Quasi-Newton Methods

Werner C. Rheinboldt

These are excerpts of material relating to the books [OR00] and [Rhe98] and of write-ups prepared for courses held at the University of Pittsburgh. Some further references are [Kel95], [Kel99], [DS98].

1 Broyden's Method

Let

$$F(x) = 0, \quad F : \mathbb{R}^n \longrightarrow \mathbb{R}^n, \tag{1}$$

be a given system of nonlinear equations defined by a sufficiently smooth function F. A linearization method for the numerical solution of (1) has the general form

$$x^{k+1} = x^k - B_k^{-1} F(x^k), \quad k = 0, 1, \dots \tag{2}$$

where the $n \times n$ matrices B_k are suitably chosen.

The integral mean value theorem for F states that

$$\left[\int_0^1 DF(x + t(y - x))dt \right] (y - x) = F(y) - F(x), \qquad x, y \in \mathbb{R}^n,$$

The matrix in brackets can be interpreted as the average of the Jacobian matrix on the line segment between the points x and y. This suggests requiring the matrices B_k to satisfy the so called quasi-Newton condition

$$B_{k+1}(x^{k+1} - x^k) = F(x^{k+1}) - F(x^k).$$

In several papers around 1967, C. G. Broyden suggested that it is numerically advantageous to choose the matrices in (2) such that $\mathrm{rank}(B_{k+1} - B_k)$ is small. This led to the development of the so-called *quasi-Newton methods*, which can be characterized by the following three properties:

$$\left.\begin{array}{ll} (a) & B_k(x^{k+1} - x^k) + F(x^k) = 0, \\ (b) & B_{k+1}(x^{k+1} - x^k) = F(x^{k+1}) - F(x^k) \\ (c) & B_{k+1} = B_k + \Delta B_k, \quad \mathrm{rank}\,\Delta B_k = m \geq 1 \end{array}\right\} \quad k = 0, 1, \dots. \tag{3}$$

Up to now only the values $m = 1$ or $m = 2$ have been used in the design of quasi-Newton methods. From (3) we obtain some frequently used relations

$$\begin{array}{ll} (a) & (B_{k+1} - B_k)s^k = F(x^{k+1}), \quad s^k = x^{k+1} - x^k), \\ (b) & F(x^{k+1}) = y^k - B_k s^k, \quad y^k = F(x^{k+1}) - F(x^k). \end{array} \tag{4}$$

C. G. Broyden himself developed two quasi-Newton methods with $m = 1$ and called one of them his *good* method. This terminology has persisted. The good method uses

$$B_{k+1} := B_k + \frac{F(x^{k+1})(s^k)^\top}{(s^k)^\top s^k}, \tag{5}$$

or, in view of (4)(b),

$$B_{k+1} := B_k + \frac{(y^k - B_k s^k)(s^k)^\top}{(s^k)^\top s^k}. \tag{6}$$

As for all standard linearization methods, the matrices B_k should be invertible. Recall the well–known Sherman-Morrison formula:

1.1. *For $u, v \in \mathbb{R}^n$ the matrix $I + uv^\top$ is invertible if and only if $1 + u^\top v \neq 0$, and in that case*

$$\left(I + uv^\top\right)^{-1} = I - \frac{1}{1 + u^\top v} uv^\top$$

If in the Broyden method the matrix B_k is nonsingular, then **1.1** shows that

$$B_{k+1} = B_k \left[I + \frac{(B_k^{-1} F(x^{k+1}))(s^k)^\top}{\|s^k\|_2^2} \right] \tag{7}$$

is again nonsingular, provided that

$$\|(s^k\|_2^2 + (B_k^{-1} F(x^{k+1}))^\top s^k \neq 0,$$

in which case, the inverse is

$$B_{k+1}^{-1} = \left[I - \frac{(B_k^{-1} F(x^{k+1}))((s^k)^\top}{\|s^k\|_2^2 + (B_k^{-1} F(x^{k+1}))^\top s^k} \right] B_k^{-1}. \tag{8}$$

With $H_k = B_k^{-1}$ and $H_{k+1} = B_{k+1}^{-1}$ this can be written in the form

$$H_{k+1} = H_k + \frac{(s^k - H_k y^k)(s^k)^\top}{(s^k)^\top H_k y^k} H_k. \tag{9}$$

Various convergence results for Broyden's method have been proved. We refer to the cited references and cite only a simplified version of such a result:

1.2. *Let $F : \Omega \subset \mathbb{R}^n \longrightarrow \mathbb{R}^n$ be continuously differentiable on an open set Ω. Suppose that $x^* \in \Omega$ a solution of $F(x) = 0$ where $DF(x^*)$ is invertible and*

$$\|DF(x) - DF(x^*)\| \leq \gamma \|x - x^*\|, \qquad \forall x \in \Omega.$$

Then there exist $\delta, \eta > 0$ such that for $\|x^0 - x^*\| < \delta$ and $\|B_0 - DF(x^*)\| < \eta$ Broyden's method converges to x^* and the rate of convergence is superlinear in the sense that

$$\lim_{k \to \infty} \frac{\|x^{k+2} - x^{k+1}\|}{\|x^{k+1} - x^k\|} = 0. \tag{10}$$

2 Recursive Implementation

With the notation

$$w^k = B_k^{-1} F(x^{k+1}), \tag{11}$$

the inverse formula (8) is

$$B_{k+1}^{-1} = \left[I - \frac{w^k (s^k)^\top}{\|s^k\|_2^2 + (w^k)^\top s^k} \right] B_k^{-1}, \tag{12}$$

and the next step equals

$$\begin{aligned} s^{k+1} = -B_{k+1}^{-1} F(x^{k+1}) &= -\left[I - \frac{w^k (s^k)^\top}{\|s^k\|_2^2 + (w^k)^\top s^k} \right] w^k \\ &= -\frac{\|s^k\|_2^2}{\|s^k\|_2^2 + (w^k)^\top s^k} w^k. \end{aligned} \tag{13}$$

From (13) it follows that

$$\frac{s^{k+1}(s^k)^\top}{\|s^k\|_2^2} = -\frac{w^k (s^k)^\top}{\|s^k\|_2^2 + (w^k)^\top s^k}$$

whence (12) becomes

$$B_{k+1}^{-1} = \left[I + \frac{s^{k+1}(s^k)^\top}{\|s^k\|_2^2} \right] B_k^{-1} = \prod_{j=0}^{k} \left[I + \frac{s^{j+1}(s^j)^\top}{\|s^j\|_2^2} \right] B_0^{-1}. \tag{14}$$

while (13) can be written as

$$\Delta x^{k+1} = -\left[I + \frac{s^{k+1}(s^k)^\top}{\|s^k\|_2^2} \right] w^k, \tag{15}$$

that is,

$$\left[1 + \frac{(s^k)^\top w^k}{\|s^k\|_2^2} \right] s^{k+1} = -w^k. \tag{16}$$

Suppose now that the steps s^j, $j = 0, 1, \ldots, k$ and their norms have been stored. Then (14) and (16) imply that

$$\begin{aligned} w^k &= \prod_{j=0}^{k-1} \left[I + \frac{s^{j+1}(s^j)^\top}{\|s^j\|_2^2} \right] w, \quad w := B_0^{-1} F(x^{k+1}) \\ s^{k+1} &= -\frac{1}{1+\tau_k} w^k, \quad \tau_k = \frac{(s^k)^\top w^k}{\|s^k\|_2^2}, \end{aligned} \tag{17}$$

which can be evaluated by the recursive algorithm

$$\begin{aligned} &w := B_0^{-1} F(x^{k+1}); \\ &\textbf{for} \quad j = 0, \ldots, k-1 \\ &\quad \tau := [(s^j)^\top w] / \|s^j\|_2^2; \\ &\quad w := w + \tau s^{j+1}; \\ &\textbf{endfor} \\ &\tau := [(s^k)^\top w] / \|s^k\|_2^2; \\ &s^{k+1} := -[1/(1+\tau)] w; \end{aligned} \tag{18}$$

In order to complete this algorithm, we need some divergence and convergence criteria. In the convergence proof a controlling quantity is the quotient

$$\Theta_k := \frac{\|B_k^{-1}F(x^{k+1})\|_2}{\|s^k\|_2}, \quad \forall k \geq 0, \tag{19}$$

and it turns out, that we should declare divergence if the condition

$$\Theta_k < \frac{1}{2} \tag{20}$$

is violated. In view of the superlinear convergence it suffices to declare convergence as soon as $\|s^{k+1}\|_2 \leq tol$.

Altogether the Broyden algorithm can now be formulated as follows, where in contrast to (18) we work with $v = -w$:

input: $x^0, B_0, k_{\max}, tol$;
  solve $B_0 s^0 = -F(x^0)$;
  $\xi_0 := \|s^0\|_2$;
  store ξ_0, s^0;
  for $k = 0, 1, \ldots, k_{\max}$
    $x^{k+1} := x^k + s^k$;
    solve $B_0 v = -F(x^{k+1})$;
    if $k > 0$
      for $j = 1, \ldots, k$
        $\tau := [(s^{j-1})^\top v]/\xi_{j-1}^2$;
        $v := v + \tau s^j$;
      endfor
    endif
    $\tau := [(s^k)^\top v]/\xi_k^2$;
    $\Theta_k := \|v\|_2/\xi_k$;
    if $\Theta_k \geq 1/2$ **then** return $\{divergence\}$;
    $s^{k+1} := v/(1-\tau)$;
    $\xi_{k+1} := \|s^{k+1}\|_2$;
    store ξ_{k+1}, s^{k+1};
    if $\xi_{k+1} \leq tol$ **then** return $\{x^* := x^{k+1} + s^{k+1}\}$;
  endfor
  return $\{maximal\ number\ of\ steps\}$

An implementation of this algorithm is the FORTRAN program NLEQ1 of P. Deuflhard, U. Nowak, and L. Weimann available in the ZIB-Elib library. There exists also a Matlab version. A somewhat different Matlab program is brsol.m by C. T. Kelley [Kel03].

The recursive form of the Broyden method has shown itself to be very economical in practice. But it has been observed occasionally, that the condition of the matrices may deteriorate over several steps causing the method to become instable. For any matrix

$$A = I + \frac{uv^\top}{\|v\|_2^2}, \quad u, v \in \mathbb{R}^n, \quad \kappa = \frac{\|u\|_2}{\|v\|_2} < 1,$$

we have $\|uv^\top\|_2 \le \|u\|_2\|v\|_2$ and therefore

$$1-\kappa \le 1 - \frac{\|uv^\top\|_2}{\|v\|_2^2} \le \|A\|_2 \le 1 + \frac{\|uv^\top\|_2}{\|v\|_2^2} \le 1+\kappa,$$

This shows that $\|A\|^{-1} \le (1-\kappa)^{-1}$ and

$$\mathrm{cond}_2(A) := \|A\|_2\|A^{-1}\|_2 \le \frac{1+\kappa}{1-\kappa},$$

Hence, for the Broyden matrices (7) it follows from the convergence condition (19), (20), that

$$\mathrm{cond}_2(B_{k+1}) \le \frac{1+\Theta_k}{1-\Theta_k}\mathrm{cond}_2(B_k) < 3\mathrm{cond}_2(B_k),$$

and hence that the growth of the condition numbers is not unduly fast and can be controlled by means of these estimates.

3 Linear Equations

The recursive form of the Broyden method also provides a very useful iterative method for linear problems

$$Ax = b, \quad A \in GL(\mathbb{R}^n).$$

In that case (5) has the form

$$B_{k+1} := B_k + \frac{(b - Ax^{k+1})(s^k)^\top}{\|s^k\|_2}$$

and with $(B_k - A)s^k + b - Ax^{k+1} = 0$ it follows that

$$B_{k+1} - A = (B_k - A)(I - P_k), \quad P_k = \frac{s^k(s^k)^\top}{(s^k)^\top s^k}. \tag{21}$$

Here $I - P_k$ is the orthogonal projection onto the orthogonal complement of the linear space spanned by s^k.

We introduce now the matrices $E_j = A^{-1}B_j - I$. Then it follows from (21) that $\|E_{j+1}\|_2 \le \|E_j\|_2$, $j \ge 0$. Moreover,

$$B_j s^j = Ax^j - b = A(x^j - x^*), \quad x^* = A^{-1}b,$$

implies that

$$x^j - x^* = A^{-1}B_j s^j = (E_j + I)s^j,$$

and hence that

$$\Big(1 - \frac{\|E_j s^j\|_2}{\|s^j\|_2}\Big)\|s^j\|_2 \le \|x^j - x^*\|_2 \le \Big(1 + \frac{\|E_j s^j\|_2}{\|s^j\|_2}\Big)\|s^j\|_2.$$

Under the conditions of the local convergence theorem **1.2** one can show that $\lim_{j\to\infty} \|E_j s^j\|_2/\|s^j\|_2 = 0$. This leads to the asymptotic error estimate

$$\|x^j - x^*\|_2 \;\approx\; \|s^j\|_2$$

In order to smooth any possible erratic behavior, it is here useful to work with the average of several steps and to declare convergence if

$$\epsilon := \frac{1}{2}\Big[\|s^{j-1}\|_2 + 2\|s^j\|_2 + \|s^{j+1}\|_2\Big]^{1/2} \;\le\; \eta\;\|x^j\|_2\; tol, \tag{22}$$

with some given safety factor $\beta < 1$. Then the algorithm has the form:

> **input:** $A, b, y, B_0, k_{\max}, \tau_{\min}, tol$;
> $r := b - Ay$;
> $s^0 = B_0^{-1} r$; $\eta_0 := (s^0)^\top s^0$; store s^0, η_0;
> **for** $k = 0, 1, \ldots, k_{\max}$
> $q := As^k$; $z = B_0^{-1} q$;
> **if** $k > 0$
> **for** $j = 1, \ldots, k$
> $z := z + \Big[(s^{j-1})^\top z \;/\; \eta_j\Big](s^j - s^{j-1})$;
> **endfor**
> **endif**
> $\tau := \eta_k/[(s^k)^\top z]$;
> **if** $\tau < \tau_{\min}$ **then** return $\{restart\}$;
> $x := x + s^k$; $s^{k+1} := \tau(s^k - z)$;
> $\eta_{k+1} := (s^{k+1})^\top s^{k+1}$; Store s^{k+1}, η_{k+1};
> $\epsilon := (1/2)[s^{k-1} + 2s^k + s^{k+1}]^{1/2}$;
> **if** $\epsilon \le \beta\;\|x\|_2\; tol$ **then** return $\{x^* := x + s^{k+1}\}$;
> **endfor**
> return $\{maximal\ number\ of\ steps\}$

A FORTRAN implementation is the GBITR program in the ZIB-Elib library. Note that for the matrix A only a facility for computing the product Ax, $\forall x \in \mathbb{R}^n$, has to be provided.

4 Rank-Two Updates

The variety of possible methods increases considerably in the rank-two case. Many of these methods have been developed for application in optimization problems. In that case the interest centers on update formulas, which preserve the symmetry of the matrices. Evidently, the direct updates should then have the form

$$B_{k+1} = B_k + \begin{pmatrix} b & c \end{pmatrix} \Sigma \begin{pmatrix} b^\top \\ c^\top \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \sigma_1 & \sigma_2 \\ \sigma_2 & \sigma_3 \end{pmatrix}, \quad b, c \in \mathbb{R}^n. \tag{23}$$

Some examples show that here the matrix Σ should be nonsingular with a negative determinant, otherwise there may be convergence problems.

Since the vectors b, c are essentially free, some suitable basis in $\mathbb{R}^2$ may be chosen in which Σ assumes a simpler form. In particular, because of $\det \Sigma < 0$ we may transform Σ such that either σ_1 or σ_3 is zero. In fact, if, say, $\sigma_3 \neq 0$ then a simple calculation shows that

$$\Sigma = \begin{pmatrix} 1 & \mu \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & \delta \\ \delta & \sigma_3 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ \mu & 1 \end{pmatrix}, \; \mu = \frac{\sigma_2 - \delta}{\sigma_3}, \; \delta = \sqrt{-\det \Sigma}.$$

Thus, there is no loss of generality to assume that $\sigma_1 = 0$ in (23). As before we use the abbreviations

$$s^k = x^{k+1} - x^k, \quad y^k = F(x^{k+1}) - F(x^k). \tag{24}$$

The condition (4) requires $y^k - B_k s^k$ to be in the subspace spanned by b and c and hence it is no restriction to set $b = y^k - B_k s^k$. Then, for any $c \in \mathbb{R}^n$ such that $c^\top s^k \neq 0$ it follows that $\sigma_2 = 1/c^\top s^k$ and $\sigma_3 = -(y^k - B_k s^k)^\top s^k/(c^\top s^k)^2$. In other words, all symmetric direct update formulas with nonpositive determinant can be written in the form

$$B_{k+1} = B_k + \frac{(y^k - B_k s^k)c^\top + c(y^k - B_k s^k)^\top}{c^\top s^k} - \frac{(y^k - B_k s^k)^\top s^k}{(c^\top s^k)^2}\, cc^\top, \tag{25}$$

provided, of course, that $c^\top s^k \neq 0$.

For $c = s^k$ (25) becomes the *Powell-symmetric-Broyden* (PSB) update formula

$$B_{k+1} = B_k + \frac{(y^k - B_k s^k)(s^k)^\top + s^k(y^k - B_k s^k)^\top}{(s^k)^\top s^k} - \frac{(y^k - B_k s^k)^\top s^k}{((s^k)^\top s^k)^2}\, s^k(s^k)^\top \tag{26}$$

of M. J. D. Powell, while for $c = y^k$ we obtain the *Davidon–Fletcher–Powell* (DFP) update formula

$$B_{k+1} = B_k + \frac{(y^k - B_k s^k)(y^k)^\top + y^k(y^k - B_k s^k)^\top}{(y^k)^\top s^k} - \frac{(y^k - B_k s^k)^\top s^k}{((y^k)^\top s^k)^2}\, y^k(y^k)^\top \tag{27}$$

given by D. Davidon and independently by R. Fletcher and M. J. D. Powell.

Instead of working with the direct update (23) we may consider updating the inverses $H_k = B_k^{-1}$ such that $H_{k+1} - H_k$ has rank two. Here we can begin with $H_{k+1} - H_k$ in a form analogous with (23) and then proceed as before. We will not go into details, but mention only one of the formulas that can be obtained in this way. It was independently suggested by C. G. Broyden, R. Fletcher, D. Goldfarb and D. F. Shanno, and is generally called the *BFGS formula* reflecting the first letters of the four authors.

$$H_{k+1} = \left(I - \frac{s^k\,(y^k)^\top}{(y^k)^\top s^k}\right) H_k \left(I - \frac{y^k\,(s^k)^\top}{(y^k)^\top s^k}\right) + \frac{s^k\,(s^k)^\top}{(y^k)^\top s^k}. \tag{28}$$

This is widely considered the most effective update formula for minimization problems.

As before, we can apply here the Sherman-Morrison formula **1.1** and then obtain the direct-update form of the BFGS update

$$B_{k+1} = B_k + \frac{y^k\,(y^k)^\top}{(y^k)^\top s^k} - \frac{B_k s^k (B_k s^k)^\top}{(s^k)^\top B_k s^k}. \tag{29}$$

5 The BFGS Method in Optimization

Extremal problems are of foremost importance in almost all applications of mathematics. Many boundary value problems of mathematical physics may be phrased as variational problems. For instance, holonomic equilibrium problem in Lagrangian mechanics derive from the minimization of a suitable energy function. Similarly, the determination of a geodesic between two points on a manifold is a minimization problem, and so are optimal control problems in engineering, or problems involving the optimal determination of unknown parameters of a technical process. There are close connections between such extremal problems and the solution of nonlinear equations, as is readily seen in the finite dimensional case.

Let $g : \Omega \subset \mathbb{R}^n \longrightarrow \mathbb{R}^1$ be some functional on some set Ω. A point $x^* \in \Omega$ is a *local minimizer* of g in Ω if there exists an open neighborhood $\mathcal{U}$ of x^* in $\mathbb{R}^n$ such that

$$g(x) \geq g(x^*), \quad \forall\, x \in \mathcal{U} \cap \Omega, \tag{30}$$

and a *global minimizer* on Ω if the inequality (30) holds for all $x \in \Omega$. A point x^* in the interior $\mathrm{int}(\Omega)$ of Ω is a *critical point* of g if g has a derivative at x^* and $Dg(x^*)^\top = 0$. A well-known result states that if $x^* \in \mathrm{int}(E)$ is a local minimizer where g is differentiable, then x^* is a critical point of g.

Of course, a critical point need not be local minimizer. But if g has a continuous second derivative at a critical point $x^* \in \mathrm{int}\Omega$ and the Hessian matrix $D^2g(x^*)$ is positive definite then x^* is a proper local minimizer; that is, strict inequality holds in (30) for all $x \in \mathcal{U} \cap \Omega$, $x \neq x^*$. Conversely, at a local minimizer x^*, $D^2g(x^*)$ is positive semi-definite.

For a differentiable functional $g : \Omega \subset \mathbb{R}^n \longrightarrow \mathbb{R}^1$ we call the transposed first derivative $\nabla g(x) = Dg(x)^\top \in \mathbb{R}^n$ the gradient of g at $x \in \Omega$. The problem of finding critical points of g is precisely that of solving the gradient system

$$\nabla g(x) = 0, \quad x \in \Omega. \tag{31}$$

Conversely, a differentiable mapping $F : \Omega \subset \mathbb{R}^n \longrightarrow \mathbb{R}^n$ is called a *gradient* or *potential mapping* on Ω if there exists a differentiable functional $g : \Omega \subset \mathbb{R}^n \longrightarrow \mathbb{R}^1$ such that $F(x) = \nabla g(x)$ for all $x \in \Omega$. A continuously differentiable mapping F on an open convex set Ω is a gradient mapping on Ω if and only if $DF(x)$ is symmetric for all $x \in \Omega$. This is called Kerner's theorem. For any gradient mapping the problem of solving $F(x) = 0$ may be replaced by that of minimizing the functional g, provided, of course, we keep in mind that a local minimizer of g need not be a critical point, nor that a critical point is necessarily a minimizer.

Let $g : \Omega \subset \mathbb{R}^n \longrightarrow \mathbb{R}^1$ be a (sufficiently smooth) functional for which we want to compute a minimizer. Many of the iterative methods for this purpose have the general form

$$x^{k+1} = x^k - \lambda_k d^k, \qquad k = 0, 1, \cdots, \tag{32}$$

involving a direction vector $d^k \in \mathbb{R}^n$ and a steplength $\lambda_k \geq 0$ chosen such that

$$g(x^k) > g(x^{k+1}), \qquad k = 0, 1, \cdots, \tag{33}$$

Obviously, it will not suffice to ensure only a decrease of the value of g, but to require that the decrease (33) is sufficiently large. Thus, at the k-th step of the methods the major tasks are the selection of a suitable direction vector d^k and the construction of an appropriate steplength λ_k. The literature in this area is very extensive, see, e.g., [Kel99] and [Rhe98] for an introduction and further references.

Clearly, given a current point $x \in \Omega$, we want to use a (nonzero direction vector d such that for some $\delta > 0$ we have $g(x - td) \leq g(x)$ for $t \in [0, \delta)$. From $\lim_{t\to 0}(1/t)[g(x) - g(x - tp)] = Dg(x)p$ it follows that, in order for this to hold it is sufficient that $Dg(x)d > 0$ and necessary that $Dg(x)d \geq 0$. Accordingly, we call a vector $d \neq 0$ an *admissible direction* of g at a point x if $Dg(x)d > 0$.

In accordance with the linearization methods (2) we consider now methods of the form

$$x^{k+1} = x^k + \lambda_k B_k^{-1} \nabla g(x^k), \qquad k = 0, 1, \cdots . \tag{34}$$

Hence the direction vectors are here

$$d^k := B_k^{-1} \nabla g(x^k), \qquad k = 0, 1, \cdots . \tag{35}$$

If the matrices B_k are assumed to be symmetric, positive definite, then we have

$$Dg(x)d^k = Dg(x)B_k^{-1}\nabla g(x^k) = (\nabla g(x^k))^\top B_k^{-1} \nabla g(x^k) > 0 \quad \text{if } \nabla g(x^k) \neq 0, \tag{36}$$

that is, the directions (35) are admissible.

This is the reason, why in section **4** the emphasis was placed on the construction of update formulas that preserve symmetry. Actually many of these update methods also preserve positive definiteness. In particular, this holds for the BFGS formula:

5.1. *With the abbreviations (24) suppose that B_k is symmetric, positive definite, and that $(y^k)^\top s^k > 0$. Then B_{k+1} given by (29) is also symmetric, positive definite.*

Proof. By (28) we have

$$B_{k+1}^{-1} = \left(I - \frac{s^k\ (y^k)^\top}{(y^k)^\top s^k}\right) B_k^{-1} \left(I - \frac{y^k\ (s^k)^\top}{(y^k)^\top s^k}\right) + \frac{s^k\ (s^k)^\top}{(y^k)^\top s^k}. \tag{37}$$

Thus, under the stated conditions we have

$$(z^\top B_k s^k) \leq ((s^k)^\top B_k s^k)\ (z^\top B_k z) \qquad \forall z \neq 0,$$

with equality only if $z = 0$ and $s^k = 0$. Moreover, it follows from (29) that

$$z^\top B_{k+1} z = \frac{(z^\top\ y)^2}{y^\top\ s^k} + z^\top B_k z - \frac{(z^\top B_k s^k)^2}{(s^k)^\top B_k s^k}$$

whence

$$z^\top B_{k+1} z > \frac{(z^\top\ y^k)^2}{(y^k)^\top\ s^k} \geq 0.$$

as claimed. □

A step of a descent method of the form (34) with the BFGS update formula has now the generic form:

> Compute the search direction $d^k = H_k \nabla g(x^k)$;
> Determine suitable $\lambda_k > 0$ such that $g(x^k) - g(x^k + \lambda_k\ d^k) > 0$;
> $s^k = \lambda_k\ d^k$;
> $x^{k+1} = x^k + s^k$;
> $y^k = \nabla g(x^{k+1}) - \nabla g(x^k)$;
> If $(y^k)^\top s^k \leq 0$ then return;
> Update H_k to H_{k+1} by means of the BFGS formula.

Numerous algorithms have been proposed for constructing an acceptable step λ_k. One of the simplest is the so called Armijo rule, where we search along the line $t > 0 \longmapsto x^k + td^k$ for a point such that

$$g(x^k) - g(x^k + td^k) > t\alpha \|\nabla g(x^k)\|_2^2, \tag{38}$$

where, say, $\alpha = 10^{-4}$. More specifically we use a backtracking approach and test (38) first with $t = 1$ and then with succesively smaller $t = \beta^j, \quad j = 0, 1, \dots, jmax$, where $0 < \beta < 1$. In other words, the algorithm has the generic form:

> **input** $g, x^k, d^k,\ \alpha, \beta, j_{\max}$,
> $\quad gk = g(x^k); \quad p = \nabla g(x^k)$;
> $\quad \gamma = \alpha\ \|p\|_2^2; \quad t = 1$;
> $\quad$ **for** $j = 0 : j_{\max}$
> $\quad\quad$ **if** $gk - g(x^k + td^k) > t\gamma$ **then** return $\{\lambda_k = t\}$;
> $\quad\quad t = \beta t$;
> $\quad$ **endfor**
> $\quad$ return $\{failure\}$

For the implementation of the overall algorithm one has to decide on the storage of all needed data and on a strategy for a more effective handling of the error case $(y^k)^\top s^k \leq 0$. These issues are discussed, e.g., in chapter 4 of [Kel99].

The simplest approach is to store the entire matrix H_k, which then allows for the computation of the update once the vectors s^k and y^k are available. Clearly, this is costly in storage for large dimensions. A second possibility is to store the sequences $\{s^k\}$ and $\{y^k\}$ and then to recompute recursively the matrices by means of (29) when they are needed.

It turns out that with only a modest increase in complexity the required storage can be decreased to one vector per iteration step. We will not enter into the details, but refer to the discussion in section 4.2.1 of [Kel99]. There also a Matlab implementation 'bfgsopt' involving the above Armijo algorithm is given.

Certainly the BFGS updates are not the only possible choice. In fact, numerous other software packages exist that implement quasi-Newton methods for minimization problems have been written; see, e.g., [MW93].

References

[Deu04] P. Deuflhard, *Newton Methods for Nonlinear Problems*, Springer verlag, Heidelberg, New York, 2004.

[DS98] J. E. Dennis and Robert B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, Classics in Applied Mathematics, Vol 16, SIAM Publications, Philadelphia, PA, 1998. Originally published by Prentice Hall 1983.

[Kel03] C. T. Kelley, *Solving Nonlinear Equations with Newton's Method*, Fundamentals of Algorithms, SIAM Publications, Philadelphia, PA, 2003.

[Kel95] ______, *Iterative Methods for Linear and Nonlinear equations*, Frontiers in Appl. Math., vol. 16, SIAM Publications, Philadelphia, PA, 1995.

[Kel99] ______, *Iterative Methods for Optimization*, Frontiers in Appl. Math., vol. 18, SIAM Publications, Philadelphia, PA, 1999.

[MW93] J. J. More and S. J. Wright, *Optimization Software Guide*, SIAM Publications, Philadelphia, PA, 1993.

[OR00] J. M. Ortega and W. C. Rheinboldt, *Iterative Solutions of Nonlinear Equations in Several Variables*, Classics in Applied Mathematics, Vol 30, SIAM Publications, Philadelphia, PA, 2000. Originally published by Academic Press, 1970, Russian translation 1976, Chinese translation 1982.

[Rhe98] W. C. Rheinboldt, *Methods for Solving Systems of Nonlinear Equations*, Regional Conf. Series in Appl. Math., Vol. 70, Siam Publications, Philadelphia, PA, 1998.