ be $\mathbf{d}^{(k)} = -\mathbf{c}^{(k)}$.

Step 4. Let the search direction at the current point $\mathbf{x}^{(k)}$ be $\mathbf{d}^{(k)} = -\mathbf{c}^{(k)}$.

Step 5. Calculate a step size α_k that minimizes $f(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)})$. Any one-dimensional search algorithm may be used to determine α_k.

Step 6. Update the design as $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{d}^{(k)}$. Set $k = k + 1$, and go to Step 2.

EXAMPLE 8.4 Use of Steepest Descent Algorithm

Minimize $f(x_1, x_2) = x_1^2 + x_2^2 - 2x_1x_2$ using the steepest descent method starting from the point (1, 0).

1. The starting design is given as $\mathbf{x}^{(0)} = (1, 0)$.
2. $\mathbf{c}^{(0)} = (2x_1 - 2x_2, 2x_2 - 2x_1) = (2, -2)$.
3. $\|\mathbf{c}^{(0)}\| = 2\sqrt{2} \neq 0$
4. Set $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = (-2, 2)$.
5. Calculate α to minimize $f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)})$ where $\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)} = (1 - 2\alpha, 2\alpha)$:

$$f(\mathbf{x}^{(0)} + \alpha\mathbf{d}^{(0)}) = (1-2\alpha)^2 + (2\alpha)^2 + (2\alpha)^2 - 2(1-2\alpha)(2\alpha)$$
$$= 16\alpha^2 - 8\alpha + 1 = f(\alpha)$$

Using the analytic approach

$$\frac{df(\alpha)}{d\alpha} = 0; \qquad 32\alpha - 8 = 0 \qquad \text{or} \qquad \alpha_0 = 0.25 \qquad\qquad \frac{d^2 f(\alpha)}{d\alpha^2} = 32 > 0.$$

6. Updating the design $(\mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)})$: $x_1^{(1)} = 1 - 0.25(2) = 0.5$, $x_2^{(1)} = 0 + 0.25(2) = 0.5$ Solving for $\mathbf{c}^{(1)}$ from the expression in Step 2, we see that $\mathbf{c}^{(1)} = (0, 0)$, which satisfies the stopping criterion. Therefore, (0.5, 0.5) is a minimum point for $f(\mathbf{x})$ and $f^* = 0$.

b. Conjugate gradient method

Actually, the *conjugate gradient directions* $\mathbf{d}^{(i)}$ are orthogonal with respect to a symmetric and positive definite matrix $\mathbf{A}$, i.e., $\mathbf{d}^{(i)T}\mathbf{A}\mathbf{d}^{(j)} = 0$ for all i and j, $i \neq j$. The *conjugate gradient algorithm* is stated as follows:

Step 1. Estimate a starting design as $\mathbf{x}^{(0)}$. Set the iteration counter $k = 0$. Select the convergence parameter ε. Calculate

$$\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = -\nabla f(\mathbf{x}^{(0)}) \tag{8.21a}$$

Check stopping criterion. If $\|\mathbf{c}^{(0)}\| < \varepsilon$, then stop. Otherwise, go to Step 4 (note that Step 1 of the conjugate gradient and the steepest descent methods is the same).

Step 2. Compute the gradient of the cost function as $\mathbf{c}^{(k)} = \nabla f(\mathbf{x}^{(k)})$.
Step 3. Calculate $\|\mathbf{c}^{(k)}\|$. If $\|\mathbf{c}^{(k)}\| < \varepsilon$, then stop; otherwise continue.
Step 4. Calculate the new conjugate direction as

$$\mathbf{d}^{(k)} = -\mathbf{c}^{(k)} + \beta_k \mathbf{d}^{(k-1)}; \qquad \beta_k = \left(\|\mathbf{c}^{(k)}\| / \|\mathbf{c}^{(k-1)}\|\right)^2 \tag{8.21b}$$

Step 5. Compute a step size $\alpha_k = \alpha$ to minimize $f(\mathbf{x}^{(k)}\ \alpha\mathbf{d}^{(k)})$.
Step 6. Change the design as follows, set $k = k + 1$ and go to Step 2.

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)} \tag{8.22}$$

EXAMPLE 8.6 Use of Conjugate Gradient Algorithm

Consider the problem solved in Example 8.5: minimize $f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3$

Carry out two iterations of the conjugate gradient method starting from the design (2, 4, 10).

The first iteration $\mathbf{c}^{(0)} = (12, 40, 48)$; $\|\mathbf{c}^{(0)}\| = 63.6$, $f(\mathbf{x}^{(0)}) = 332.0$

$$\mathbf{x}^{(1)} = (0.0956, -2.348, 2.381)$$

The second iteration

2. $\mathbf{c}^{(1)} = (-4.5, -4.438, 4.828)$, $f(\mathbf{x}^{(1)}) = 10.75$
3. $\|\mathbf{c}^{(1)}\| = 7.952 > \varepsilon$, so continue.
4. $\beta_1 = [\|\mathbf{c}^{(1)}\|/\mathbf{c}^{(0)}]^2 = (7.952/63.3)^2 = 0.015633$

$$\mathbf{d}^{(1)} = -\mathbf{c}^{(1)} + \beta_1\mathbf{d}^{(0)} = \begin{bmatrix} 4.500 \\ 4.438 \\ -4.828 \end{bmatrix} + (0.015633)\begin{bmatrix} -12 \\ -40 \\ -48 \end{bmatrix} = \begin{bmatrix} 4.31241 \\ 3.81268 \\ -5.57838 \end{bmatrix}$$

5. Step size in the direction $\mathbf{d}^{(1)}$ is calculated as $\alpha = 0.3156$.
6. The design is updated as $\mathbf{x}^{(2)} = \begin{bmatrix} 0.0956 \\ -2.348 \\ 2.381 \end{bmatrix} + \alpha \begin{bmatrix} 4.31241 \\ 3.81268 \\ -5.57838 \end{bmatrix} = \begin{bmatrix} 1.4566 \\ -1.1447 \\ 0.6205 \end{bmatrix}$

Calculating the gradient at this point, we get $\mathbf{c}^{(2)} = (0.6238, -0.4246, 0.1926)$. $\|\mathbf{c}^2\| = 0.7788 > \varepsilon$, so we need to continue the iterations. Note that $\mathbf{c}^{(2)} \cdot \mathbf{d}^{(1)} = 0$.

TABLE 8-3 Optimum Solution for Example 8.6 with the Conjugate Gradient Method: $f(x_1, x_2, x_3) = x_1^2 + 2x_2^2 + 2x_3^2 + 2x_1x_2 + 2x_2x_3$

Starting values of design variables:	2, 4, 10
Optimum design variables:	−6.4550E−10, −5.8410E−10, 1.3150E−10.
Optimum cost function value:	6.8520E−20.
Norm of the gradient at optimum:	3.0512E−05.
Number of iterations:	4
Number of function evaluations:	10

Homework 21: *Optimization Reading 2*

Chapters 8.3, 8.4

c. Newton's method

The basic idea of the Newton's method is to use a second-order Taylor's expansion of the function about the current design point.

$$f(\mathbf{x}+\Delta\mathbf{x}) = f(\mathbf{x}) + \mathbf{c}^T\Delta\mathbf{x} + 0.5\Delta\mathbf{x}^T\mathbf{H}\Delta\mathbf{x} \tag{80}$$

The optimality conditions ($\partial f/\partial(\Delta\mathbf{x}) = 0$) for the function above

$$\mathbf{c} + \mathbf{H}\Delta\mathbf{x} = 0 \qquad \Delta\mathbf{x} = -\mathbf{H}^{-1}\mathbf{c} \tag{81}$$

The optimal step size must be calculated for design optimization.

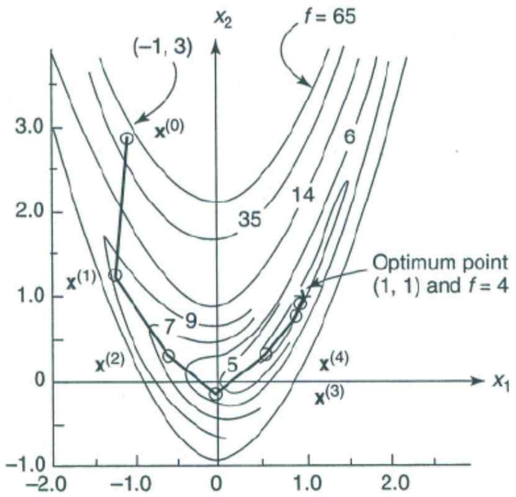
Use of Modified Newton's Method

$$\text{Minimize } f(\mathbf{x}) = 10x_1^4 - 20x_1^2x_2 + 10x_2^2 + x_1^2 - 2x_1 + 5 \quad \text{(a)}$$

using the computer program for the modified Newton's method given in Appendix D from the point (−1, 3). Golden section search may be used for step size determination with $\delta = 0.05$ and line search accuracy equal to 0.0001. For the stopping criterion, use $\varepsilon = 0.005$.

$$\mathbf{c} = \nabla f(\mathbf{x}) = (40x_1^3 - 40x_1x_2 + 2x_1 - 2, -20x_1^2 + 20x_2) \quad \text{(b)}$$

$$\mathbf{H} = \nabla^2 f(\mathbf{x}) = \begin{bmatrix} 120x_1^2 - 40x_2 + 2 & -40x_1 \\ -40x_1 & 20 \end{bmatrix} \quad \text{(c)}$$



Optimum Solution with Modified Newton's Method

Starting point:	−1, 3
Optimum design variables:	9.99880E−01, 9.99681E−01
Optimum cost function value:	4.0
Norm of gradient at optimum:	3.26883E−03
Number of iterations:	8
Number of function evaluations:	198

- M-file

```
function opt()
x0 = [-1;3];
x = x0;
[x,fval,exitflag,output,grad]= fminunc(@obj,x0);
end

function f = obj(x)
```

```
    f = 10*x(1)^4-20*x(1)^2*x(2)+10*x(2)^2+x(1)^2-2*x(1)+5
    end
```

The *drawbacks* of the modified Newton's method for general applications are:

1. It requires calculations of second-order derivatives at each iteration, which is usually quite time consuming. In some applications it may not even be possible to calculate such derivatives. Also, a linear system of equations in Eq. (9.11) needs to be solved. Therefore, each iteration of the method requires substantially more calculations compared with the steepest descent or conjugate gradient method.
2. The Hessian of the cost function may be singular at some iterations. Thus, Eq. (9.11) cannot be used to compute the search direction. Also, unless the Hessian is positive definite, the search direction cannot be guaranteed to be that of descent for the cost function, as discussed earlier.
3. The method is not convergent unless the Hessian remains positive definite and a step size is calculated along the search direction to update design. However, the method has a quadratic rate of convergence when it converges. For a strictly convex quadratic function, the method converges in just one iteration from any starting design.

Comparison of Steepest Descent, Conjugate Gradient, and Modified Newton Methods

Minimize $f(\mathbf{x}) = 50(x_2 - x_1^2)^2 + (2 - x_1)^2$ starting from the point (5, –5). Use the steepest descent, Newton, and conjugate gradient methods, and compare their performance.

TABLE 9-3 Evaluation of Three Methods for Example 9.8: $f(\mathbf{x}) = 50(x_2 - x_1^2)^2 + (2 - x_1)^2$

	Steepest descent	*Conjugate gradient*	*Modified Newton*
x_1	1.9941	2.0000	2.0000
x_2	3.9765	3.9998	3.9999
f	3.4564E–05	1.0239E–08	2.5054E–10
$\lVert c \rVert$	3.3236E–03	1.2860E–04	9.0357E–04
No. of function evaluations	138,236	65	349
No. of iterations	9670	22	13

d. Quasi-Newton Method

Only the first derivatives of the function are used to generate these Newton approximations. Therefore the methods have desirable features of both the conjugate gradient and the Newton's methods. They are called *quasi-Newton methods*.

$$\mathbf{c} + \mathbf{H}\Delta\mathbf{x} = 0 \qquad \Delta\mathbf{x} = -\mathbf{H}^{-1}\mathbf{c} \tag{82}$$

There are several ways to approximate the Hessian or its inverse. The basic idea is to update the current approximation of the Hessian using two pieces of information: the gradient vectors and their changes in between two successive iterations. While updating, the properties of symmetry and positive definiteness are preserved. Positive definiteness is essential because the search direction may not be a descent direction for the cost function with the property.

Hessian Updating: BFGS (Broyden-Fletcher-Goldfarb-Shanno) Method

Step 1. Estimate an initial design $\mathbf{x}^{(0)}$. Choose a symmetric positive definite $n \times n$ matrix $\mathbf{H}^{(0)}$ as an estimate for the Hessian of the cost function. In the absence of more information, let $\mathbf{H}^{(0)} = \mathbf{I}$. Choose a convergence parameter ε. Set $k = 0$, and compute the gradient vector as $\mathbf{c}^{(0)} = \nabla f(\mathbf{x}^{(0)})$.

Step 2. Calculate the norm of the gradient vector as $\|\mathbf{c}^{(k)}\|$. If $\|\mathbf{c}^{(k)}\| < \varepsilon$ then stop the iterative process; otherwise continue.

Step 3. Solve the linear system of equations $\mathbf{H}^{(k)}\mathbf{d}^{(k)} = -\mathbf{c}^{(k)}$ to obtain the search direction.

Step 4. Compute optimum step size $\alpha_k = \alpha$ to minimize $f(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)})$.

Step 5. Update the design as $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k\mathbf{d}^{(k)}$

Step 6. Update the Hessian approximation for the cost function as

$$\mathbf{H}^{(k+1)} = \mathbf{H}^{(k)} + \mathbf{D}^{(k)} + \mathbf{E}^{(k)} \tag{a}$$

where the correction matrices $\mathbf{D}^{(k)}$ and $\mathbf{E}^{(k)}$ are given as

$$\mathbf{D}^{(k)} = \frac{\mathbf{y}^{(k)}\mathbf{y}^{(k)T}}{(\mathbf{y}^{(k)} \cdot \mathbf{s}^{(k)})}; \qquad \mathbf{E}^{(k)} = \frac{\mathbf{c}^{(k)}\mathbf{c}^{(k)T}}{(\mathbf{c}^{(k)} \cdot \mathbf{d}^{(k)})} \tag{b}$$

$$\mathbf{s}^{(k)} = \alpha_k\mathbf{d}^{(k)} \text{ (change in design);} \qquad \mathbf{y}^{(k)} = \mathbf{c}^{(k+1)} - \mathbf{c}^{(k)} \text{ (change in gradient);} \quad \mathbf{c}^{(k+1)} = \nabla f(\mathbf{x}^{(k+1)}) \tag{c}$$

Step 7. Set $k = k + 1$ and go to Step 2.

Example of BFGS Method

Execute two iterations of the BFGS method for the problem:

minimize $f(\mathbf{x}) = 5x_1^2 + 2x_1x_2 + x_2^2 + 7$ starting from the point (1, 2).

Solution. We shall follow steps of the algorithm. Note that the first iteration gives steepest descent step for the cost function.

Iteration 1 (k = 0).

1. $\mathbf{x}^{(0)}(1, 2)$, $\mathbf{H}^{(0)} = \mathbf{I}$, $\varepsilon = 0.001$, $k = 0$
   $\mathbf{c}^{(0)} = (10x_1 + 2x_2, 2x_1 + 2x_2) = (14, 6)$
2. $\|\mathbf{c}^{(0)}\| = \sqrt{14^2 + 6^2} = 15.232 > \varepsilon$, so continue
3. $\mathbf{d}^{(0)} = -\mathbf{c}^{(0)} = (-14, -6)$; since $\mathbf{H}^{(0)} = \mathbf{I}$
4. Step size determination (same as Example 9.9): $\alpha_0 = 0.099$
5. $\mathbf{x}^{(1)} = \mathbf{x}^{(0)} + \alpha_0\mathbf{d}^{(0)} = (-0.386, 1.407)$
6. $\mathbf{s}^{(0)} = \alpha_0\mathbf{d}^{(0)} = (-1.386, 0.593)$; $\mathbf{c}^{(1)} = (-1.046, 2.042)$

$$\mathbf{y}^{(0)} = \mathbf{c}^{(1)} - \mathbf{c}^{(0)} = (-15.046, -3.958); \quad \mathbf{y}^{(0)} \cdot \mathbf{s}^{(0)} = 23.20; \quad \mathbf{c}^{(0)} \cdot \mathbf{d}^{(0)} = -232.0 \quad \text{(a)}$$

$$\mathbf{y}^{(0)}\mathbf{y}^{(0)T} = \begin{bmatrix} 226.40 & 59.55 \\ 59.55 & 15.67 \end{bmatrix}; \qquad \mathbf{D}^{(0)} = \frac{\mathbf{y}^{(0)}\,\mathbf{y}^{(0)T}}{\mathbf{y}^{(0)} \cdot \mathbf{s}^{(0)}} = \begin{bmatrix} 9.760 & 2.567 \\ 2.567 & 0.675 \end{bmatrix} \quad \text{(b)}$$

$$\mathbf{c}^{(0)}\mathbf{c}^{(0)T} = \begin{bmatrix} 196 & 84 \\ 84 & 36 \end{bmatrix}; \qquad \mathbf{E}^{(0)} = \frac{\mathbf{c}^{(0)}\,\mathbf{c}^{(0)T}}{\mathbf{c}^{(0)} \cdot \mathbf{d}^{(0)}} = \begin{bmatrix} -0.845 & -0.362 \\ -0.362 & -0.155 \end{bmatrix} \quad \text{(c)}$$

$$\mathbf{H}^{(1)} = \mathbf{H}^{(0)} + \mathbf{D}^{(0)} + \mathbf{E}^{(0)} = \begin{bmatrix} 9.915 & 2.205 \\ 2.205 & 0.520 \end{bmatrix} \quad \text{(d)}$$

Iteration 2 (k = 1).

2. $\|\mathbf{c}^{(1)}\| = 2.29 > \varepsilon$, so continue
3. $\mathbf{H}^{(1)}\mathbf{d}^{(1)} = -\mathbf{c}^{(1)}$; or, $\mathbf{d}^{(1)} = (17.20, -76.77)$
4. Step size determination: $\alpha_1 = 0.018455$
5. $\mathbf{x}^{(2)} = \mathbf{x}^{(1)} + \alpha_1\mathbf{d}^{(1)} = (-0.0686, -0.0098)$
6. $\mathbf{s}^{(1)} = \alpha_1\mathbf{d}^{(1)} = (0.317, -1.417)$; $\mathbf{c}^{(2)} = (-0.706, -0.157)$ (e)

$$\mathbf{y}^{(1)} = \mathbf{c}^{(2)} - \mathbf{c}^{(1)} = (0.317, -2.199); \quad \mathbf{y}^{(1)} \cdot \mathbf{s}^{(1)} = 3.224; \quad \mathbf{c}^{(1)} \cdot \mathbf{d}^{(1)} = -174.76 \quad \text{(f)}$$

$$\mathbf{y}^{(1)}\mathbf{y}^{(1)T} = \begin{bmatrix} 0.1156 & -0.748 \\ -0.748 & 4.836 \end{bmatrix}; \qquad \mathbf{D}^{(1)} = \frac{\mathbf{y}^{(1)}\,\mathbf{y}^{(1)T}}{\mathbf{y}^{(1)} \cdot \mathbf{s}^{(1)}} = \begin{bmatrix} 0.036 & -0.232 \\ -0.232 & 1.500 \end{bmatrix} \quad \text{(g)}$$

$$\mathbf{c}^{(1)}\mathbf{c}^{(1)T} = \begin{bmatrix} 1.094 & -2.136 \\ -2.136 & 4.170 \end{bmatrix}; \qquad \mathbf{E}^{(1)} = \frac{\mathbf{c}^{(1)}\,\mathbf{c}^{(1)T}}{\mathbf{c}^{(1)} \cdot \mathbf{d}^{(1)}} = \begin{bmatrix} -0.0063 & 0.0122 \\ 0.0122 & -0.0239 \end{bmatrix} \quad \text{(h)}$$

$$\mathbf{H}^{(2)} = \mathbf{H}^{(1)} + \mathbf{D}^{(1)} + \mathbf{E}^{(1)} = \begin{bmatrix} 9.945 & 1.985 \\ 1.985 & 1.996 \end{bmatrix} \quad \text{(i)}$$

It can be verified that $\mathbf{H}^{(2)}$ is quite close to the Hessian of the given cost function. One more iteration of the BFGS method will yield the optimum solution of (0, 0).

Table 6.1: Summary of Numerical Aspects in Unconstrained Optimization Algorithms

Methods	Steepest	Conjugate	Newton	Quasi Newton
Requirements	Function, Gradient	Function, Gradient	Function, Gradient, Hessian	Function, Gradient
Stability	Good	Good	Good	Good
Efficiency	Bad	Good	Bad	Good
Speed	Bad	Good	Good	Good

Let us recall a constrained design optimization formulated as

$$\begin{aligned} &\text{Minimize} && f(\mathbf{x}) \\ &\text{Subject to} && h_i(\mathbf{x}) = 0, \quad i = 1, \cdots, p \\ & && g_j(\mathbf{x}) \le 0, \quad j = 1, \cdots, m \\ & && \mathbf{x}_L \le \mathbf{x} \le \mathbf{x}_U, \quad \mathbf{x} \in R^n \end{aligned} \tag{83}$$

where m: no of inequality constraints, feasible where $g_j(\mathbf{x}) \le 0$

p: no of equality constraints, feasible where $h_i(\mathbf{x}) = 0$

Homework 22: *Optimization Reading 3*

Chapters 9.1, 9.3, 9.4.1, 9.4.2, 9.5

6.5 Sequential Linear Programmning (SLP)

At each iteration, most numerical methods for constrained optimization compute design change by solving an approximate subproblem that is obtained by writing linear Taylor's expansions for the cost and constraint functions.

$$\begin{aligned} &\text{Minimize} && f(\mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)}) \cong f(\mathbf{x}^{(k)}) + \nabla f^T(\mathbf{x}^{(k)})\Delta\mathbf{x}^{(k)} \\ &\text{Subject to} && h(\mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)}) \cong h(\mathbf{x}^{(k)}) + \nabla h^T(\mathbf{x}^{(k)})\Delta\mathbf{x}^{(k)} = 0, \quad i = 1, \cdots, p \\ & && g(\mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)}) \cong g(\mathbf{x}^{(k)}) + \nabla g^T(\mathbf{x}^{(k)})\Delta\mathbf{x}^{(k)} \le 0, \quad j = 1, \cdots, m \\ & && \mathbf{x}_L \le \mathbf{x} \le \mathbf{x}_U, \quad \mathbf{x} \in R^n \end{aligned} \tag{84}$$

The linearization of the problem can be rewritten in a simple form as

$$\begin{aligned} &\text{Minimize} \quad f \cong \bar{f} = \sum_{i=1}^{nd} c_i d_i = \mathbf{c}^T \mathbf{d} \\ &\text{Subject to} \quad h \cong \bar{h} = \sum_{i=1}^{nd} n_{ij} d_i = e_j; \quad i = 1, \cdots, p \quad \Rightarrow \quad \mathbf{N}^T \mathbf{d} = \mathbf{e} \\ &\qquad g \cong \bar{g} = \sum_{i=1}^{nd} a_{ij} d_i \le b_j, \quad j = 1, \cdots, m \quad \Rightarrow \quad \mathbf{A}^T \mathbf{d} \le \mathbf{b} \\ &\qquad \mathbf{x}_L \le \mathbf{x} \le \mathbf{x}_U, \quad \mathbf{x} \in R^n \end{aligned} \tag{85}$$

It must be noted that the problem may not have a bounded solution, or the changes in design may become too large. Therefore, limits must be imposed on changes in design. Such constraints are usually called “move limits”, expressed as

$$-\Delta_{il}^{(k)} \le d_i \le \Delta_{iu}^{(k)} \qquad i = 1 \text{ to } n \tag{86}$$

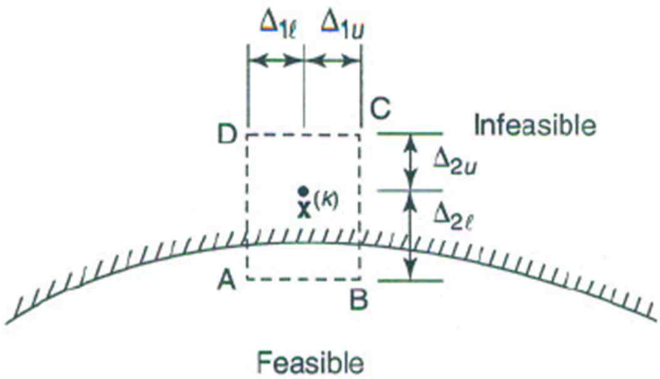


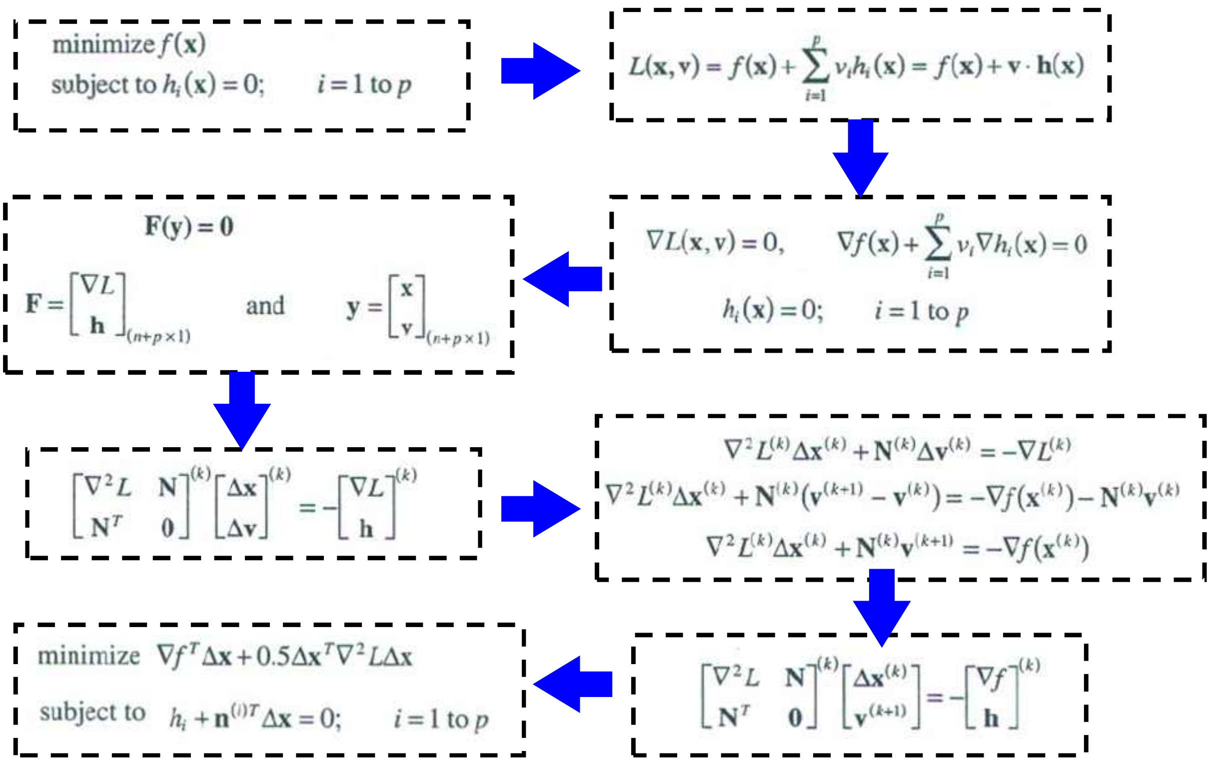
Figure 6.6: Linear move limits on design changes

1. *The method should not be used as a black box approach for engineering design problems.* The selection of move limits is a trial and error process and can be best achieved in an interactive mode. The move limits can be too restrictive resulting in no solution for the LP subproblem. Move limits that are too large can cause oscillations in the design point during iterations. Thus performance of the method depends heavily on selection of move limits.
2. *The method may not converge to the precise minimum* since no descent function is defined, and line search is not performed along the search direction to compute a step size. Thus progress toward the solution point cannot be monitored.
3. *The method can cycle between two points* if the optimum solution is not a vertex of the feasible set.
4. The method is quite simple conceptually as well as numerically. Although it may not be possible to reach the precise optimum with the method, it can be used to obtain improved designs in practice.

6.6 Sequential Quadratic Programmning (SQP)

There are several ways to derive the quadratic programming (QP) subproblem that has to be solved at each optimization iteration. The QP subproblem can be defined as

$$\begin{aligned} &\text{Minimize} \quad f \cong \bar{f} = \mathbf{c}^T\mathbf{d} + \frac{1}{2}\mathbf{d}^T\mathbf{H}\mathbf{d} \\ &\text{Subject to} \quad \mathbf{N}^T\mathbf{d} = \mathbf{e} \\ &\qquad\qquad\quad \mathbf{A}^T\mathbf{d} \le \mathbf{b} \\ &\qquad\qquad\quad \mathbf{d}_L \le \mathbf{d} \le \mathbf{d}_U, \quad \mathbf{d} \in R^n \end{aligned} \tag{87}$$



The Hessian matrix can be updated using the quasi-Newton method. The optimization with the equality constraints can be extended to that with both equality and inequality constraints. There is no need to define a move limit unlike SLP.

$$\begin{aligned} &\text{minimize} \quad \nabla f^T \Delta\mathbf{x} + 0.5\Delta\mathbf{x}^T \nabla^2 L \Delta\mathbf{x} \\ &\text{subject to} \quad h_i + \mathbf{n}^{(i)T} \Delta\mathbf{x} = 0; \qquad i = 1 \text{ to } p \end{aligned}$$

$$\Downarrow$$

$$\begin{aligned} &\text{minimize } \bar{f} = \mathbf{c}^T\mathbf{d} + 0.5\mathbf{d}^T\mathbf{H}\mathbf{d} \\ &\text{subject to} \qquad \mathbf{n}^{(i)^T}\mathbf{d} = e_i; \qquad i = 1 \text{ to } p \\ &\qquad\qquad\qquad\; \mathbf{a}^{(i)^T}\mathbf{d} \le b_i; \qquad i = 1 \text{ to } m \end{aligned}$$

Homework 22: *Optimization Reading 4*

Chapters 9.1, 9.3, 9.4.1, 9.4.2, 9.5

Homework 23: *Design Optimization of Crashworthiness Problem*

A vehicle side impact problem is considered for design optimization. All the design variables are shown in Table 6.2. In this example, the weight of the vehicle is treated as an objective function with ten constraints defined in Table 5.3.

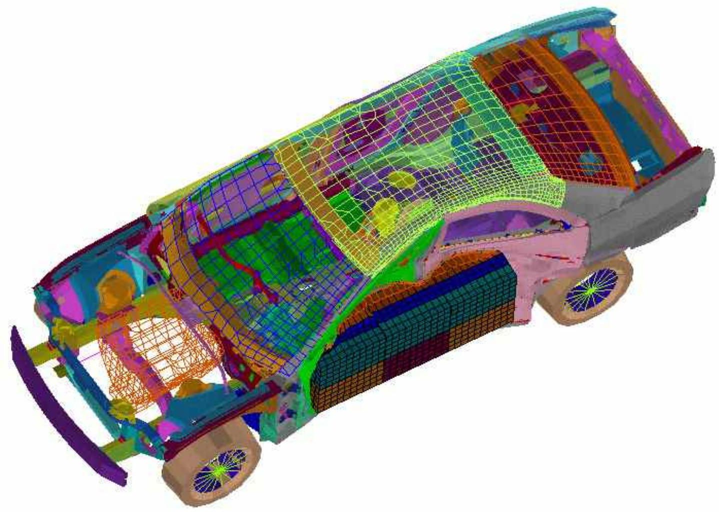


Table 6.2: Properties of design variables
(X_{10} and X_{11} have "0" value)

Random Variables	d^L	d	d^U
X_1	0.500	1.000	1.500
X_2	0.500	1.000	1.500
X_3	0.500	1.000	1.500
X_4	0.500	1.000	1.500
X_5	0.500	1.000	1.500
X_6	0.500	1.000	1.500
X_7	0.500	1.000	1.500
X_8	0.192	0.300	0.345
X_9	0.192	0.300	0.345
X_{10} X_{11}	X_{10} and X_{11} are not design variables		

Table 6.3: Design variables and their bounds

Constraints G_j		Safety Criteria G_j^c
G1: Abdomen load (kN)		≤ 1
G2-G4: Rib deflection (mm)	Upper	≤ 32
	Middle	
	Lower	
G5-G7: VC (m/s)	Upper	≤ 0.32
	Middle	
	Lower	
G8: Pubic symphysis force (kN)		≤ 4
G9: Velocity of B-pillar		≤ 9.9
G10: Velocity of front door at B-pillar		≤ 15.7

Responses:

Cost(weight) = 1.98+4.90*x(1)+6.67*x(2)+6.98*x(3)+4.01*x(4)+1.78*x(5)+2.73*x(7)

G1= (1.16-0.3717*x(2)*x(4)-0.00931*x(2)*x(10)-
0.484*x(3)*x(9)+0.01343*x(6)*x(10))-1;

G2 = (28.98+3.818*x(3)-4.2*x(1)*x(2)+0.0207*x(5)*x(10)+6.63*x(6)*x(9)-
7.7*x(7)*x(8)+0.32*x(9)*x(10))-32;

G3= (33.86+2.95*x(3)+0.1792*x(10)-5.057*x(1)*x(2)-11*x(2)*x(8)-
0.0215*x(5)*x(10)-9.98*x(7)*x(8)+22*x(8)*x(9))-32;

G4 = (46.36-9.9*x(2)-12.9*x(1)*x(8)+0.1107*x(3)*x(10))-32;

G5 = (0.261-0.0159*x(1)*x(2)-0.188*x(1)*x(8)-

```
0.019*x(2)*x(7)+0.0144*x(3)*x(5)+0.0008757*x(5)*x(10)+0.08045*x(6)*x(9)+0.00
139*x(8)*x(11)+0.00001575*x(10)*x(11))-0.32;

G6 = (0.214+0.00817*x(5)-0.131*x(1)*x(8)-0.0704*x(1)*x(9)+ 0.03099*x(2)*x(6)-
0.018*x(2)*x(7)+0.0208*x(3)*x(8)+ 0.121*x(3)*x(9)-
0.00364*x(5)*x(6)+0.0007715*x(5)*x(10)-
0.0005354*x(6)*x(10)+0.00121*x(8)*x(11)+0.00184*x(9)*x(10)- 0.018*x(2).^2)-
0.32;

G7 = (0.74-0.61*x(2)-0.163*x(3)*x(8)+0.001232*x(3)*x(10)-
0.166*x(7)*x(9)+0.227*x(2).^2)-0.32;

G8 = (4.72-0.5*x(4)-0.19*x(2)*x(3)-
0.0122*x(4)*x(10)+0.009325*x(6)*x(10)+0.000191*x(11).^2)-4;

G9 = (10.58-0.674*x(1)*x(2)-1.95*x(2)*x(8)+0.02054*x(3)*x(10)-
0.0198*x(4)*x(10)+0.028*x(6)*x(10))-9.9;

G10 = (16.45-0.489*x(3)*x(7)-0.843*x(5)*x(6)+0.0432*x(9)*x(10)-
0.0556*x(9)*x(11)-0.000786*x(11).^2)-15.7;
```

The Design Optimization is formulated as

$$\begin{aligned} &\text{Minimize} \quad f(\mathbf{x}) \\ &\text{Subject to} \quad g_j(\mathbf{x}) = G_j(\mathbf{x}) - G_j^c \le 0, \quad j = 1, \cdots, 9 \\ &\qquad \mathbf{x}_L \le \mathbf{x} \le \mathbf{x}_U, \quad \mathbf{x} \in R^9 \end{aligned}$$

Solve this optimization problem using the sequential quadratic programming (use the matlab function, ‘fmincon’, in Matlab). Make your own discussion and conclusion.

