

# CHAPTER 6

## Quasi-Newton Methods

In the mid 1950s, W.C. Davidon, a physicist working at Argonne National Laboratory, was using the coordinate descent method (see Section 9.3) to perform a long optimization calculation. At that time computers were not very stable, and to Davidon's frustration, the computer system would always crash before the calculation was finished. So Davidon decided to find a way of accelerating the iteration. The algorithm he developed—the first quasi-Newton algorithm—turned out to be one of the most creative ideas in nonlinear optimization. It was soon demonstrated by Fletcher and Powell that the new algorithm was much faster and more reliable than the other existing methods, and this dramatic

advance transformed nonlinear optimization overnight. During the following twenty years, numerous variants were proposed and hundreds of papers were devoted to their study. An interesting historical irony is that Davidon's paper [87] was not accepted for publication; it remained as a technical report for more than thirty years until it appeared in the first issue of the *SIAM Journal on Optimization* in 1991 [88].

Quasi-Newton methods, like steepest descent, require only the gradient of the objective function to be supplied at each iterate. By measuring the changes in gradients, they construct a model of the objective function that is good enough to produce superlinear convergence. The improvement over steepest descent is dramatic, especially on difficult problems. Moreover, since second derivatives are not required, quasi-Newton methods are sometimes more efficient than Newton's method. Today, optimization software libraries contain a variety of quasi-Newton algorithms for solving unconstrained, constrained, and large-scale optimization problems. In this chapter we discuss quasi-Newton methods for small and medium-sized problems, and in Chapter 7 we consider their extension to the large-scale setting.

The development of automatic differentiation techniques has made it possible to use Newton's method without requiring users to supply second derivatives; see Chapter 8. Still, automatic differentiation tools may not be applicable in many situations, and it may be much more costly to work with second derivatives in automatic differentiation software than with the gradient. For these reasons, quasi-Newton methods remain appealing.

## 6.1 THE BFGS METHOD

The most popular quasi-Newton algorithm is the BFGS method, named for its discoverers Broyden, Fletcher, Goldfarb, and Shanno. In this section we derive this algorithm (and its close relative, the DFP algorithm) and describe its theoretical properties and practical implementation.

We begin the derivation by forming the following quadratic model of the objective function at the current iterate  $x_k$ :

$$m_k(p) = f_k + \nabla f_k^T p + \frac{1}{2} p^T B_k p. \quad (6.1)$$

Here  $B_k$  is an  $n \times n$  symmetric positive definite matrix that will be revised or *updated* at every iteration. Note that the function value and gradient of this model at  $p = 0$  match  $f_k$  and  $\nabla f_k$ , respectively. The minimizer  $p_k$  of this convex quadratic model, which we can write explicitly as

$$p_k = -B_k^{-1} \nabla f_k, \quad (6.2)$$

is used as the search direction, and the new iterate is

$$x_{k+1} = x_k + \alpha_k p_k, \quad (6.3)$$

where the step length  $\alpha_k$  is chosen to satisfy the Wolfe conditions (3.6). This iteration is quite similar to the line search Newton method; the key difference is that the approximate Hessian  $B_k$  is used in place of the true Hessian.

Instead of computing  $B_k$  afresh at every iteration, Davidon proposed to update it in a simple manner to account for the curvature measured during the most recent step. Suppose that we have generated a new iterate  $x_{k+1}$  and wish to construct a new quadratic model, of the form

$$m_{k+1}(p) = f_{k+1} + \nabla f_{k+1}^T p + \frac{1}{2} p^T B_{k+1} p.$$

What requirements should we impose on  $B_{k+1}$ , based on the knowledge gained during the latest step? One reasonable requirement is that the gradient of  $m_{k+1}$  should match the gradient of the objective function  $f$  at the latest two iterates  $x_k$  and  $x_{k+1}$ . Since  $\nabla m_{k+1}(0)$  is precisely  $\nabla f_{k+1}$ , the second of these conditions is satisfied automatically. The first condition can be written mathematically as

$$\nabla m_{k+1}(-\alpha_k p_k) = \nabla f_{k+1} - \alpha_k B_{k+1} p_k = \nabla f_k.$$

By rearranging, we obtain

$$B_{k+1} \alpha_k p_k = \nabla f_{k+1} - \nabla f_k. \quad (6.4)$$

To simplify the notation it is useful to define the vectors

$$s_k = x_{k+1} - x_k = \alpha_k p_k, \quad y_k = \nabla f_{k+1} - \nabla f_k, \quad (6.5)$$

so that (6.4) becomes

$$B_{k+1} s_k = y_k. \quad (6.6)$$

We refer to this formula as the *secant equation*.

Given the displacement  $s_k$  and the change of gradients  $y_k$ , the secant equation requires that the symmetric positive definite matrix  $B_{k+1}$  map  $s_k$  into  $y_k$ . This will be possible only if  $s_k$  and  $y_k$  satisfy the *curvature condition*

$$s_k^T y_k > 0, \quad (6.7)$$

as is easily seen by premultiplying (6.6) by  $s_k^T$ . When  $f$  is strongly convex, the inequality (6.7) will be satisfied for any two points  $x_k$  and  $x_{k+1}$  (see Exercise 6.1). However, this condition

will not always hold for nonconvex functions, and in this case we need to enforce (6.7) explicitly, by imposing restrictions on the line search procedure that chooses the step length  $\alpha$ . In fact, the condition (6.7) is guaranteed to hold if we impose the Wolfe (3.6) or strong Wolfe conditions (3.7) on the line search. To verify this claim, we note from (6.5) and (3.6b) that  $\nabla f_{k+1}^T s_k \geq c_2 \nabla f_k^T s_k$ , and therefore

$$y_k^T s_k \geq (c_2 - 1) \alpha_k \nabla f_k^T p_k. \quad (6.8)$$

Since  $c_2 < 1$  and since  $p_k$  is a descent direction, the term on the right is positive, and the curvature condition (6.7) holds.

When the curvature condition is satisfied, the secant equation (6.6) always has a solution  $B_{k+1}$ . In fact, it admits an infinite number of solutions, since the  $n(n+1)/2$  degrees of freedom in a symmetric positive definite matrix exceed the  $n$  conditions imposed by the secant equation. The requirement of positive definiteness imposes  $n$  additional inequalities—all principal minors must be positive—but these conditions do not absorb the remaining degrees of freedom.

To determine  $B_{k+1}$  uniquely, we impose the additional condition that *among all symmetric matrices satisfying the secant equation,  $B_{k+1}$  is, in some sense, closest to the current matrix  $B_k$* . In other words, we solve the problem

$$\min_B \|B - B_k\| \quad (6.9a)$$

$$\text{subject to } B = B^T, \quad B s_k = y_k, \quad (6.9b)$$

where  $s_k$  and  $y_k$  satisfy (6.7) and  $B_k$  is symmetric and positive definite. Different matrix norms can be used in (6.9a), and each norm gives rise to a different quasi-Newton method. A norm that allows easy solution of the minimization problem (6.9) and gives rise to a scale-invariant optimization method is the weighted Frobenius norm

$$\|A\|_W \equiv \|W^{1/2} A W^{1/2}\|_F, \quad (6.10)$$

where  $\|\cdot\|_F$  is defined by  $\|C\|_F^2 = \sum_{i=1}^n \sum_{j=1}^n c_{ij}^2$ . The weight matrix  $W$  can be chosen as *any* matrix satisfying the relation  $W y_k = s_k$ . For concreteness, the reader can assume that  $W = \bar{G}_k^{-1}$  where  $\bar{G}_k$  is the *average Hessian* defined by

$$\bar{G}_k = \left[ \int_0^1 \nabla^2 f(x_k + \tau \alpha_k p_k) d\tau \right]. \quad (6.11)$$

The property

$$y_k = \bar{G}_k \alpha_k p_k = \bar{G}_k s_k \quad (6.12)$$

follows from Taylor's theorem, Theorem 2.1. With this choice of weighting matrix  $W$ , the

norm (6.10) is non-dimensional, which is a desirable property, since we do not wish the solution of (6.9) to depend on the units of the problem.

With this weighting matrix and this norm, the unique solution of (6.9) is

$$(DFP) \quad B_{k+1} = (I - \rho_k y_k s_k^T) B_k (I - \rho_k s_k y_k^T) + \rho_k y_k y_k^T, \quad (6.13)$$

with

$$\rho_k = \frac{1}{y_k^T s_k}. \quad (6.14)$$

This formula is called the DFP updating formula, since it is the one originally proposed by Davidon in 1959, and subsequently studied, implemented, and popularized by Fletcher and Powell.

The inverse of  $B_k$ , which we denote by

$$H_k = B_k^{-1},$$

is useful in the implementation of the method, since it allows the search direction (6.2) to be calculated by means of a simple matrix–vector multiplication. Using the Sherman–Morrison–Woodbury formula (A.28), we can derive the following expression for the update of the inverse Hessian approximation  $H_k$  that corresponds to the DFP update of  $B_k$  in (6.13):

$$(DFP) \quad H_{k+1} = H_k - \frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k} + \frac{s_k s_k^T}{y_k^T s_k}. \quad (6.15)$$

Note that the last two terms in the right-hand-side of (6.15) are rank-one matrices, so that  $H_k$  undergoes a rank-two modification. It is easy to see that (6.13) is also a rank-two modification of  $B_k$ . This is the fundamental idea of quasi-Newton updating: Instead of recomputing the approximate Hessians (or inverse Hessians) from scratch at every iteration, we apply a simple modification that combines the most recently observed information about the objective function with the existing knowledge embedded in our current Hessian approximation.

The DFP updating formula is quite effective, but it was soon superseded by the BFGS formula, which is presently considered to be the most effective of all quasi-Newton updating formulae. BFGS updating can be derived by making a simple change in the argument that led to (6.13). Instead of imposing conditions on the Hessian approximations  $B_k$ , we impose similar conditions on their inverses  $H_k$ . The updated approximation  $H_{k+1}$  must be symmetric and positive definite, and must satisfy the secant equation (6.6), now written as

$$H_{k+1} y_k = s_k.$$

The condition of closeness to  $H_k$  is now specified by the following analogue of (6.9):

$$\min_H \|H - H_k\| \quad (6.16a)$$

$$\text{subject to } H = H^T, \quad H y_k = s_k. \quad (6.16b)$$

The norm is again the weighted Frobenius norm described above, where the weight matrix  $W$  is now any matrix satisfying  $Ws_k = y_k$ . (For concreteness, we assume again that  $W$  is given by the average Hessian  $\tilde{G}_k$  defined in (6.11).) The unique solution  $H_{k+1}$  to (6.16) is given by

$$\text{(BFGS)} \quad H_{k+1} = (I - \rho_k s_k y_k^T) H_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T, \quad (6.17)$$

with  $\rho_k$  defined by (6.14).

Just one issue has to be resolved before we can define a complete BFGS algorithm: How should we choose the initial approximation  $H_0$ ? Unfortunately, there is no magic formula that works well in all cases. We can use specific information about the problem, for instance by setting it to the inverse of an approximate Hessian calculated by finite differences at  $x_0$ . Otherwise, we can simply set it to be the identity matrix, or a multiple of the identity matrix, where the multiple is chosen to reflect the scaling of the variables.

**Algorithm 6.1** (BFGS Method).

Given starting point  $x_0$ , convergence tolerance  $\epsilon > 0$ ,

inverse Hessian approximation  $H_0$ ;

$k \leftarrow 0$ ;

**while**  $\|\nabla f_k\| > \epsilon$ ;

    Compute search direction

$$p_k = -H_k \nabla f_k; \quad (6.18)$$

    Set  $x_{k+1} = x_k + \alpha_k p_k$  where  $\alpha_k$  is computed from a line search procedure to satisfy the Wolfe conditions (3.6);

    Define  $s_k = x_{k+1} - x_k$  and  $y_k = \nabla f_{k+1} - \nabla f_k$ ;

    Compute  $H_{k+1}$  by means of (6.17);

$k \leftarrow k + 1$ ;

**end (while)**

Each iteration can be performed at a cost of  $O(n^2)$  arithmetic operations (plus the cost of function and gradient evaluations); there are no  $O(n^3)$  operations such as linear system solves or matrix–matrix operations. The algorithm is robust, and its rate of convergence is superlinear, which is fast enough for most practical purposes. Even though Newton’s method converges more rapidly (that is, quadratically), its cost per iteration usually is higher, because of its need for second derivatives and solution of a linear system.

We can derive a version of the BFGS algorithm that works with the Hessian approximation  $B_k$  rather than  $H_k$ . The update formula for  $B_k$  is obtained by simply applying the Sherman–Morrison–Woodbury formula (A.28) to (6.17) to obtain

$$\text{(BFGS)} \quad B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}. \quad (6.19)$$

A naive implementation of this variant is not efficient for unconstrained minimization, because it requires the system  $B_k p_k = -\nabla f_k$  to be solved for the step  $p_k$ , thereby increasing the cost of the step computation to  $O(n^3)$ . We discuss later, however, that less expensive implementations of this variant are possible by updating Cholesky factors of  $B_k$ .

### PROPERTIES OF THE BFGS METHOD

It is usually easy to observe the superlinear rate of convergence of the BFGS method on practical problems. Below, we report the last few iterations of the steepest descent, BFGS, and an inexact Newton method on Rosenbrock's function (2.22). The table gives the value of  $\|x_k - x^*\|$ . The Wolfe conditions were imposed on the step length in all three methods. From the starting point  $(-1.2, 1)$ , the steepest descent method required 5264 iterations, whereas BFGS and Newton took only 34 and 21 iterations, respectively to reduce the gradient norm to  $10^{-5}$ .

steepest descent	BFGS	Newton
1.827e-04	1.70e-03	3.48e-02
1.826e-04	1.17e-03	1.44e-02
1.824e-04	1.34e-04	1.82e-04
1.823e-04	1.01e-06	1.17e-08

A few points in the derivation of the BFGS and DFP methods merit further discussion. Note that the minimization problem (6.16) that gives rise to the BFGS update formula does not explicitly require the updated Hessian approximation to be positive definite. It is easy to show, however, that  $H_{k+1}$  will be positive definite whenever  $H_k$  is positive definite, by using the following argument. First, note from (6.8) that  $y_k^T s_k$  is positive, so that the updating formula (6.17), (6.14) is well-defined. For any nonzero vector  $z$ , we have

$$z^T H_{k+1} z = w^T H_k w + \rho_k (z^T s_k)^2 \geq 0,$$

where we have defined  $w = z - \rho_k y_k (s_k^T z)$ . The right hand side can be zero only if  $s_k^T z = 0$ , but in this case  $w = z \neq 0$ , which implies that the first term is greater than zero. Therefore,  $H_{k+1}$  is positive definite.

To make quasi-Newton updating formulae invariant to transformations in the variables (such as scaling transformations), it is necessary for the objectives (6.9a) and (6.16a) to be invariant under the same transformations. The choice of the weighting matrices  $W$  used to define the norms in (6.9a) and (6.16a) ensures that this condition holds. Many other choices of the weighting matrix  $W$  are possible, each one of them giving a different update formula. However, despite intensive searches, no formula has been found that is significantly more effective than BFGS.

The BFGS method has many interesting properties when applied to quadratic functions. We discuss these properties later in the more general context of the Broyden family of updating formulae, of which BFGS is a special case.

It is reasonable to ask whether there are situations in which the updating formula such as (6.17) can produce bad results. If at some iteration the matrix  $H_k$  becomes a poor approximation to the true inverse Hessian, is there any hope of correcting it? For example, when the inner product  $y_k^T s_k$  is tiny (but positive), then it follows from (6.14), (6.17) that  $H_{k+1}$  contains very large elements. Is this behavior reasonable? A related question concerns the rounding errors that occur in finite-precision implementation of these methods. Can these errors grow to the point of erasing all useful information in the quasi-Newton approximate Hessian?

These questions have been studied analytically and experimentally, and it is now known that the BFGS formula has very effective self-correcting properties. If the matrix  $H_k$  incorrectly estimates the curvature in the objective function, and if this bad estimate slows down the iteration, then the Hessian approximation will tend to correct itself within a few steps. It is also known that the DFP method is less effective in correcting bad Hessian approximations; this property is believed to be the reason for its poorer practical performance. The self-correcting properties of BFGS hold only when an adequate line search is performed. In particular, the Wolfe line search conditions ensure that the gradients are sampled at points that allow the model (6.1) to capture appropriate curvature information.

It is interesting to note that the DFP and BFGS updating formulae are *duals* of each other, in the sense that one can be obtained from the other by the interchanges  $s \leftrightarrow y$ ,  $B \leftrightarrow H$ . This symmetry is not surprising, given the manner in which we derived these methods above.

## IMPLEMENTATION

A few details and enhancements need to be added to Algorithm 6.1 to produce an efficient implementation. The line search, which should satisfy either the Wolfe conditions (3.6) or the strong Wolfe conditions (3.7), should always try the step length  $\alpha_k = 1$  first, because this step length will eventually always be accepted (under certain conditions), thereby producing superlinear convergence of the overall algorithm. Computational observations strongly suggest that it is more economical, in terms of function evaluations, to perform a fairly inaccurate line search. The values  $c_1 = 10^{-4}$  and  $c_2 = 0.9$  are commonly used in (3.6).

As mentioned earlier, the initial matrix  $H_0$  often is set to some multiple  $\beta I$  of the identity, but there is no good general strategy for choosing the multiple  $\beta$ . If  $\beta$  is too large, so that the first step  $p_0 = -\beta g_0$  is too long, many function evaluations may be required to find a suitable value for the step length  $\alpha_0$ . Some software asks the user to prescribe a value  $\delta$  for the norm of the first step, and then set  $H_0 = \delta \|g_0\|^{-1} I$  to achieve this norm.

A heuristic that is often quite effective is to scale the starting matrix *after* the first step has been computed but before the first BFGS update is performed. We change the



provisional value  $H_0 = I$  by setting

$$H_0 \leftarrow \frac{y_k^T s_k}{y_k^T y_k} I, \quad (6.20)$$

before applying the update (6.14), (6.17) to obtain  $H_1$ . This formula attempts to make the size of  $H_0$  similar to that of  $\nabla^2 f(x_0)^{-1}$ , in the following sense. Assuming that the average Hessian defined in (6.11) is positive definite, there exists a square root  $\bar{G}_k^{1/2}$  satisfying  $\bar{G}_k = \bar{G}_k^{1/2} \bar{G}_k^{1/2}$  (see Exercise 6.6). Therefore, by defining  $z_k = \bar{G}_k^{1/2} s_k$  and using the relation (6.12), we have

$$\frac{y_k^T s_k}{y_k^T y_k} = \frac{(\bar{G}_k^{1/2} s_k)^T \bar{G}_k^{1/2} s_k}{(\bar{G}_k^{1/2} s_k)^T \bar{G}_k \bar{G}_k^{1/2} s_k} = \frac{z_k^T z_k}{z_k^T \bar{G}_k z_k}. \quad (6.21)$$

The *reciprocal* of (6.21) is an approximation to one of the eigenvalues of  $\bar{G}_k$ , which in turn is close to an eigenvalue of  $\nabla^2 f(x_k)$ . Hence, the quotient (6.21) itself approximates an eigenvalue of  $\nabla^2 f(x_k)^{-1}$ . Other scaling factors can be used in (6.20), but the one presented here appears to be the most successful in practice.

In (6.19) we gave an update formula for a BFGS method that works with the Hessian approximation  $B_k$  instead of the the inverse Hessian approximation  $H_k$ . An efficient implementation of this approach does not store  $B_k$  explicitly, but rather the Cholesky factorization  $L_k D_k L_k^T$  of this matrix. A formula that updates the factors  $L_k$  and  $D_k$  *directly* in  $O(n^2)$  operations can be derived from (6.19). Since the linear system  $B_k p_k = -\nabla f_k$  also can be solved in  $O(n^2)$  operations (by performing triangular substitutions with  $L_k$  and  $L_k^T$  and a diagonal substitution with  $D_k$ ), the total cost is quite similar to the variant described in Algorithm 6.1. A potential advantage of this alternative strategy is that it gives us the option of modifying diagonal elements in the  $D_k$  factor if they are not sufficiently large, to prevent instability when we divide by these elements during the calculation of  $p_k$ . However, computational experience suggests no real advantages for this variant, and we prefer the simpler strategy of Algorithm 6.1.

The performance of the BFGS method can degrade if the line search is not based on the Wolfe conditions. For example, some software implements an Armijo backtracking line search (see Section 3.1): The unit step length  $\alpha_k = 1$  is tried first and is successively decreased until the sufficient decrease condition (3.6a) is satisfied. For this strategy, there is no guarantee that the curvature condition (6.7) will be satisfied by the chosen step, since a step length greater than 1 may be required to satisfy this condition. To cope with this shortcoming, some implementations simply *skip* the BFGS update by setting  $H_{k+1} = H_k$  when  $y_k^T s_k$  is negative or too close to zero. This approach is not recommended, because the updates may be skipped much too often to allow  $H_k$  to capture important curvature information for the objective function  $f$ . In Chapter 18 we discuss a *damped* BFGS update that is a more effective strategy for coping with the case where the curvature condition (6.7) is not satisfied.

## 6.2 THE SR1 METHOD

In the BFGS and DFP updating formulae, the updated matrix  $B_{k+1}$  (or  $H_{k+1}$ ) differs from its predecessor  $B_k$  (or  $H_k$ ) by a rank-2 matrix. In fact, as we now show, there is a simpler rank-1 update that maintains symmetry of the matrix and allows it to satisfy the secant equation. Unlike the rank-two update formulae, this *symmetric-rank-1*, or SR1, update does not guarantee that the updated matrix maintains positive definiteness. Good numerical results have been obtained with algorithms based on SR1, so we derive it here and investigate its properties.

The symmetric rank-1 update has the general form

$$B_{k+1} = B_k + \sigma v v^T,$$

where  $\sigma$  is either  $+1$  or  $-1$ , and  $\sigma$  and  $v$  are chosen so that  $B_{k+1}$  satisfies the secant equation (6.6), that is,  $y_k = B_{k+1}s_k$ . By substituting into this equation, we obtain

$$y_k = B_k s_k + [\sigma v^T s_k] v. \quad (6.22)$$

Since the term in brackets is a scalar, we deduce that  $v$  must be a multiple of  $y_k - B_k s_k$ , that is,  $v = \delta(y_k - B_k s_k)$  for some scalar  $\delta$ . By substituting this form of  $v$  into (6.22), we obtain

$$(y_k - B_k s_k) = \sigma \delta^2 [s_k^T (y_k - B_k s_k)] (y_k - B_k s_k), \quad (6.23)$$

and it is clear that this equation is satisfied if (and only if) we choose the parameters  $\delta$  and  $\sigma$  to be

$$\sigma = \text{sign} [s_k^T (y_k - B_k s_k)], \quad \delta = \pm |s_k^T (y_k - B_k s_k)|^{-1/2}.$$

Hence, we have shown that the only symmetric rank-1 updating formula that satisfies the secant equation is given by

$$(SR1) \quad B_{k+1} = B_k + \frac{(y_k - B_k s_k)(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}. \quad (6.24)$$

By applying the Sherman–Morrison formula (A.27), we obtain the corresponding update formula for the inverse Hessian approximation  $H_k$ :

$$(SR1) \quad H_{k+1} = H_k + \frac{(s_k - H_k y_k)(s_k - H_k y_k)^T}{(s_k - H_k y_k)^T y_k}. \quad (6.25)$$

This derivation is so simple that the SR1 formula has been rediscovered a number of times.

It is easy to see that even if  $B_k$  is positive definite,  $B_{k+1}$  may not have the same property. (The same is, of course, true of  $H_k$ .) This observation was considered a major drawback

in the early days of nonlinear optimization when only line search iterations were used. However, with the advent of trust-region methods, the SR1 updating formula has proved to be quite useful, and its ability to generate indefinite Hessian approximations can actually be regarded as one of its chief advantages.

The main drawback of SR1 updating is that the denominator in (6.24) or (6.25) can vanish. In fact, even when the objective function is a convex quadratic, there may be steps on which there is no symmetric rank-1 update that satisfies the secant equation. It pays to reexamine the derivation above in the light of this observation.

By reasoning in terms of  $B_k$  (similar arguments can be applied to  $H_k$ ), we see that there are three cases:

1. If  $(y_k - B_k s_k)^T s_k \neq 0$ , then the arguments above show that there is a unique rank-one updating formula satisfying the secant equation (6.6), and that it is given by (6.24).
2. If  $y_k = B_k s_k$ , then the only updating formula satisfying the secant equation is simply  $B_{k+1} = B_k$ .
3. If  $y_k \neq B_k s_k$  and  $(y_k - B_k s_k)^T s_k = 0$ , then (6.23) shows that there is no symmetric rank-one updating formula satisfying the secant equation.

The last case clouds an otherwise simple and elegant derivation, and suggests that numerical instabilities and even breakdown of the method can occur. It suggests that rank-one updating does not provide enough freedom to develop a matrix with all the desired characteristics, and that a rank-two correction is required. This reasoning leads us back to the BFGS method, in which positive definiteness (and thus nonsingularity) of all Hessian approximations is guaranteed.

Nevertheless, we are interested in the SR1 formula for the following reasons.

- (i) A simple safeguard seems to adequately prevent the breakdown of the method and the occurrence of numerical instabilities.
- (ii) The matrices generated by the SR1 formula tend to be good approximations to the true Hessian matrix—often better than the BFGS approximations.
- (iii) In quasi-Newton methods for constrained problems, or in methods for partially separable functions (see Chapters 18 and 7), it may not be possible to impose the curvature condition  $y_k^T s_k > 0$ , and thus BFGS updating is not recommended. Indeed, in these two settings, indefinite Hessian approximations are desirable insofar as they reflect indefiniteness in the true Hessian.

We now introduce a strategy to prevent the SR1 method from breaking down. It has been observed in practice that SR1 performs well simply by skipping the update if the denominator is small. More specifically, the update (6.24) is applied only if

$$|s_k^T (y_k - B_k s_k)| \geq r \|s_k\| \|y_k - B_k s_k\|, \quad (6.26)$$

where  $r \in (0, 1)$  is a small number, say  $r = 10^{-8}$ . If (6.26) does not hold, we set  $B_{k+1} = B_k$ . Most implementations of the SR1 method use a skipping rule of this kind.

Why do we advocate skipping of updates for the SR1 method, when in the previous section we discouraged this strategy in the case of BFGS? The two cases are quite different. The condition  $s_k^T (y_k - B_k s_k) \approx 0$  occurs infrequently, since it requires certain vectors to be aligned in a specific way. When it does occur, skipping the update appears to have no negative effects on the iteration. This is not surprising, since the skipping condition implies that  $s_k^T \bar{G} s_k \approx s_k^T B_k s_k$ , where  $\bar{G}$  is the average Hessian over the last step—meaning that the curvature of  $B_k$  along  $s_k$  is already correct. In contrast, the curvature condition  $s_k^T y_k \geq 0$  required for BFGS updating may easily fail if the line search does not impose the Wolfe conditions (for example, if the step is not long enough), and therefore skipping the BFGS update can occur often and can degrade the quality of the Hessian approximation.

We now give a formal description of an SR1 method using a trust-region framework, which we prefer over a line search framework because it can accommodate indefinite Hessian approximations more easily.

**Algorithm 6.2** (SR1 Trust-Region Method).

Given starting point  $x_0$ , initial Hessian approximation  $B_0$ ,  
 trust-region radius  $\Delta_0$ , convergence tolerance  $\epsilon > 0$ ,  
 parameters  $\eta \in (0, 10^{-3})$  and  $r \in (0, 1)$ ;  
 $k \leftarrow 0$ ;  
**while**  $\|\nabla f_k\| > \epsilon$ ;  
     Compute  $s_k$  by solving the subproblem

$$\min_s \nabla f_k^T s + \frac{1}{2} s^T B_k s \quad \text{subject to } \|s\| \leq \Delta_k; \quad (6.27)$$

    Compute

$$\begin{aligned} y_k &= \nabla f(x_k + s_k) - \nabla f_k, \\ \text{ared} &= f_k - f(x_k + s_k) \quad (\text{actual reduction}) \\ \text{pred} &= - \left( \nabla f_k^T s_k + \frac{1}{2} s_k^T B_k s_k \right) \quad (\text{predicted reduction}); \end{aligned}$$

**if**  $\text{ared}/\text{pred} > \eta$   
      $x_{k+1} = x_k + s_k$ ;  
**else**  
      $x_{k+1} = x_k$ ;  
**end (if)**

```

if ared/pred > 0.75
    if  $\|s_k\| \leq 0.8\Delta_k$ 
         $\Delta_{k+1} = \Delta_k$ ;
    else
         $\Delta_{k+1} = 2\Delta_k$ ;
    end (if)
else if  $0.1 \leq \text{ared}/\text{pred} \leq 0.75$ 
     $\Delta_{k+1} = \Delta_k$ ;
else
     $\Delta_{k+1} = 0.5\Delta_k$ ;
end (if)
if (6.26) holds
    Use (6.24) to compute  $B_{k+1}$  (even if  $x_{k+1} = x_k$ );
else
     $B_{k+1} \leftarrow B_k$ ;
end (if)
 $k \leftarrow k + 1$ ;
end (while)

```

This algorithm has the typical form of a trust region method (cf. Algorithm 4.1). For concreteness, we have specified a particular strategy for updating the trust region radius, but other heuristics can be used instead.

To obtain a fast rate of convergence, it is important for the matrix  $B_k$  to be updated even along a failed direction  $s_k$ . The fact that the step was poor indicates that  $B_k$  is an inadequate approximation of the true Hessian in this direction. Unless the quality of the approximation is improved, steps along similar directions could be generated on later iterations, and repeated rejection of such steps could prevent superlinear convergence.

### PROPERTIES OF SR1 UPDATING

One of the main advantages of SR1 updating is its ability to generate good Hessian approximations. We demonstrate this property by first examining a quadratic function. For functions of this type, the choice of step length does not affect the update, so to examine the effect of the updates, we can assume for simplicity a uniform step length of 1, that is,

$$p_k = -H_k \nabla f_k, \quad x_{k+1} = x_k + p_k. \quad (6.28)$$

It follows that  $p_k = s_k$ .

#### Theorem 6.1.

*Suppose that  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is the strongly convex quadratic function  $f(x) = b^T x + \frac{1}{2}x^T A x$ , where  $A$  is symmetric positive definite. Then for any starting point  $x_0$  and any*

symmetric starting matrix  $H_0$ , the iterates  $\{x_k\}$  generated by the SR1 method (6.25), (6.28) converge to the minimizer in at most  $n$  steps, provided that  $(s_k - H_k y_k)^T y_k \neq 0$  for all  $k$ . Moreover, if  $n$  steps are performed, and if the search directions  $p_i$  are linearly independent, then  $H_n = A^{-1}$ .

PROOF. Because of our assumption  $(s_k - H_k y_k)^T y_k \neq 0$ , the SR1 update is always well-defined. We start by showing inductively that

$$H_k y_j = s_j, \quad j = 0, 1, \dots, k-1. \quad (6.29)$$

In other words, we claim that the secant equation is satisfied not only along the most recent search direction, but along all previous directions.

By definition, the SR1 update satisfies the secant equation, so we have  $H_1 y_0 = s_0$ . Let us now assume that (6.29) holds for some value  $k > 1$  and show that it holds also for  $k+1$ . From this assumption, we have from (6.29) that

$$(s_k - H_k y_k)^T y_j = s_k^T y_j - y_k^T (H_k y_j) = s_k^T y_j - y_k^T s_j = 0, \quad \text{all } j < k, \quad (6.30)$$

where the last equality follows because  $y_i = A s_i$  for the quadratic function we are considering here. By using (6.30) and the induction hypothesis (6.29) in (6.25), we have

$$H_{k+1} y_j = H_k y_j = s_j, \quad \text{for all } j < k.$$

Since  $H_{k+1} y_k = s_k$  by the secant equation, we have shown that (6.29) holds when  $k$  is replaced by  $k+1$ . By induction, then, this relation holds for all  $k$ .

If the algorithm performs  $n$  steps and if these steps  $\{s_j\}$  are linearly independent, we have

$$s_j = H_n y_j = H_n A s_j, \quad j = 0, 1, \dots, n-1.$$

It follows that  $H_n A = I$ , that is,  $H_n = A^{-1}$ . Therefore, the step taken at  $x_n$  is the Newton step, and so the next iterate  $x_{n+1}$  will be the solution, and the algorithm terminates.

Consider now the case in which the steps become linearly dependent. Suppose that  $s_k$  is a linear combination of the previous steps, that is,

$$s_k = \xi_0 s_0 + \dots + \xi_{k-1} s_{k-1}, \quad (6.31)$$

for some scalars  $\xi_i$ . From (6.31) and (6.29) we have that

$$\begin{aligned} H_k y_k &= H_k A s_k \\ &= \xi_0 H_k A s_0 + \dots + \xi_{k-1} H_k A s_{k-1} \end{aligned}$$

$$\begin{aligned}
&= \xi_0 H_k y_0 + \cdots + \xi_{k-1} H_k y_{k-1} \\
&= \xi_0 s_0 + \cdots + \xi_{k-1} s_{k-1} \\
&= s_k.
\end{aligned}$$

Since  $y_k = \nabla f_{k+1} - \nabla f_k$  and since  $s_k = p_k = -H_k \nabla f_k$  from (6.28), we have that

$$H_k(\nabla f_{k+1} - \nabla f_k) = -H_k \nabla f_k,$$

which, by the nonsingularity of  $H_k$ , implies that  $\nabla f_{k+1} = 0$ . Therefore,  $x_{k+1}$  is the solution point.  $\square$

The relation (6.29) shows that when  $f$  is quadratic, the secant equation is satisfied along all previous search directions, regardless of how the line search is performed. A result like this can be established for BFGS updating only under the restrictive assumption that the line search is exact, as we show in the next section.

For general nonlinear functions, the SR1 update continues to generate good Hessian approximations under certain conditions.

### Theorem 6.2.

*Suppose that  $f$  is twice continuously differentiable, and that its Hessian is bounded and Lipschitz continuous in a neighborhood of a point  $x^*$ . Let  $\{x_k\}$  be any sequence of iterates such that  $x_k \rightarrow x^*$  for some  $x^* \in \mathbf{R}^n$ . Suppose in addition that the inequality (6.26) holds for all  $k$ , for some  $r \in (0, 1)$ , and that the steps  $s_k$  are uniformly linearly independent. Then the matrices  $B_k$  generated by the SR1 updating formula satisfy*

$$\lim_{k \rightarrow \infty} \|B_k - \nabla^2 f(x^*)\| = 0.$$

The term “uniformly linearly independent steps” means, roughly speaking, that the steps do not tend to fall in a subspace of dimension less than  $n$ . This assumption is usually, but not always, satisfied in practice (see the Notes and References at the end of this chapter).

## 6.3 THE BROYDEN CLASS

So far, we have described the BFGS, DFP, and SR1 quasi-Newton updating formulae, but there are many others. Of particular interest is the *Broyden class*, a family of updates specified by the following general formula:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k} + \phi_k (s_k^T B_k s_k) v_k v_k^T, \quad (6.32)$$

where  $\phi_k$  is a scalar parameter and

$$v_k = \left[ \frac{y_k}{y_k^T s_k} - \frac{B_k s_k}{s_k^T B_k s_k} \right]. \quad (6.33)$$

The BFGS and DFP methods are members of the Broyden class—we recover BFGS by setting  $\phi_k = 0$  and DFP by setting  $\phi_k = 1$  in (6.32). We can therefore rewrite (6.32) as a “linear combination” of these two methods, that is,

$$B_{k+1} = (1 - \phi_k) B_{k+1}^{\text{BFGS}} + \phi_k B_{k+1}^{\text{DFP}}.$$

This relationship indicates that all members of the Broyden class satisfy the secant equation (6.6), since the BFGS and DFP matrices themselves satisfy this equation. Also, since BFGS and DFP updating preserve positive definiteness of the Hessian approximations when  $s_k^T y_k > 0$ , this relation implies that the same property will hold for the Broyden family if  $0 \leq \phi_k \leq 1$ .

Much attention has been given to the so-called *restricted Broyden class*, which is obtained by restricting  $\phi_k$  to the interval  $[0, 1]$ . It enjoys the following property when applied to quadratic functions. Since the analysis is independent of the step length, we assume for simplicity that each iteration has the form

$$p_k = -B_k^{-1} \nabla f_k, \quad x_{k+1} = x_k + p_k. \quad (6.34)$$

### Theorem 6.3.

Suppose that  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is the strongly convex quadratic function  $f(x) = b^T x + \frac{1}{2} x^T A x$ , where  $A$  is symmetric and positive definite. Let  $x_0$  be any starting point for the iteration (6.34) and  $B_0$  be any symmetric positive definite starting matrix, and suppose that the matrices  $B_k$  are updated by the Broyden formula (6.32) with  $\phi_k \in [0, 1]$ . Define  $\lambda_1^k \leq \lambda_2^k \leq \dots \leq \lambda_n^k$  to be the eigenvalues of the matrix

$$A^{\frac{1}{2}} B_k^{-1} A^{\frac{1}{2}}. \quad (6.35)$$

Then for all  $k$ , we have

$$\min\{\lambda_i^k, 1\} \leq \lambda_i^{k+1} \leq \max\{\lambda_i^k, 1\}, \quad i = 1, 2, \dots, n. \quad (6.36)$$

Moreover, the property (6.36) does not hold if the Broyden parameter  $\phi_k$  is chosen outside the interval  $[0, 1]$ .

Let us discuss the significance of this result. If the eigenvalues  $\lambda_i^k$  of the matrix (6.35) are all 1, then the quasi-Newton approximation  $B_k$  is identical to the Hessian  $A$  of the quadratic objective function. This situation is the ideal one, so we should be hoping for these eigenvalues to be as close to 1 as possible. In fact, relation (6.36) tells us that the



eigenvalues  $\{\lambda_i^k\}$  converge monotonically (but not strictly monotonically) to 1. Suppose, for example, that at iteration  $k$  the smallest eigenvalue is  $\lambda_1^k = 0.7$ . Then (6.36) tells us that at the next iteration  $\lambda_1^{k+1} \in [0.7, 1]$ . We cannot be sure that this eigenvalue has actually moved closer to 1, but it is reasonable to expect that it has. In contrast, the first eigenvalue can become smaller than 0.7 if we allow  $\phi_k$  to be outside  $[0, 1]$ . Significantly, the result of Theorem 6.3 holds even if the line searches are not exact.

Although Theorem 6.3 seems to suggest that the best update formulas belong to the restricted Broyden class, the situation is not at all clear. Some analysis and computational testing suggest that algorithms that allow  $\phi_k$  to be negative (in a strictly controlled manner) may in fact be superior to the BFGS method. The SR1 formula is a case in point: It is a member of the Broyden class, obtained by setting

$$\phi_k = \frac{s_k^T y_k}{s_k^T y_k - s_k^T B_k s_k},$$

but it does not belong to the restricted Broyden class, because this value of  $\phi_k$  may fall outside the interval  $[0, 1]$ .

In the remaining discussion of this section, we determine more precisely the range of values of  $\phi_k$  that preserve positive definiteness.

The last term in (6.32) is a rank-one correction, which by the interlacing eigenvalue theorem (Theorem A.1) increases the eigenvalues of the matrix when  $\phi_k$  is positive. Therefore  $B_{k+1}$  is positive definite for all  $\phi_k \geq 0$ . On the other hand, by Theorem A.1 the last term in (6.32) decreases the eigenvalues of the matrix when  $\phi_k$  is negative. As we decrease  $\phi_k$ , this matrix eventually becomes singular and then indefinite. A little computation shows that  $B_{k+1}$  is singular when  $\phi_k$  has the value

$$\phi_k^c = \frac{1}{1 - \mu_k}, \quad (6.37)$$

where

$$\mu_k = \frac{(y_k^T B_k^{-1} y_k)(s_k^T B_k s_k)}{(y_k^T s_k)^2}. \quad (6.38)$$

By applying the Cauchy–Schwarz inequality (A.5) to (6.38), we see that  $\mu_k \geq 1$  and therefore  $\phi_k^c \leq 0$ . Hence, if the initial Hessian approximation  $B_0$  is symmetric and positive definite, and if  $s_k^T y_k > 0$  and  $\phi_k > \phi_k^c$  for each  $k$ , then all the matrices  $B_k$  generated by Broyden's formula (6.32) remain symmetric and positive definite.

When the line search is exact, *all* methods in the Broyden class with  $\phi_k \geq \phi_k^c$  generate the same sequence of iterates. This result applies to general nonlinear functions and is based on the observation that when all the line searches are exact, the directions generated by Broyden-class methods differ only in their lengths. The line searches identify the same

minima along the chosen search direction, though the values of the step lengths may differ because of the different scaling.

The Broyden class has several remarkable properties when applied with exact line searches to quadratic functions. We state some of these properties in the next theorem, whose proof is omitted.

**Theorem 6.4.**

*Suppose that a method in the Broyden class is applied to the strongly convex quadratic function  $f(x) = b^T x + \frac{1}{2}x^T A x$ , where  $x_0$  is the starting point and  $B_0$  is any symmetric positive definite matrix. Assume that  $\alpha_k$  is the exact step length and that  $\phi_k \geq \phi_k^c$  for all  $k$ , where  $\phi_k^c$  is defined by (6.37). Then the following statements are true.*

- (i) *The iterates are independent of  $\phi_k$  and converge to the solution in at most  $n$  iterations.*
- (ii) *The secant equation is satisfied for all previous search directions, that is,*

$$B_k s_j = y_j, \quad j = k - 1, k - 2, \dots, 1.$$

- (iii) *If the starting matrix is  $B_0 = I$ , then the iterates are identical to those generated by the conjugate gradient method (see Chapter 5). In particular, the search directions are conjugate, that is,*

$$s_i^T A s_j = 0, \quad \text{for } i \neq j.$$

- (iv) *If  $n$  iterations are performed, we have  $B_n = A$ .*

Note that parts (i), (ii), and (iv) of this result echo the statement and proof of Theorem 6.1, where similar results were derived for the SR1 update formula.

We can generalize Theorem 6.4 slightly: It continues to hold if the Hessian approximations remain nonsingular but not necessarily positive definite. (Hence, we could allow  $\phi_k$  to be smaller than  $\phi_k^c$ , provided that the chosen value did not produce a singular updated matrix.) We can also generalize point (iii) as follows. If the starting matrix  $B_0$  is not the identity matrix, then the Broyden-class method is identical to the preconditioned conjugate gradient method that uses  $B_0$  as preconditioner.

We conclude by commenting that results like Theorem 6.4 would appear to be of mainly theoretical interest, since the inexact line searches used in practical implementations of Broyden-class methods (and all other quasi-Newton methods) cause their performance to differ markedly. Nevertheless, it is worth noting that this type of analysis guided much of the development of quasi-Newton methods.

## 6.4 CONVERGENCE ANALYSIS

In this section we present global and local convergence results for practical implementations of the BFGS and SR1 methods. We give more details for BFGS because its analysis is more general and illuminating than that of SR1. The fact that the Hessian approximations evolve by means of updating formulas makes the analysis of quasi-Newton methods much more complex than that of steepest descent and Newton's method.

Although the BFGS and SR1 methods are known to be remarkably robust in practice, we will not be able to establish truly global convergence results for general nonlinear objective functions. That is, we cannot prove that the iterates of these quasi-Newton methods approach a stationary point of the problem from any starting point and any (suitable) initial Hessian approximation. In fact, it is not yet known if the algorithms enjoy such properties. In our analysis we will either assume that the objective function is convex or that the iterates satisfy certain properties. On the other hand, there are well known local, superlinear convergence results that are true under reasonable assumptions.

Throughout this section we use  $\|\cdot\|$  to denote the Euclidean vector or matrix norm, and denote the Hessian matrix  $\nabla^2 f(x)$  by  $G(x)$ .

### GLOBAL CONVERGENCE OF THE BFGS METHOD

We study the global convergence of the BFGS method, with a practical line search, when applied to a smooth convex function from an arbitrary starting point  $x_0$  and from any initial Hessian approximation  $B_0$  that is symmetric and positive definite. We state our precise assumptions about the objective function formally, as follows.

#### Assumption 6.1.

- (i) *The objective function  $f$  is twice continuously differentiable.*
- (ii) *The level set  $\mathcal{L} = \{x \in \mathbb{R}^n \mid f(x) \leq f(x_0)\}$  is convex, and there exist positive constants  $m$  and  $M$  such that*

$$m\|z\|^2 \leq z^T G(x)z \leq M\|z\|^2 \quad (6.39)$$

for all  $z \in \mathbb{R}^n$  and  $x \in \mathcal{L}$ .

Part (ii) of this assumption implies that  $G(x)$  is positive definite on  $\mathcal{L}$  and that  $f$  has a unique minimizer  $x^*$  in  $\mathcal{L}$ .

By using (6.12) and (6.39) we obtain

$$\frac{y_k^T s_k}{s_k^T s_k} = \frac{s_k^T \bar{G}_k s_k}{s_k^T s_k} \geq m, \quad (6.40)$$

where  $\bar{G}_k$  is the average Hessian defined in (6.11). Assumption 6.1 implies that  $\bar{G}_k$  is positive definite, so its square root is well-defined. Therefore, as in (6.21), we have by defining  $z_k = \bar{G}_k^{1/2} s_k$  that

$$\frac{y_k^T y_k}{y_k^T s_k} = \frac{s_k^T \bar{G}_k^2 s_k}{s_k^T \bar{G}_k s_k} = \frac{z_k^T \bar{G}_k z_k}{z_k^T z_k} \leq M. \quad (6.41)$$

We are now ready to present the global convergence result for the BFGS method. It does not seem to be possible to establish a bound on the condition number of the Hessian approximations  $B_k$ , as is done in Section 3.2. Instead, we will introduce two new tools in the analysis, the trace and determinant, to estimate the size of the largest and smallest eigenvalues of the Hessian approximations. The trace of a matrix (denoted by  $\text{trace}(\cdot)$ ) is the sum of its eigenvalues, while the determinant (denoted by  $\det(\cdot)$ ) is the product of the eigenvalues; see the Appendix for a brief discussion of their properties.

**Theorem 6.5.**

*Let  $B_0$  be any symmetric positive definite initial matrix, and let  $x_0$  be a starting point for which Assumption 6.1 is satisfied. Then the sequence  $\{x_k\}$  generated by Algorithm 6.1 (with  $\epsilon = 0$ ) converges to the minimizer  $x^*$  of  $f$ .*

PROOF. We start by defining

$$m_k = \frac{y_k^T s_k}{s_k^T s_k}, \quad M_k = \frac{y_k^T y_k}{y_k^T s_k}, \quad (6.42)$$

and note from (6.40) and (6.41) that

$$m_k \geq m, \quad M_k \leq M. \quad (6.43)$$

By computing the trace of the BFGS update (6.19), we obtain that

$$\text{trace}(B_{k+1}) = \text{trace}(B_k) - \frac{\|B_k s_k\|^2}{s_k^T B_k s_k} + \frac{\|y_k\|^2}{y_k^T s_k} \quad (6.44)$$

(see Exercise 6.11). We can also show (Exercise 6.10) that

$$\det(B_{k+1}) = \det(B_k) \frac{y_k^T s_k}{s_k^T B_k s_k}. \quad (6.45)$$

We now define

$$\cos \theta_k = \frac{s_k^T B_k s_k}{\|s_k\| \|B_k s_k\|}, \quad q_k = \frac{s_k^T B_k s_k}{s_k^T s_k}, \quad (6.46)$$

so that  $\theta_k$  is the angle between  $s_k$  and  $B_k s_k$ . We then obtain that

$$\frac{\|B_k s_k\|^2}{s_k^T B_k s_k} = \frac{\|B_k s_k\|^2 \|s_k\|^2 s_k^T B_k s_k}{(s_k^T B_k s_k)^2 \|s_k\|^2} = \frac{q_k}{\cos^2 \theta_k}. \quad (6.47)$$

In addition, we have from (6.42) that

$$\det(B_{k+1}) = \det(B_k) \frac{y_k^T s_k}{s_k^T s_k} \frac{s_k^T s_k}{s_k^T B_k s_k} = \det(B_k) \frac{m_k}{q_k}. \quad (6.48)$$

We now combine the trace and determinant by introducing the following function of a positive definite matrix  $B$ :

$$\psi(B) = \text{trace}(B) - \ln(\det(B)), \quad (6.49)$$

where  $\ln(\cdot)$  denotes the natural logarithm. It is not difficult to show that  $\psi(B) > 0$ ; see Exercise 6.9. By using (6.42) and (6.44)–(6.49), we have that

$$\begin{aligned} \psi(B_{k+1}) &= \text{trace}(B_k) + M_k - \frac{q_k}{\cos^2 \theta_k} - \ln(\det(B_k)) - \ln m_k + \ln q_k \\ &= \psi(B_k) + (M_k - \ln m_k - 1) \\ &\quad + \left[ 1 - \frac{q_k}{\cos^2 \theta_k} + \ln \frac{q_k}{\cos^2 \theta_k} \right] + \ln \cos^2 \theta_k. \end{aligned} \quad (6.50)$$

Now, since the function  $h(t) = 1 - t + \ln t$  is nonpositive for all  $t > 0$  (see Exercise 6.8), the term inside the square brackets is nonpositive, and thus from (6.43) and (6.50) we have

$$0 < \psi(B_{k+1}) \leq \psi(B_0) + c(k+1) + \sum_{j=0}^k \ln \cos^2 \theta_j, \quad (6.51)$$

where we can assume the constant  $c = M - \ln m - 1$  to be positive, without loss of generality.

We now relate these expressions to the results given in Section 3.2. Note from the form  $s_k = -\alpha_k B_k^{-1} \nabla f_k$  of the quasi-Newton iteration that  $\cos \theta_k$  defined by (6.46) is the angle between the steepest descent direction and the search direction, which plays a crucial role in the global convergence theory of Chapter 3. From (3.22), (3.23) we know that the sequence  $\|\nabla f_k\|$  generated by the line search algorithm is bounded away from zero only if  $\cos \theta_j \rightarrow 0$ .

Let us then proceed by contradiction and assume that  $\cos \theta_j \rightarrow 0$ . Then there exists  $k_1 > 0$  such that for all  $j > k_1$ , we have

$$\ln \cos^2 \theta_j < -2c,$$

where  $c$  is the constant defined above. Using this inequality in (6.51) we find the following relations to be true for all  $k > k_1$ :

$$\begin{aligned} 0 &< \psi(B_0) + c(k+1) + \sum_{j=0}^{k_1} \ln \cos^2 \theta_j + \sum_{j=k_1+1}^k (-2c) \\ &= \psi(B_0) + \sum_{j=0}^{k_1} \ln \cos^2 \theta_j + 2ck_1 + c - ck. \end{aligned}$$

However, the right-hand-side is negative for large  $k$ , giving a contradiction. Therefore, there exists a subsequence of indices  $\{j_k\}_{k=1,2,\dots}$  such that  $\cos \theta_{j_k} \geq \delta > 0$ . By Zoutendijk's result (3.14) this limit implies that  $\liminf \|\nabla f_k\| \rightarrow 0$ . Since the problem is strongly convex, the latter limit is enough to prove that  $x_k \rightarrow x^*$ .  $\square$

Theorem 6.5 has been generalized to the entire restricted Broyden class, *except for the DFP method*. In other words, Theorem 6.5 can be shown to hold for all  $\phi_k \in [0, 1)$  in (6.32), but the argument seems to break down as  $\phi_k$  approaches 1 because some of the self-correcting properties of the update are weakened considerably.

An extension of the analysis just given shows that the rate of convergence of the iterates is linear. In particular, we can show that the sequence  $\|x_k - x^*\|$  converges to zero rapidly enough that

$$\sum_{k=1}^{\infty} \|x_k - x^*\| < \infty. \quad (6.52)$$

We will not prove this claim, but rather establish that if (6.52) holds, then the rate of convergence is actually superlinear.

### SUPERLINEAR CONVERGENCE OF THE BFGS METHOD

The analysis of this section makes use of the Dennis and Moré characterization (3.36) of superlinear convergence. It applies to general nonlinear—not just convex—objective functions. For the results that follow we need to make an additional assumption.

#### Assumption 6.2.

*The Hessian matrix  $G$  is Lipschitz continuous at  $x^*$ , that is,*

$$\|G(x) - G(x^*)\| \leq L\|x - x^*\|,$$

*for all  $x$  near  $x^*$ , where  $L$  is a positive constant.*

We start by introducing the quantities

$$\tilde{s}_k = G_*^{1/2} s_k, \quad \tilde{y}_k = G_*^{-1/2} y_k, \quad \tilde{B}_k = G_*^{-1/2} B_k G_*^{-1/2},$$

where  $G_* = G(x^*)$  and  $x^*$  is a minimizer of  $f$ . Similarly to (6.46), we define

$$\cos \tilde{\theta}_k = \frac{\tilde{s}_k^T \tilde{B}_k \tilde{s}_k}{\|\tilde{s}_k\| \|\tilde{B}_k \tilde{s}_k\|}, \quad \tilde{q}_k = \frac{\tilde{s}_k^T \tilde{B}_k \tilde{s}_k}{\|\tilde{s}_k\|^2},$$

while we echo (6.42) and (6.43) in defining

$$\tilde{M}_k = \frac{\|\tilde{y}_k\|^2}{\tilde{y}_k^T \tilde{s}_k}, \quad \tilde{m}_k = \frac{\tilde{y}_k^T \tilde{s}_k}{\tilde{s}_k^T \tilde{s}_k}.$$

By pre- and postmultiplying the BFGS update formula (6.19) by  $G_*^{-1/2}$  and grouping terms appropriately, we obtain

$$\tilde{B}_{k+1} = \tilde{B}_k - \frac{\tilde{B}_k \tilde{s}_k \tilde{s}_k^T \tilde{B}_k}{\tilde{s}_k^T \tilde{B}_k \tilde{s}_k} + \frac{\tilde{y}_k \tilde{y}_k^T}{\tilde{y}_k^T \tilde{s}_k}.$$

Since this expression has precisely the same form as the BFGS formula (6.19), it follows from the argument leading to (6.50) that

$$\begin{aligned} \psi(\tilde{B}_{k+1}) &= \psi(\tilde{B}_k) + (\tilde{M}_k - \ln \tilde{m}_k - 1) \\ &\quad + \left[ 1 - \frac{\tilde{q}_k}{\cos^2 \tilde{\theta}_k} + \ln \frac{\tilde{q}_k}{\cos^2 \tilde{\theta}_k} \right] \\ &\quad + \ln \cos^2 \tilde{\theta}_k. \end{aligned} \tag{6.53}$$

Recalling (6.12), we have that

$$y_k - G_* s_k = (\tilde{G}_k - G_*) s_k,$$

and thus

$$\tilde{y}_k - \tilde{s}_k = G_*^{-1/2} (\tilde{G}_k - G_*) G_*^{-1/2} \tilde{s}_k.$$

By Assumption 6.2, and recalling the definition (6.11), we have

$$\|\tilde{y}_k - \tilde{s}_k\| \leq \|G_*^{-1/2}\|^2 \|\tilde{s}_k\| \|\tilde{G}_k - G_*\| \leq \|G_*^{-1/2}\|^2 \|\tilde{s}_k\| L \epsilon_k,$$

where  $\epsilon_k$  is defined by

$$\epsilon_k = \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\}.$$

We have thus shown that

$$\frac{\|\tilde{y}_k - \tilde{s}_k\|}{\|\tilde{s}_k\|} \leq \bar{c}\epsilon_k, \quad (6.54)$$

for some positive constant  $\bar{c}$ . This inequality and (6.52) play an important role in superlinear convergence, as we now show.

**Theorem 6.6.**

*Suppose that  $f$  is twice continuously differentiable and that the iterates generated by the BFGS algorithm converge to a minimizer  $x^*$  at which Assumption 6.2 holds. Suppose also that (6.52) holds. Then  $x_k$  converges to  $x^*$  at a superlinear rate.*

PROOF. From (6.54), we have from the triangle inequality (A.4a) that

$$\|\tilde{y}_k\| - \|\tilde{s}_k\| \leq \bar{c}\epsilon_k \|\tilde{s}_k\|, \quad \|\tilde{s}_k\| - \|\tilde{y}_k\| \leq \bar{c}\epsilon_k \|\tilde{s}_k\|,$$

so that

$$(1 - \bar{c}\epsilon_k)\|\tilde{s}_k\| \leq \|\tilde{y}_k\| \leq (1 + \bar{c}\epsilon_k)\|\tilde{s}_k\|. \quad (6.55)$$

By squaring (6.54) and using (6.55), we obtain

$$(1 - \bar{c}\epsilon_k)^2 \|\tilde{s}_k\|^2 - 2\tilde{y}_k^T \tilde{s}_k + \|\tilde{s}_k\|^2 \leq \|\tilde{y}_k\|^2 - 2\tilde{y}_k^T \tilde{s}_k + \|\tilde{s}_k\|^2 \leq \bar{c}^2 \epsilon_k^2 \|\tilde{s}_k\|^2,$$

and therefore

$$2\tilde{y}_k^T \tilde{s}_k \geq (1 - 2\bar{c}\epsilon_k + \bar{c}^2 \epsilon_k^2 + 1 - \bar{c}^2 \epsilon_k^2) \|\tilde{s}_k\|^2 = 2(1 - \bar{c}\epsilon_k) \|\tilde{s}_k\|^2.$$

It follows from the definition of  $\tilde{m}_k$  that

$$\tilde{m}_k = \frac{\tilde{y}_k^T \tilde{s}_k}{\|\tilde{s}_k\|^2} \geq 1 - \bar{c}\epsilon_k. \quad (6.56)$$

By combining (6.55) and (6.56), we obtain also that

$$\tilde{M}_k = \frac{\|\tilde{y}_k\|^2}{\tilde{y}_k^T \tilde{s}_k} \leq \frac{1 + \bar{c}\epsilon_k}{1 - \bar{c}\epsilon_k}. \quad (6.57)$$

Since  $x_k \rightarrow x^*$ , we have that  $\epsilon_k \rightarrow 0$ , and thus by (6.57) there exists a positive constant  $c > \bar{c}$  such that the following inequalities hold for all sufficiently large  $k$ :

$$\tilde{M}_k \leq 1 + \frac{2\bar{c}}{1 - \bar{c}\epsilon_k} \epsilon_k \leq 1 + c\epsilon_k. \quad (6.58)$$



We again make use of the nonpositiveness of the function  $h(t) = 1 - t + \ln t$ . Therefore, we have

$$\frac{-x}{1-x} - \ln(1-x) = h\left(\frac{1}{1-x}\right) \leq 0.$$

Now, for  $k$  large enough we can assume that  $\bar{c}\epsilon_k < \frac{1}{2}$ , and therefore

$$\ln(1 - \bar{c}\epsilon_k) \geq \frac{-\bar{c}\epsilon_k}{1 - \bar{c}\epsilon_k} \geq -2\bar{c}\epsilon_k.$$

This relation and (6.56) imply that for sufficiently large  $k$ , we have

$$\ln \tilde{m}_k \geq \ln(1 - \bar{c}\epsilon_k) \geq -2\bar{c}\epsilon_k > -2c\epsilon_k. \quad (6.59)$$

We can now deduce from (6.53), (6.58), and (6.59) that

$$0 < \psi(\tilde{B}_{k+1}) \leq \psi(\tilde{B}_k) + 3c\epsilon_k + \ln \cos^2 \tilde{\theta}_k + \left[1 - \frac{\tilde{q}_k}{\cos^2 \tilde{\theta}_k} + \ln \frac{\tilde{q}_k}{\cos^2 \tilde{\theta}_k}\right]. \quad (6.60)$$

By summing this expression and making use of (6.52) we have that

$$\sum_{j=0}^{\infty} \left( \ln \frac{1}{\cos^2 \tilde{\theta}_j} - \left[1 - \frac{\tilde{q}_j}{\cos^2 \tilde{\theta}_j} + \ln \frac{\tilde{q}_j}{\cos^2 \tilde{\theta}_j}\right] \right) \leq \psi(\tilde{B}_0) + 3c \sum_{j=0}^{\infty} \epsilon_j < +\infty.$$

Since the term in the square brackets is nonpositive, and since  $\ln(1/\cos^2 \tilde{\theta}_j) \geq 0$  for all  $j$ , we obtain the two limits

$$\lim_{j \rightarrow \infty} \ln \frac{1}{\cos^2 \tilde{\theta}_j} = 0, \quad \lim_{j \rightarrow \infty} \left(1 - \frac{\tilde{q}_j}{\cos^2 \tilde{\theta}_j} + \ln \frac{\tilde{q}_j}{\cos^2 \tilde{\theta}_j}\right) = 0,$$

which imply that

$$\lim_{j \rightarrow \infty} \cos \tilde{\theta}_j = 1, \quad \lim_{j \rightarrow \infty} \tilde{q}_j = 1. \quad (6.61)$$

The essence of the result has now been proven; we need only to interpret these limits in terms of the Dennis–Moré characterization of superlinear convergence.

Recalling (6.47), we have

$$\begin{aligned}
 \frac{\|G_*^{-1/2}(B_k - G_*)s_k\|^2}{\|G_*^{1/2}s_k\|^2} &= \frac{\|(\tilde{B}_k - I)\tilde{s}_k\|^2}{\|\tilde{s}_k\|^2} \\
 &= \frac{\|\tilde{B}_k\tilde{s}_k\|^2 - 2\tilde{s}_k^T\tilde{B}_k\tilde{s}_k + \tilde{s}_k^T\tilde{s}_k}{\tilde{s}_k^T\tilde{s}_k} \\
 &= \frac{\tilde{q}_k^2}{\cos^2\tilde{\theta}_k} - 2\tilde{q}_k + 1.
 \end{aligned}$$

Since by (6.61) the right-hand-side converges to 0, we conclude that

$$\lim_{k \rightarrow \infty} \frac{\|(B_k - G_*)s_k\|}{\|s_k\|} = 0.$$

The limit (3.36) and Theorem 3.6 imply that the unit step length  $\alpha_k = 1$  will satisfy the Wolfe conditions near the solution, and hence that the rate of convergence is superlinear.  $\square$

### CONVERGENCE ANALYSIS OF THE SR1 METHOD

The convergence properties of the SR1 method are not as well understood as those of the BFGS method. No global results like Theorem 6.5 or local superlinear results like Theorem 6.6 have been established, except the results for quadratic functions discussed earlier. There is, however, an interesting result for the trust-region SR1 algorithm, Algorithm 6.2. It states that when the objective function has a unique stationary point and the condition (6.26) holds at every step (so that the SR1 update is never skipped) and the Hessian approximations  $B_k$  are bounded above, then the iterates converge to  $x^*$  at an  $(n + 1)$ -step superlinear rate. The result does not require exact solution of the trust-region subproblem (6.27).

We state the result formally as follows.

#### Theorem 6.7.

Suppose that the iterates  $x_k$  are generated by Algorithm 6.2. Suppose also that the following conditions hold:

- (c1) The sequence of iterates does not terminate, but remains in a closed, bounded, convex set  $D$ , on which the function  $f$  is twice continuously differentiable, and in which  $f$  has a unique stationary point  $x^*$ ;
- (c2) the Hessian  $\nabla^2 f(x^*)$  is positive definite, and  $\nabla^2 f(x)$  is Lipschitz continuous in a neighborhood of  $x^*$ ;
- (c3) the sequence of matrices  $\{B_k\}$  is bounded in norm;
- (c4) condition (6.26) holds at every iteration, where  $r$  is some constant in  $(0, 1)$ .

Then  $\lim_{k \rightarrow \infty} x_k = x^*$ , and we have that

$$\lim_{k \rightarrow \infty} \frac{\|x_{k+n+1} - x^*\|}{\|x_k - x^*\|} = 0.$$

Note that the BFGS method does not require the boundedness assumption (c3) to hold. As we have mentioned already, the SR1 update does not necessarily maintain positive definiteness of the Hessian approximations  $B_k$ . In practice,  $B_k$  may be indefinite at any iteration, which means that the trust region bound may continue to be active for arbitrarily large  $k$ . Interestingly, however, it can be shown that the SR1 Hessian approximations tend to be positive definite most of the time. The precise result is that

$$\lim_{k \rightarrow \infty} \frac{\text{number of indices } j = 1, 2, \dots, k \text{ for which } B_j \text{ is positive semidefinite}}{k} = 1,$$

under the assumptions of Theorem 6.7. This result holds regardless of whether the initial Hessian approximation is positive definite or not.

## NOTES AND REFERENCES

For a comprehensive treatment of quasi-Newton methods see Dennis and Schnabel [92], Dennis and Moré [91], and Fletcher [101]. A formula for updating the Cholesky factors of the BFGS matrices is given in Dennis and Schnabel [92].

Several safeguards and modifications of the SR1 method have been proposed, but the condition (6.26) is favored in the light of the analysis of Conn, Gould, and Toint [71]. Computational experiments by Conn, Gould, and Toint [70, 73] and Khalfan, Byrd, and Schnabel [181], using both line search and trust-region approaches, indicate that the SR1 method appears to be competitive with the BFGS method. The proof of Theorem 6.7 is given in Byrd, Khalfan, and Schnabel [51].


A study of the convergence of BFGS matrices for nonlinear problems can be found in Ge and Powell [119] and Boggs and Tolle [32]; however, the results are not as satisfactory as for SR1 updating.


The global convergence of the BFGS method was established by Powell [246]. This result was extended to the restricted Broyden class, except for DFP, by Byrd, Nocedal, and Yuan [53]. For a discussion of the self-correcting properties of quasi-Newton methods see Nocedal [229]. Most of the early analysis of quasi-Newton methods was based on the *bounded deterioration* principle. This is a tool for the local analysis that quantifies the worst-case behavior of quasi-Newton updating. Assuming that the starting point is sufficiently close to the solution  $x^*$  and that the initial Hessian approximation is sufficiently close to  $\nabla^2 f(x^*)$ , one can use the bounded deterioration bounds to prove that the iteration cannot stray away from the solution. This property can then be used to show that the quality of the quasi-Newton approximations is good enough to yield superlinear convergence. For details, see Dennis and Moré [91] or Dennis and Schnabel [92].


---


 **EXERCISES**
 **6.1**


- (a) Show that if  $f$  is strongly convex, then (6.7) holds for any vectors  $x_k$  and  $x_{k+1}$ .
- (b) Give an example of a function of one variable satisfying  $g(0) = -1$  and  $g(1) = -\frac{1}{4}$  and show that (6.7) does not hold in this case.


 **6.2** Show that the second strong Wolfe condition (3.7b) implies the curvature condition (6.7).


 **6.3** Verify that (6.19) and (6.17) are inverses of each other.


 **6.4** Use the Sherman–Morrison formula (A.27) to show that (6.24) is the inverse of (6.25).

 **6.5** Prove the statements (ii) and (iii) given in the paragraph following (6.25).

 **6.6** The square root of a matrix  $A$  is a matrix  $A^{1/2}$  such that  $A^{1/2}A^{1/2} = A$ . Show that any symmetric positive definite matrix  $A$  has a square root, and that this square root is itself symmetric and positive definite. (Hint: Use the factorization  $A = UDU^T$  (A.16), where  $U$  is orthogonal and  $D$  is diagonal with positive diagonal elements.)


 **6.7** Use the Cauchy–Schwarz inequality (A.5) to verify that  $\mu_k \geq 1$ , where  $\mu_k$  is defined by (6.38).

 **6.8** Define  $h(t) = 1 - t + \ln t$ , and note that  $h'(t) = -1 + 1/t$ ,  $h''(t) = -1/t^2 < 0$ ,  $h(1) = 0$ , and  $h'(1) = 0$ . Show that  $h(t) \leq 0$  for all  $t > 0$ .

 **6.9** Denote the eigenvalues of the positive definite matrix  $B$  by  $\lambda_1, \lambda_2, \dots, \lambda_n$ , where  $0 < \lambda_1 \leq \lambda_2 \leq \dots \leq \lambda_n$ . Show that the  $\psi$  function defined in (6.49) can be written as

$$\psi(B) = \sum_{i=1}^n (\lambda_i - \ln \lambda_i).$$

Use this form to show that  $\psi(B) > 0$ .

 **6.10** The object of this exercise is to prove (6.45).

- (a) Show that  $\det(I + xy^T) = 1 + y^T x$ , where  $x$  and  $y$  are  $n$ -vectors. Hint: Assuming that  $x \neq 0$ , we can find vectors  $w_1, w_2, \dots, w_{n-1}$  such that the matrix  $Q$  defined by

$$Q = [x, w_1, w_2, \dots, w_{n-1}]$$

is nonsingular and  $x = Qe_1$ , where  $e_1 = (1, 0, 0, \dots, 0)^T$ . If we define

$$y^T Q = (z_1, z_2, \dots, z_n),$$

then

$$z_1 = y^T Qe_1 = y^T Q(Q^{-1}x) = y^T x,$$


and


$$\det(I + xy^T) = \det(Q^{-1}(I + xy^T)Q) = \det(I + e_1y^T Q).$$

(b) Use a similar technique to prove that

$$\det(I + xy^T + uv^T) = (1 + y^T x)(1 + v^T u) - (x^T v)(y^T u).$$

(c) Use this relation to establish (6.45).

 **6.11** Use the properties of the trace of a symmetric matrix and the formula (6.19) to prove (6.44).

 **6.12** Show that if  $f$  satisfies Assumption 6.1 and if the sequence of gradients satisfies  $\liminf \|\nabla f_k\| = 0$ , then the whole sequence of iterates  $x$  converges to the solution  $x^*$ .