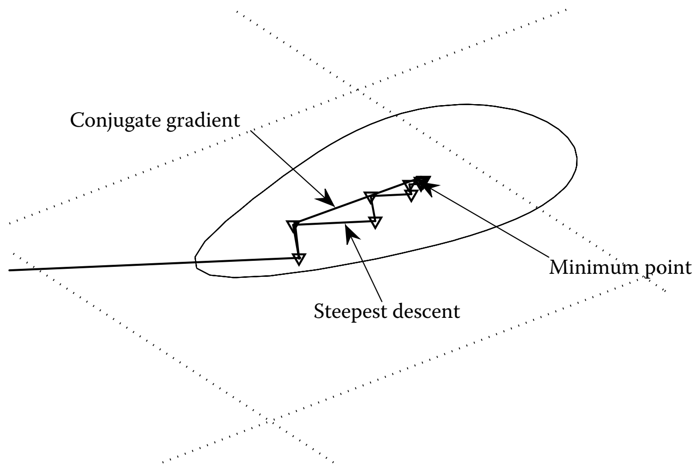

FIGURE 3.8
Convergence plot of conjugate gradient/steepest descent method.

is compared with the first-order, steepest descent method. The conjugate method does not show sluggishness in reaching the minimum point.

Initial function value = 1452.2619

No.	x-vector		f(x)	Deriv.
1	0.095	0.023	-2.704	1006.074
2	0.178	0.145	-5.404	37.036
3	0.507	0.136	-9.627	23.958
4	0.510	0.123	-9.655	4.239
5	0.505	0.121	-9.656	0.605
6	0.504	0.122	-9.656	0.340
7	0.504	0.122	-9.656	0.023

3.4.6 DFP Method

In the DFP method, the inverse of the Hessian is approximated by a matrix [A] and the search direction is given by

$$\boldsymbol{S}_i = -[\boldsymbol{A}]\nabla f(\boldsymbol{x}_i) \tag{3.24}$$

The information stored in the matrix [A] is called as the metric and because it changes with every iteration, the DFP method is known as the variable metric method. Because this method uses first-order derivatives and has the property of quadratic convergence, it is referred to as a *quasi-Newton* method. The inverse of the Hessian matrix can be approximated as

$$[\boldsymbol{A}]_{i+1} = [\boldsymbol{A}]_i + \frac{\Delta\boldsymbol{x}\Delta\boldsymbol{x}^T}{\Delta\boldsymbol{x}^T\nabla g} - \frac{[\boldsymbol{A}]_i\nabla g\nabla g^T[\boldsymbol{A}]_i}{\nabla g^T[\boldsymbol{A}]_i\nabla g} \tag{3.25}$$

where

$$\Delta\boldsymbol{x} = \Delta\boldsymbol{x}_i - \Delta\boldsymbol{x}_{i-1} \tag{3.26}$$

$$\nabla g = \nabla g_i - \nabla g_{i-1} \tag{3.27}$$

The matrix [$\boldsymbol{A}$] is initialized to the identity matrix. The algorithm for the DFP method is described in Table 3.7 and a MATLAB code of its implementation is given in *dfp.m*.

On executing the code with a starting value of $\boldsymbol{x}$ as (–3, 2) the following output is displayed in the command window for the test problem. Observe that in the second and the third iterations, search points are similar in this method and the conjugate gradient method, indicating that search directions were similar. In further iterations, however, the search direction is different. Further, on typing *inv*(*A*) in the MATLAB command prompt and then

TABLE 3.7

Algorithm for the DFP Method

Step 1:	Given $\boldsymbol{x}_i$ (starting value of design variable)
	ε_1 (tolerance of function value from previous iteration)
	ε_2 (tolerance on gradient value)
	$\Delta\boldsymbol{x}$ (required for gradient computation)
	[$\boldsymbol{A}$] (initialize to identity matrix)
Step 2:	Compute $f(\boldsymbol{x}_i)$ and $\nabla f(\boldsymbol{x}_i)$ (function and gradient vector)
	$\boldsymbol{S}_i = -\nabla f(\boldsymbol{x}_i)$ (search direction)
	$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + \alpha\boldsymbol{S}_i$ (update the design vector)
	Minimize $f(\boldsymbol{x}_{i+1})$ and determine α (use the golden section method)
Step 3:	Compute $\Delta\boldsymbol{x}$ and ∇g
	$[\boldsymbol{A}]_{i+1} = [\boldsymbol{A}]_i + \frac{\Delta\boldsymbol{x}\Delta\boldsymbol{x}^T}{\Delta\boldsymbol{x}^T\nabla g} - \frac{[\boldsymbol{A}]_i\nabla g\nabla g^T[\boldsymbol{A}]_i}{\nabla g^T[\boldsymbol{A}]_i\nabla g}$
	$\boldsymbol{S}_{i+1} = -[\boldsymbol{A}]_{i+1}\nabla f(\boldsymbol{x}_{i+1})$
	$\boldsymbol{x}_{i+2} = \boldsymbol{x}_{i+1} + \alpha\boldsymbol{S}_{i+1}$
	Minimize $f(\boldsymbol{x}_{i+2})$ and determine α (use the golden section method)
	If $\lvert f(\boldsymbol{x}_{i+2}) - f(\boldsymbol{x}_{i+1})\rvert > \varepsilon_1$ or $\lVert\nabla f(\boldsymbol{x}_{i+1})\rVert > \varepsilon_2$
	then goto Step 3
	else goto Step 4
Step 4:	Converged. Print $\boldsymbol{x}^* = \boldsymbol{x}_{i+2}$, $f(\boldsymbol{x}^*) = f(\boldsymbol{x}_{i+2})$

printing the Hessian matrix at the converged value of $\boldsymbol{x}$, it is observed that $[A]$ approaches $[\boldsymbol{H}]^{-1}$.

```
      Initial function value = 1452.2619
No.       x-vector          f(x)         Deriv.
-------------------------------------------------
1     0.095       0.023    -2.704       1006.074
2     0.179       0.145    -5.418         37.036
3     0.508       0.145    -9.576         23.983
4     0.501       0.122    -9.656          7.004
5     0.504       0.122    -9.656          0.396
6     0.504       0.122    -9.656          0.053
7     0.504       0.122    -9.656          0.038
8     0.504       0.122    -9.656          0.028
9     0.504       0.122    -9.656          0.005
-------------------------------------------------

>> A
A =
    0.0091    0.0005
    0.0005    0.0033
>> inv(hessian(x,delx,n_of_var))
ans =
    0.0091    0.0005
    0.0005    0.0033
```

3.4.7 BFGS Method

In the BFGS method, the Hessian is approximated using the variable metric matrix $[A]$ given by the equation

$$[\boldsymbol{A}]_{i+1} = [\boldsymbol{A}]_i + \frac{g\nabla g^T}{\nabla g^T \Delta \boldsymbol{x}} + \frac{\nabla f(\boldsymbol{x}_i)\nabla f(\boldsymbol{x}_i)^T}{\nabla f(\boldsymbol{x}_i)^T \boldsymbol{S}_i} \tag{3.28}$$

It is important to note that whereas the matrix $[\boldsymbol{A}]$ converges to the inverse of the Hessian in the DFP method, the matrix $[\boldsymbol{A}]$ converges to the Hessian itself in the BFGS method. As the BFGS method needs fewer restarts as compared to the DFP method, it is more popular than the DFP method. The algorithm for the BFGS method is described in Table 3.8 and a MATLAB code of its implementation is given in *BFGS.m*.

On executing the code with a starting value of $\boldsymbol{x}$ as (–3, 2) the following output is displayed in the command window for the test problem. Again, it is observed that in the second and third iterations, search points are similar to this method as compared to DFP and the conjugate gradient methods, indicating that search directions were similar. Further, on typing A in the