Nonlinear programming

Unconstrained optimization techniques

Introduction

- This chapter deals with the various methods of solving the unconstrained minimization problem:

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

- It is true that rarely a practical design problem would be unconstrained; still, a study of this class of problems would be important for the following reasons:

  – The constraints do not have significant influence in certain design problems.

  – Some of the powerful and robust methods of solving constrained minimization problems require the use of unconstrained minimization techniques.

  – The unconstrained minimization methods can be used to solve certain complex engineering analysis problems. For example, the displacement response (linear or nonlinear) of any structure under any specified load condition can be found by minimizing its potential energy. Similarly, the eigenvalues and eigenvectors of any discrete system can be found by minimizing the Rayleigh quotient.

Classification of unconstrained minimization methods

Direct search methods

- Random search method
- Grid search method
- Univariate method
- Pattern search methods
  - Powell's method
  - Hooke-Jeeves method
- Rosenbrock's method
- Simplex method

Descent methods

- Steepest descent (Cauchy method)
- Fletcher-Reeves method
- Newton's method
- Marquardt method
- Quasi-Newton methods
  - Davidon-Fletcher-Powell method
  - Broyden-Fletcher-Goldfarb-Shanno method

Direct search methods

- They require only the objective function values but not the partial derivatives of the function in finding the minimum and hence are often called the nongradient methods.

- The direct search methods are also known as zeroth-order methods since they use zeroth-order derivatives of the function.

- These methods are most suitable for simple problems involving a relatively small numbers of variables.

- These methods are in general less efficient than the descent methods.

Descent methods

- The descent techniques require, in addition to the function values, the first and in some cases the second derivatives of the objective function.
- Since more information about the function being minimized is used (through the use of derivatives), descent methods are generally more efficient than direct search techniques.
- The descent methods are known as *gradient methods.*
- Among the gradient methods, those requiring only first derivatives of the function are called *first-order methods;* those requiring both first and second derivatives of the function are termed *second-order methods.*

General approach

- All unconstrained minimization methods are iterative in nature and hence they start from an initial trial solution and proceed toward the minimum point in a sequential manner.

- Different unconstrained minimization techniques differ from one another only in the method of generating the new point X_{i+1} from X_i and in testing the point X_{i+1} for optimality.

Direct search methods

Random Search Methods: Random serach methods are based on the use of random numbers in finding the minimum point. Since most of the computer libraries have random number generators, these methods can be used quite conveniently. Some of the best known random search methods are:

- Random jumping method
- Random walk method

Random jumping method

- Although the problem is an unconstrained one, we establish the bounds l_i and u_i for each design variable x_i, $i=1,2,\ldots,n$, for generating the random values of x_i:

$$l_i \le x_i \le u_{i,} \qquad i = 1,2,\ldots..,n$$

- In the random jumping method, we generate sets of n random numbers, $(r_1, r_2, \ldots.., r_n)$, that are uniformly distributed between 0 and 1. Each set of these numbers, is used to find a point, **X,** inside the hypercube defined by the above equation as

$$\mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{Bmatrix} = \begin{Bmatrix} l_1 + r_1(u_1 - l_1) \\ l_2 + r_2(u_2 - l_2) \\ \vdots \\ l_n + r_n(u_n - l_n) \end{Bmatrix}$$

and the value of the function is evaluated at this point **X**.

Random jumping method

- By generating a large number of random points $\mathbf{X}$ and evaluating the value of the objective function at each of these points, we can take the smallest value of $f(\mathbf{X})$ as the desired minimum point.

Random walk method

- The random walk method is based on generating a sequence of improved approximations to the minimum, each derived from the preceding approximation.

- Thus, if $\mathbf{X_i}$ is the approximation to the minimum obtained in the (*i*-1)*th* stage (or step or iteration), the new or improved approximation in the *i*th stage is found from the relation

$$\mathbf{X_{i+1}} = \mathbf{X_i} + \lambda \mathbf{u_i}$$

where λ is a prescribed scalar step length and $\mathbf{u}_i$ is a unit random vector generated in the *i*th stage.

Advantages of random search methods

1. These methods can work even if the objective function is discontinuous and nondifferentiable at some of the points.

1. The random methods can be used to find the global minimum when the objective function possesses several relative minima.

1. These methods are applicable when other methods fail due to local difficulties such as sharply varying functions and shallow regions.

1. Although the random methods are not very efficient by themselves, they can be used in the early stages of optimization to detect the region where the global minimum is likely to be found. Once this region is found, some of the more efficient techniques can be used to find the precise location of the global minimum point.

Univariate method

- In this method, we change only one variable at a time and seek to produce a sequence of improved approximations to the minimum point.

- By starting at a base point $\mathbf{X}_i$ in the *i*th iteration, we fix the values of *n-1* variables and vary the remaining variable. Since only one variable is changed, the problem becomes a one-dimensional minimization problem and any of the methods discussed in the previous chapter on one dimensional minimization methods can be used to produce a new base point $\mathbf{X}_{i+1}$.

- The search is now continued in a new direction. This new direction is obtained by changing any one of the *n-1* variables that were fixed in the previous iteration.

Example

Minimize

$$f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1 x_2 + x_2^2$$

With the starting point (0,0).

Solution: We will take the probe length ε as 0.01 to find the correct direction for decreasing the function value in step 3. Further, we will use the differential calculus method to find the optimum step length λ_i* along the direction ± S_i in step 4.

Example

Iteration *i=1*

Step 2: Choose the search direction $\mathbf{S_1}$ as $\mathbf{S}_1 = \begin{Bmatrix} 1 \\ 0 \end{Bmatrix}$

Step 3: To find whether the value of *f* decreases along $\mathbf{S_1}$ or **–**$\mathbf{S_1}$**,** we use the probe length **ε.** Since

$$f_1 = f(\mathbf{X_1}) = f(0,0) = 0$$

$$f^+ = f(\mathbf{X_1} + \varepsilon \mathbf{S_1}) = f(\varepsilon, 0) = 0.01 - 0 + 2(0.0001) + 0 + 0 = 0.0102 > f_1$$

$$f^- = f(\mathbf{X_1} - \varepsilon \mathbf{S_1}) = f(-\varepsilon, 0) = -0.01 - 0 + 2(0.0001) + 0 + 0 = -0.9998 < f_1$$

-$\mathbf{S_1}$ is the correct direction for minimizing *f* from $\mathbf{X_1}$**.**

Example

Step 4: To find the optimum step length λ_1*, we minimize

$$f(\mathbf{X}_1 - \lambda_1 \mathbf{S}) = f(-\lambda_1, 0)$$

$$= (-\lambda_1) - 0 + 2(-\lambda_1)^2 + 0 + 0 = 2\lambda_1^2 - \lambda_1$$

$$\text{As } \frac{\mathrm{d}f}{\mathrm{d}\lambda_1} = 0 \text{ at } \lambda_1 = \frac{1}{4}, \text{ we have } \lambda_1^* = \frac{1}{4}$$

Step 5: Set

$$\mathbf{X}_2 = \mathbf{X}_1 - \lambda_1^* \mathbf{S}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} - \frac{1}{4} \begin{Bmatrix} 1 \\ 0 \end{Bmatrix} = \begin{Bmatrix} -\frac{1}{4} \\ 0 \end{Bmatrix}$$

$$f_2 = f(\mathbf{X}_2) = f(-\frac{1}{4}, 0) = -\frac{1}{8}$$

Example

Iteration *i=2*

Step 2: Choose the search direction $\mathbf{S_2}$ as $\mathbf{S}_2 = \begin{Bmatrix} 0 \\ 1 \end{Bmatrix}$

Step 3: Since

$$f_2 = f(\mathbf{X_2}) = -0.125$$
$$f^+ = f(\mathbf{X_2} + \varepsilon\mathbf{S_2}) = f(-0.25, 0.01) = -0.1399 < f_2$$
$$f^- = f(\mathbf{X_2} - \varepsilon\mathbf{S_2}) = f(-0.25, -0.01) = -0.1099 > f_2$$

$\mathbf{S_2}$ is the correct direction for decreasing the value of f from $\mathbf{X_2}$**.**

Example

Step 4: We minimize $f\,(\mathbf{X_2} + \boldsymbol{\lambda}_\mathbf{2}\mathbf{S_2})$ to find λ_2*.

Here

$$f\,(\mathbf{X}_2+\lambda_\mathbf{2}\mathbf{S_2}) = f\,(-0.25, \lambda_2)$$
$$= -0.25 - \lambda_2 + 2(0.25)^2 - 2(0.25)(\lambda_2) + \lambda_2^2$$
$$= \lambda_2^2 - 1.5\lambda_2 - 0.125$$
$$\frac{\mathrm{d}f}{\mathrm{d}\lambda_2} = 2\lambda_2 - 1.5 = 0 \text{ at } \lambda_2^* = 0.75$$

Step 5: Set

$$\mathbf{X}_3 = \mathbf{X}_2 + \lambda_2^*\mathbf{S_2} = \begin{Bmatrix} -0.25 \\ 0 \end{Bmatrix} + 0.75\begin{Bmatrix} 0 \\ 1 \end{Bmatrix} = \begin{Bmatrix} -0.25 \\ 0.75 \end{Bmatrix}$$

$$f_3 = f\,(\mathbf{X}_3) = -0.6875$$

Next, we set the iteration number as $i = 3$, and continue the procedure until the optimum

solution $\mathbf{X^*} = \begin{Bmatrix} -1.0 \\ 1.5 \end{Bmatrix}$ with $f\,(\mathbf{X^*}) = -1.25$ is found.

Pattern Directions

- In the univariate method, we search for the minimum along the directions parallel to the coordinate axes. We noticed that this method may not converge in some cases, and that even if it converges, its convergence will be very slow as we approach the optimum point.

- These problems can be avoided by changing the directions of search in a favorable manner instead of retaining them always parallel to the coordinate axes.

Pattern Directions

Consider the contours of the function shown in the figure. Let the points 1,2,3,... indicate the successive points found by the univariate method. It can be noticed that the lines joining the alternate points of the search (e.g.,1,3;2,4;3,5;4,6;...) lie in the general direction of the minimum and are known as pattern directions. It can be proved that if the objective function is a quadratic in two variables, all such lines pass through the minimum. Unfortunately, this property will not be valid for multivariable functions even when they are quadratics. However, this idea can still be used to achieve rapid convergence while finding the minimum of an *n*-variable function.

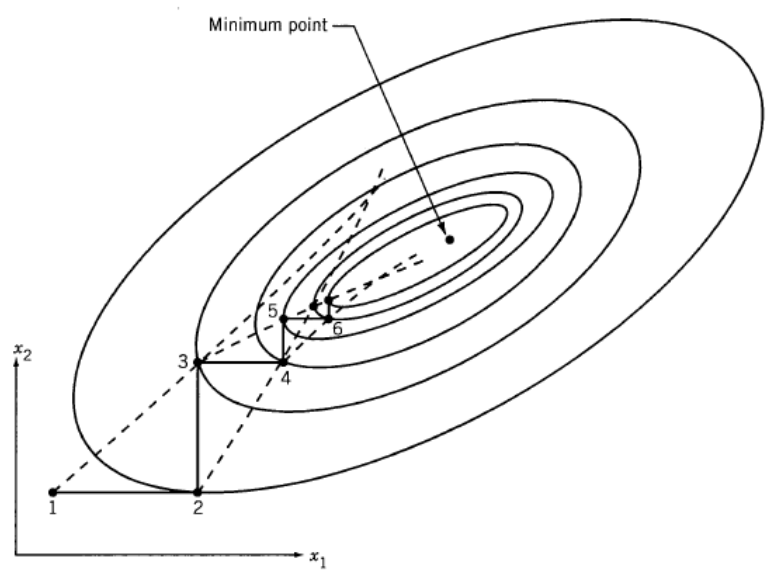

Figure 6.7 Lines defined by the alternate points lie in the general direction of the minimum.

Pattern Directions

- Methods that use pattern directions as search directions are known as **pattern search methods**.
- Two of the best known pattern search methods are:
  - Hooke-Jeeves method
  - Powell's method
- In general, a pattern search method takes *n* univariate steps, where *n* denotes the number of design variables and then searches for the minimum along the pattern direction $\mathbf{S}_i$, defined by

$$\mathbf{S}_i = \mathbf{X}_i - \mathbf{X}_{i-n}$$

where $\mathbf{X}_i$ is the point obtained at the end of *n* univariate steps.

- In general, the directions used prior to taking a move along a pattern direction need not be univariate directions.

Hooke and Jeeves’ Method

- The pattern search method of Hooke and Jeeves is a sequential technique each step of which consists of two kinds of moves, the exploratory move and the pattern move.

- The first kind of move is included to explore the local behaviour of the objective function and the second kind of move is included to take advantage of the pattern direction.

Example

Minimize

$$f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$$

starting from the point

$$\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

Take $\Delta x_1 = \Delta x_2 = 0.8$ and $\varepsilon = 0.1$.

Solution:

Step 1: We take the starting base point as

$$\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

and step lengths as $\Delta x_1 = \Delta x_2 = 0.8$ along the coordinate directions $\mathbf{u_1}$ and $\mathbf{u_2}$, respectively. Set $k=1$.

Example

Step 2: $f_1 = f(\mathbf{X_1}) = 0$, $i=1$, and

$$\mathbf{Y}_{10} = \mathbf{X}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

Step 3: To find the new temporary base point, we set $i=1$ and evaluate $f = f(\mathbf{Y}_{10}) = 0.0$

$$f^+ = f(\mathbf{Y_{10}} + \Delta x_1 \mathbf{u_1}) = f(0.8, 0.0) = 2.08$$
$$f^- = f(\mathbf{Y_{10}} - \Delta x_1 \mathbf{u_1}) = f(-0.8, 0.0) = 0.48$$

Since $f < \min(f^+, f^-)$, we take $\mathbf{Y_{11}} = \mathbf{X_1}$. Next we set $i=2$, and evaluate

$f = f(\mathbf{Y_{11}}) = 0.0$ and

$$f^+ = f(\mathbf{Y_{11}} + \Delta x_2 \mathbf{u_2}) = f(0.0, 0.8) = -0.16$$

Since $f^+ < f$, we set

$$\mathbf{Y_{12}} = \begin{Bmatrix} 0.0 \\ 0.8 \end{Bmatrix}$$

*** $\mathbf{Y}_{kj}$ indicates the temporary base point $\mathbf{X}_k$ by perturbing the *j*th component of $\mathbf{X}_\mathrm{k}$

Example

Step 4: As $\mathbf{Y}_{12}$ is different from $\mathbf{X_1}$, the new base point is taken as:

$$\mathbf{X}_2 = \begin{Bmatrix} 0.0 \\ 0.8 \end{Bmatrix}$$

Step 5: A pattern direction is established as:

$$\mathbf{S} = \mathbf{X_2} - \mathbf{X_1} = \begin{Bmatrix} 0.0 \\ 0.8 \end{Bmatrix} - \begin{Bmatrix} 0.0 \\ 0.0 \end{Bmatrix} = \begin{Bmatrix} 0.0 \\ 0.8 \end{Bmatrix}$$

The optimal step length λ* is found by minimizing

$$f(\mathbf{X}_2 + \lambda \mathbf{S}) = f(0.0, 0.8 + 0.8\lambda) = 0.64\lambda^2 + 0.48\lambda - 0.16$$

As $df / d\lambda$ = 1.28 λ+0.48 = 0 at λ* = - 0.375, we obtain the point $\mathbf{Y}_{20}$ as

$$\mathbf{Y_{20}} = \mathbf{X_2} + \lambda\,\mathbf{*S} = \begin{Bmatrix} 0.0 \\ 0.8 \end{Bmatrix} - 0.375 \begin{Bmatrix} 0.0 \\ 0.8 \end{Bmatrix} = \begin{Bmatrix} 0.0 \\ 0.5 \end{Bmatrix}$$

Example

Step 6: Set $k = 2$, $f = f_2 = f(\mathbf{Y}_{20}) = -0.25$, and repeat step 3. Thus, with $i=1$, we evaluate

$$f^{+} = f(\mathbf{Y_{20}} + \Delta x_1 \mathbf{u_1}) = f(0.8, 0.5) = 2.63$$

$$f^{-} = f(\mathbf{Y_{20}} - \Delta x_1 \mathbf{u_1}) = f(-0.8, 0.5) = -0.57$$

Since $f^{-} < f < f^{+}$, we take $\mathbf{Y_{21}} = \begin{Bmatrix} -0.8 \\ 0.5 \end{Bmatrix}$

Next, we set $i=2$ and evaluate $f = f(\mathbf{Y}_{21}) = -0.57$ and

$$f^{+} = f(\mathbf{Y_{21}} + \Delta x_2 \mathbf{u_2}) = f(-0.8, 1.3) = -1.21$$

As $f^{+} < f$, we take $\mathbf{Y_{22}} = \begin{Bmatrix} -0.8 \\ 1.3 \end{Bmatrix}$. Since $f(\mathbf{Y}_{22}) = -1.21 < f(\mathbf{X}_2) = -0.25$, we take the new base point as:

$$\mathbf{X_3} = \mathbf{Y_{22}} = \begin{Bmatrix} -0.8 \\ 1.3 \end{Bmatrix}$$

Example

Step 6 continued: After selection of the new base point, we go to step 5.

This procedure has to be continued until the optimum point

$$\mathbf{X}_{\mathbf{opt}} = \begin{Bmatrix} -1.0 \\ 1.5 \end{Bmatrix}$$

is found.

Indirect search (descent method)

Gradient of a function

The gradient of a function is an *n*-component vector given by:

$$\underset{n\times 1}{\nabla f} = \begin{Bmatrix} \dfrac{\partial f}{\partial x_1} \\ \dfrac{\partial f}{\partial x_2} \\ \vdots \\ \dfrac{\partial f}{\partial x_n} \end{Bmatrix}$$

The gradient has a very important property. If we move along the gradient direction from any point in *n* dimensional space, the function value increases at the fastest rate. Hence the gradient direction is called the direction of the steepest ascent. Unfortunately, the direction of steepest ascent is a local property not a global one.

Indirect search (descent method)

- The gradient vectors ∇f evaluated at points 1,2,3 and 4 lie along the directions 11', 22', 33',44', respectively.
- Thus the function value increases at the fastest rate in the direction 11' at point 1, but not at point 2. Similarly, the function value increases at the fastest rate in direction 22' at point 2, but not at point 3.
- In other words, the direction of steepest ascent generally varies from point to point, and if we make infinitely small moves along the direction of steepest ascent, the path will be a curved line like the curve 1-2-3-4 in the

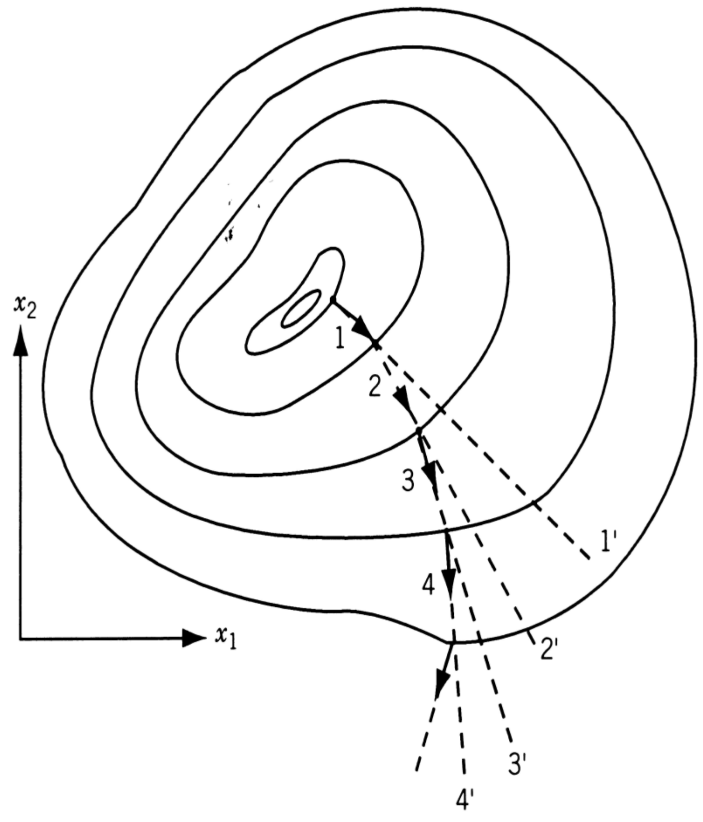

Figure 6.15 Steepest ascent directions.

Indirect search (descent method)

- Since the gradient vector represents the direction of steepest ascent, the negative of the gradient vector denotes the direction of the steepest descent.
- Thus, any method that makes use of the gradient vector can be expected to give the minimum point faster than one that does not make use of the gradient vector.
- All the descent methods make use of the gradient vector, either directly or indirectly, in finding the search directions.

Steepest descent (Cauchy method)

- The use of the negative of the gradient vector as a direction for minimization was first made by Cauchy in 1847.

- In this method, we start from an initial trial point $\mathbf{X}_1$ and iteratively move along the steepest descent directions until the optimum point is found.

Example

Minimize

$$f(x_1, x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$$

Starting from the point $\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$

Solution:

Iteration 1: The gradient of *f* is given by:

$$\nabla f = \begin{Bmatrix} \partial f / \partial x_1 \\ \partial f / \partial x_2 \end{Bmatrix} = \begin{Bmatrix} 1 + 4x_1 + 2x_2 \\ -1 + 2x_1 + 2x_2 \end{Bmatrix}$$

$$\nabla f_1 = \nabla f(\mathbf{X}_1) = \begin{Bmatrix} 1 \\ -1 \end{Bmatrix}$$

Therefore

$$\mathbf{S}_1 = -\nabla f_1 = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix}$$

Example

- To find $\mathbf{X}_2$, we need to find the optimal step length λ_1*. For this, we minimize

$$f(\mathbf{X}_1 + \lambda_1^* \mathbf{S}) = f(-\lambda_1, \lambda_1) = \lambda_1^2 - 2\lambda_1 \text{ with respect to } \lambda_1. \text{Since } \frac{df}{d\lambda_1} = 0 \text{ at } \lambda_1^* = 1,$$

we obtain:

$$\mathbf{X}_2 = \mathbf{X}_1 + \lambda_1^* \mathbf{S}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} + 1 \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix}$$

- As

$$\nabla f_2 = \nabla f(\mathbf{X}_2) = \begin{Bmatrix} -1 \\ -1 \end{Bmatrix} \neq \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}, \mathbf{X}_2 \text{ is not optimum.}$$

Example

Iteration 2:

$$\mathbf{S}_2 = -\nabla f_2 = \begin{Bmatrix} 1 \\ 1 \end{Bmatrix}$$

To minimize

$$f(\mathbf{X}_2 + \lambda_2 \mathbf{S}_2) = f(-1+\lambda_2, 1+\lambda_2) = 5\lambda_2^2 - 2\lambda_2 - 1$$

we set $\dfrac{df}{d\lambda_2} = 0$. This gives $\lambda_2^* = 1/5$, and hence

$$\mathbf{X}_3 = \mathbf{X}_2 + \lambda_2^* \mathbf{S}_2 = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} + \frac{1}{5}\begin{Bmatrix} 1 \\ 1 \end{Bmatrix} = \begin{Bmatrix} -0.8 \\ 1.2 \end{Bmatrix}$$

Since the components of the gradient at **X₃**, $\nabla f_3 = \begin{Bmatrix} 0.2 \\ -0.2 \end{Bmatrix}$ are not zero, we proceed to the next iteration.

Example

Iteration 3:

$$\mathbf{S}_3 = -\nabla f_3 = \begin{Bmatrix} -0.2 \\ 0.2 \end{Bmatrix}$$

As

$$f(\mathbf{X}_3 + \lambda_3 \mathbf{S}_3) = f(-0.8 - 0.2\lambda_3, 1.2 + 0.2\lambda_3) = 0.04\lambda_3^2 - 0.08\lambda_3 - 1.20$$

we set $\dfrac{df}{d\lambda_3} = 0$. This gives $\lambda_3^* = 1.0$. Therefore,

$$\mathbf{X}_4 = \mathbf{X}_3 + \lambda_3^* \mathbf{S}_3 = \begin{Bmatrix} -0.8 \\ 1.2 \end{Bmatrix} + 1.0 \begin{Bmatrix} -0.2 \\ 0.2 \end{Bmatrix} = \begin{Bmatrix} -1.0 \\ 1.4 \end{Bmatrix}$$

The gradient at $\mathbf{X_4}$ is given by: $\nabla f_4 = \begin{Bmatrix} -0.2 \\ -0.2 \end{Bmatrix}$

Since the components of the gradient at $\mathbf{X_4}$ are not equal to zero, $\mathbf{X_4}$ is not optimum and hence we have to proceed to the next iteration. This process has to be continued until the optimum point, $\mathbf{X^*} = \begin{Bmatrix} -1.0 \\ 1.5 \end{Bmatrix}$ is found.

Convergence Criteria

The following criteria can be used to terminate the iterative process:

1. When the change in function value in two consecutive iterations is small:

$$\left| \frac{f(\mathbf{X_{i+1}}) - f(\mathbf{X_i})}{f(\mathbf{X_i})} \right| \le \varepsilon_1$$

1. When the partial derivatives (components of the gradient) of *f* are small:

$$\left| \frac{\partial f}{\partial x_i} \right| \le \varepsilon_2, \quad i = 1,2,\ldots,n$$

1. When the change in the design vector in two consecutive iterations is small:

$$\left| \mathbf{X}_{i+1} - \mathbf{X}_i \right| \le \varepsilon_3$$

Conjugate Gradient (Fletcher-Reeves) Method

- The convergence characteristics of the steepest descent method can be improved greatly by modifying it into a conjugate gradient method which can be considered as a conjugate directions method involving the use of the gradient of the function.

- We saw that any minimization method that makes use of the conjugate directions is quadratically convergent. This property of quadratic convergence is very useful because it ensures that the method will minimize a quadratic function in *n* steps or less.

- Since any general function can be approximated reasonably well by a quadratic near the optimum point, any quadratically convergent method is expected to find the optimum point in a finite number of iterations.

Example

Minimize

$$f(x_1,x_2) = x_1 - x_2 + 2x_1^2 + 2x_1x_2 + x_2^2$$

starting from the point $\mathbf{X}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$

Solution:

Iteration 1

$$\nabla f = \begin{Bmatrix} \partial f/\partial x_1 \\ \partial f/\partial x_2 \end{Bmatrix} = \begin{Bmatrix} 1 + 4x_1 + 2x_2 \\ -1 + 2x_1 + 2x_2 \end{Bmatrix}$$

$$\nabla f_1 = \nabla f(\mathbf{X}_1) = \begin{Bmatrix} 1 \\ -1 \end{Bmatrix}$$

The search direction is taken as:

$$\mathbf{S}_1 = -\nabla f_1 = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix}.$$

Example

- To find the optimal step length λ_1* along $\mathbf{S_1}$, we minimize $f(\mathbf{X}_1 + \lambda_1 \mathbf{S}_1)$ with respect to $\lambda_{1.}$ Here

$$f(\mathbf{X}_1 + \lambda_1 \mathbf{S}_1) = f(-\lambda_1, +\lambda_1) = \lambda_1^2 - 2\lambda_1$$

$$\frac{df}{d\lambda_1} = 0 \quad \text{at} \quad \lambda_1^* = 1$$

- Therefore

$$\mathbf{X}_2 = \mathbf{X}_1 + \lambda_1^* \mathbf{S}_1 = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} + 1 \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix}$$

Example

Iteration 2: Since

$$\nabla f_2 = \nabla f(\mathbf{X}_2) = \begin{Bmatrix} -1 \\ -1 \end{Bmatrix},$$

the equation

$$\mathbf{S}_i = -\nabla f_i + \frac{|\nabla f_i|^2}{|\nabla f_{i-1}|^2} \mathbf{S}_{i-1}$$

gives the next search direction as

$$\mathbf{S}_2 = -\nabla f_2 + \frac{|\nabla f_2|^2}{|\nabla f_1|^2} \mathbf{S}_1$$

where

$$|\nabla f_1|^2 = 2 \quad \text{and} \quad |\nabla f_2|^2 = 2$$

Therefore

$$\mathbf{S}_2 = -\begin{Bmatrix} -1 \\ -1 \end{Bmatrix} + \left(\frac{2}{2}\right) \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} = \begin{Bmatrix} 0 \\ +2 \end{Bmatrix}$$

Example

- To find λ_2^*, we minimize

$$f(\mathbf{X}_2 + \lambda_2 \mathbf{S}_2) = f(-1, 1 + 2\lambda_2)$$
$$= -1 - (1 + 2\lambda_2) + 2 - 2(1 + 2\lambda_2) + (1 + 2\lambda_2)^2$$
$$= 4\lambda_2^2 - 2\lambda_2 - 1$$

with respect to λ_2. As $df/d\lambda_2 = 8\lambda_2 - 2 = 0$ at $\lambda_2^* = 1/4$, we obtain:

$$\mathbf{X}_3 = \mathbf{X}_2 + \lambda_2^* \mathbf{S}_2 = \begin{Bmatrix} -1 \\ 1 \end{Bmatrix} + \frac{1}{4}\begin{Bmatrix} 0 \\ 2 \end{Bmatrix} = \begin{Bmatrix} -1 \\ 1.5 \end{Bmatrix}$$

Thus the optimum point is reached in two iterations. Even if we do not know this point to be optimum, we will not be able to move from this point in the next iteration. This can be verified as follows:

Example

Iteration 3:

Now

$$\nabla f_3 = \nabla f(\mathbf{X}_3) = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}, \quad |\nabla f_2|^2 = 2, \quad \text{and} \quad |\nabla f_3|^2 = 0.$$

Thus,

$$\mathbf{S}_3 = -\nabla f_3 + (|\nabla f_3|^2/|\nabla f_2|^2)\mathbf{S}_2 = -\begin{Bmatrix} 0 \\ 0 \end{Bmatrix} + \left(\frac{0}{2}\right)\begin{Bmatrix} 0 \\ 0 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix}$$

This shows that there is no search direction to reduce *f* further, and hence $\mathbf{X}_3$ is optimum.