A Theoretical and Experimental Study Of the Symmetric Rank One Update *

H. Khalfan
R. H. Byrd
R. B. Schnabel

CU-CS-489-90

University of Colorado at Boulder
DEPARTMENT OF COMPUTER SCIENCE

* Supported in part by NSF Grant GJ-36461.

A Theoretical and Experimental Study of the Symmetric Rank One Update[1]

H. Khalfan[2], R. H. Byrd[3], and R. B. Schnabel[3]

CU-CS-489-90 December 1990

University of Colorado at Boulder
Technical Report CU-CS-489-90
Department of Computer Science
Campus Box 430
University of Colorado
Boulder, Colorado 80309-0430

[1]Research supported by NSF grants CCR-8702403 and CCR-8920519, AFOSR grant AFOSR-90-0109, and ARO contract DAAL 03-88-K-0086.

[2]Department of Mathematics, United Arab Emirates University

[3]Department of Computer Science, Campus Box 430, University of Colorado, Boulder, Colorado 80309 U.S.A.

A Theoretical and Experimental Study of the Symmetric Rank One Update

H. Khalfan   R. H. Byrd   R. B. Schnabel

December 1990

Abstract

In this paper, we first discuss computational experience using the SR1 update in conventional line search and trust region algorithms for unconstrained optimization. Our experiments show that the SR1 is very competitive with the widely used BFGS method. They also indicate two interesting features: the final Hessian approximations produced by the SR1 method are not generally appreciably better than those produced by the BFGS, and the sequences of steps produced by the SR1 do not usually seem to have the "uniform linear independence" property that is assumed in some recent convergence analysis. We present a new analysis that shows that the SR1 method with a line search is n+1 step q-superlinearly convergent without the assumption of linearly independent iterates. This analysis assumes that the Hessian approximations are positive definite and bounded asymptotically, which from our computational experience are reasonable assumptions.

1. Introduction

This paper is concerned with secant (quasi-Newton) methods for finding a local minimum of the unconstrained optimization problem

$$\min_{x \in R^n} f(x). \tag{1.1}$$

It will be assumed that $f(x)$ is at least twice continuously differentiable, and that the number of variables n is sufficiently small to permit storage of an $n \times n$ matrix, and $O(n^2)$ or possibly $O(n^3)$ arithmetic operations per iteration.

Algorithms for solving (1.1) are iterative, and the basic framework of an iteration of a secant method is:

> Given current iterate x_c, $f(x_c)$, $\nabla f(x_c)$ or finite difference approximation, and $B_c \in R^{n \times n}$ symmetric (secant approximation to $\nabla^2 f(x_c)$):
>
> > Select new iterate x_+ by a line search or trust region method based on quadratic model $m(x_c + d) = f(x_c) + \nabla f(x_c)^T d + \frac{1}{2} d^T B_c d$.
> >
> > Update B_c to B_+ such that B_+ is symmetric and satisfies the secant equation $B_+ s_c = y_c$, where $s_c = x_+ - x_c$ and $y_c = \nabla f(x_+) - \nabla f(x_c)$.

In this paper, we consider the SR1 update for the Hessian approximation,

$$B_+ = B_c + \frac{(y_c - B_c s_c)(y_c - B_c s_c)^T}{s_c^T (y_c - B_c s_c)} \tag{1.2}$$

and, for purpose of comparison, the BFGS update

$$B_+ = B_c + \frac{y_c y_c^T}{y_c^T y_c} + \frac{B_c s_c s_c^T B_c}{s_c^T y_c}. \tag{1.3}$$

For background on these updates and others see Fletcher [1980], Gill, Murray, and Wright [1981], and Dennis and Schnabel [1983].

The BFGS update has been the most commonly used secant update for many years. It makes a symmetric, rank two change to the previous Hessian approximation B_c, and if B_c is positive definite and $s_c^T y_c > 0$, then B_+ is positive definite.

The BFGS method has been shown by Broyden, Dennis, and Moré [1973] to be locally q-superlinearly convergent provided that the initial Hessian approximation is sufficiently accurate. Powell [1976] proved a global superlinear convergence result for the BFGS method when applied to strictly convex functions and used in conjunction with line searches that satisfy the conditions of Wolfe [1968]. The BFGS update has been used successfully in many production codes for unconstrained optimization.

The SR1 formula, on the other hand, makes a symmetric rank one change to the previous Hessian approximation B_c. Compared with other secant updates, the SR1 update is simpler and may require less computation per iteration when unfactored forms of updates are used. (Factored updates are those in which a decomposition of B_c is updated at each iteration.) A basic disadvantage to the SR1 update, however, is the fact that its denominator may be zero or nearly zero, which causes numerical instability. A simple remedy to this problem is to set $B_+ = B_c$ whenever this difficulty arises, but this may prevent fast convergence. Another problem is that the SR1 update may not preserve positive definiteness even if this is possible, i.e., when B_c is positive definite and $s_c^T y_c > 0$.

Fiacco and McCormick [1968] showed that if the SR1 update is applied to a positive definite quadratic function in a line search method, then, provided that the updates are all well defined, the solution is reached in at most $n+1$ iterations. Furthermore, if $n+1$ iterations are required, then the final Hessian approximation is the actual Hessian at the solution. This result is not true, in general, for the BFGS update or other members of the Broyden family, unless exact line searches are used.

For nonquadratic functions, however, convergence of the SR1 is not as well understood as convergence of the BFGS method. In fact, Broyden, Dennis, and Moré [1973] have shown that under their assumptions the SR1 update can be undefined, and thus their convergence analysis can not be applied in this case. Also, no global convergence result similar to that for the BFGS method given by Powell [1976] exists, so far, for the SR1 method when applied to a non-quadratic function.

Recent work by Conn, Gould, and Toint [1988a, 1988b, 1991] has sparked renewed interest in the SR1 update. Conn, Gould, and Toint [1991] proved that the sequence of matrices generated by the SR1 formula converges to the actual Hessian at the solution $\nabla^2 f(x_*)$, provided that the steps taken are uniformly linearly independent, that the SR1 update denominator is always sufficiently different from zero, and that the iterates converge to a finite limit. (Using this result it is simple to prove that the rate of convergence is q-superlinear.) On the other hand, for the BFGS method Ge and Powell [1983] proved, under a different set of assumptions, that the sequence of generated matrices converges but not necessarily to $\nabla^2 f(x_*)$.

The numerical experiments of Conn, Gould, and Toint [1988b] indicate that minimization algorithms based on the SR1 update may be competitive computationally with methods using the BFGS formula. The algorithm used by Conn, Gould, and Toint [1988b] is designed to solve problems with simple bound constraints, i.e, $l_i \leq x_i \leq u_i$, $i = 1, 2, \ldots, n$. The bound constraints are incorporated into a box constrained trust region strategy for calculating global steps, in which an inexact Newton's method oriented towards large scale problems is used. This method uses a conjugate gradient method to approximately solve the trust region problem at each iteration, and also incorporates a new procedure that allows the set of active bound constraints to change substantially at each iteration. In this context, Conn, Gould, and Toint [1988b] conclude that the SR1 performance is, in general, somewhat better than the BFGS in terms of iterations and function evaluations on their test problems. They point out that the use of a trust region removes a main disadvantage of SR1 methods by allowing a meaningful step to be taken even when the approximation is indefinite. They also point out that the skipping technique used when the SR1 denominator is nearly zero was almost never used in their tests. They attribute part of the success of the SR1 to the possible convergence of the updates to the true second derivatives as discussed above. In Conn, Gould, and Toint [1991], they tested this convergence using random search directions. These tests showed that, in comparison with other updates such as the BFGS, the DFP, or the PSB, the SR1 generates more accurate Hessian approximations.

The purpose of this paper is to better understand the computational and theoretical properties of the SR1 update in the context of basic line search and trust region methods for unconstrained optimization. In the next section, we present computational results we obtained for the SR1 and the BFGS methods using standard line search and trust region algorithms for small to medium sized unconstrained optimization problems. We also report on tests of the convergence of the sequence of Hessian approximation matrices, $\{B_k\}$, generated by the SR1 and BFGS formulas, and of the condition of uniform linear independence of the sequence of steps which is required by the theory of Conn, Gould, and Toint [1991]. These results indicate that this assumption may not be satisfied for many problems. Therefore in Section 3, we prove a new convergence result without the assumption of uniform linear independence of steps. Instead, it requires the assumption of boundedness and positive definiteness of the Hessian approximation. In Section 4, we present computational results regarding the positive definiteness of the SR1 update, and an interesting example. Finally, in Section 5 we make some brief conclusions and comments regarding future research.

2. Computational Results and Algorithms

In this section, we present and discuss some numerical experiments that were conducted in order to test the performance of secant methods for unconstrained optimization using the SR1 formula against those using the BFGS update.

The algorithms we used are from the UNCMIN unconstrained optimization software package (Schnabel, Koontz, and Weiss [1985]) which provides the options of both line search and trust region strategies for calculating global steps. The line search is based on backtracking, using quadratic or cubic modeling of $f(x)$ in the direction of search, and the trust region step is determined using the "hook step" method to approximately minimize the quadratic model within the trust region. The frameworks of these algorithms are given below.

Algorithm 2.1 Quasi-Newton method (Line Search)

Step 0 Given an initial point x_0, an initial positive definite matrix B_0, and $\alpha = 10^{-4}$, set k (iteration number)$= 0$.

Step 1 If a convergence criterion is achieved, then stop.

Step 2 Compute a quasi-Newton direction

$$p_k = -(B_k + \mu_k I)^{-1}\nabla f(x_k)$$

where μ_k is a nonnegative scalar such that $\mu_k = 0$ if B_k is safely positive definite, else $\mu_k \geq 0$ is such that $B_k + \mu_k I$ is safely positive definite.

Step 3 {Using a backtracking line search, find an acceptable steplength.}

(3.1) Set $\lambda_k = 1$.

(3.2) If $f(x_{k+1}) \leq f(x_k) + \alpha\lambda_k\nabla f(x_k)^T p_k$, then go to Step 4.

(3.3) If first backtrack, then select the new λ_k such that $x_{k+1}(\lambda_k)$ is the local minimizer of the one-dimensional quadratic interpolating $f(x_k)$, $\nabla f(x_k)^T p_k$, and $f(x_k + p_k)$ but constrain the new λ_k to be ≥ 0.1, else select the new λ_k such that $x_{k+1}(\lambda_k)$ is the local minimizer of the one-dimensional cubic interpolating $f(x_k)$, $\nabla f(x_k)^T p_k$, $f(x_{k+1}(\lambda_{prev}))$, and $f(x_{k+1}(\lambda_{2prev}))$ but constrain the new λ_k to be in $[0.1\lambda_{prev}, 0.5\lambda_{prev}]$.
($x_{k+1}(\lambda) = x_k + \lambda_{P_k}$ and $\lambda_{prev}, \lambda_{2prev}$ = previous two steplengths.)

(3.4) Go to (3.2).

Step 4 Set $x_{k+1} = x_k + \lambda_k p_k$.

Step 5 Compute the next Hessian approximation B_{k+1}.

Step 6 Set $k = k + 1$, and go to Step 1.

Algorithm 2.2 Quasi-Newton method (Trust Region)

Step 0 Given an initial point x_0, an initial positive definite matrix B_0, an initial trust region radius Δ_0, $\eta_1 \in (0, 1)$ and $\eta_2 \geq 1$, set $k = 0$.

Step 1 If a convergence criterion is achieved, then stop.

Step 2 If B_k is not positive definite set $\hat{B}_k = B_k + \mu_k I$ where μ_k is such that $\hat{B}_k = B_k + \mu_k I$ is safely positive definite, else set $\hat{B}_k = B_k$.

Step 3 {Compute trust region step by hook step approximation.}
Find an approximate solution to

$$\min_{s \in R^n} \nabla f(x_k)^T s + \frac{1}{2} s^T \hat{B}_k s \quad \text{subject to } \|s\| \leq \Delta_k$$

by selecting

$$s_k = -(\hat{B}_k + \nu_k I)^{-1} \nabla f(x_k), \ \nu_k \geq 0$$

such that $\|s_k\| \in [0.75\Delta_k, 1.5\Delta_k]$, or

$$s_k = -\hat{B}_k^{-1} \nabla f(x_k),$$

if $\|\hat{B}_k^{-1} \nabla f(x_k)\| \leq 1.5\Delta_k$.

Step 4 Set $\text{ared}_k = f(x_k + s_k) - f(x_k)$.

Step 5 If $\text{ared}_k \leq 10^{-4} \nabla f(x_k)^T s_k$, then

(5.1) set $x_{k+1} = x_k + s_k$;

(5.2) calculate $\text{pred}_k = \nabla f(x_k)^T s_k + \frac{1}{2} s_k^T B_k s_k$;

(5.3) if $\dfrac{\text{ared}_k}{\text{pred}_k} < 0.1$, then set $\Delta_{k+1} = \Delta_k / 2$, else if $\dfrac{\text{ared}_k}{\text{pred}_k} > 0.75$, then set $\Delta_{k+1} = 2\Delta_k$, otherwise $\Delta_{k+1} = \Delta_k$;

(5.4) go to Step 7;

Step 6 else

(6.1) if relative steplength is too small, then stop; else calculate the λ_k for which $x_k + \lambda_k s_k$ is the minimizer of the one-dimensional quadratic interpolating $f(x_k)$, $f(x_k + s_k)$, and $\nabla f(x_k)^T s_k$; set new $\Delta_k = \lambda_k \|s_k\|$, but constrain new Δ_k to be between 0.1 and 0.5 times current Δ_k.

(6.2) go to Step 3.

Step 7 Compute the next Hessian approximation, B_{k+1}.

Step 8 Set $k = k + 1$, and go to Step 1.

Procedures for updating λ_k in Step 3 of the line search algorithm are found in Algorithm A6.3.1 of Dennis and Schnabel [1983]. While a steplength $\lambda_k > 1$ is not considered in the reported results, in our experience permitting $\lambda_k > 1$ makes very little difference on these test problems. Procedures for finding ν_k in Step 3 of the trust region algorithm are found in Algorithm A6.4.2 of Dennis and Schnabel [1983], and are based on Hebden [1973] and Moré [1977]. In both algorithms, the procedure for selecting μ_k in Step 2 is found in Gill, Murray, and Wright [1981]. (They give an algorithm for finding a diagonal matrix D, such that $B_k + D$ is safely positive definite. If $D = 0$, then μ_k is set to 0, else an upper bound b_1 on μ_k is calculated using the Gerschgorin circle theorem, and μ_k is set to $\min\{b_1, b_2\}$ where $b_2 = \max\{[D]_{ii}, 1 \leq i \leq n\}$.) In our experience, when B_k is indefinite, μ_k is quite close to the most negative eigenvalue of B_k, so that the algorithm usually finds an approximate minimizer of the quadratic model subject to the trust region constraint.

Both algorithms terminate if one of the following stopping criteria is met.

(1) The number of iterations exceeds a given upper limit.

(2) The relative gradient, $\max\limits_{1\leq i\leq n}\left\{|[\nabla f(x_k)]_i|\dfrac{\max\{|[x_{k+1}]_i|, 1\}}{\max\{|f(x_{k+1})|, 1\}}\right\}$, is less than a given gradient tolerance.

(3) The relative step, $\max\limits_{1\leq i\leq n}\left\{\dfrac{\max\{|[x_{k+1}]_i - [x_k]_i|\}}{\max\{|[x_{k+1}]_i|, 1\}}\right\}$, is less than a given step tolerance.

All the algorithms used $B_0 = I$.

2.1. Comparison of the SR1 and the BFGS Methods

Using the above outlined algorithms, we tested the SR1 method and the BFGS method on a variety of test problems selected from Moré, Garbow, and Hillstrom [1981] and from Conn, Gould, and Toint [1988b] (see Table A1 in the appendix.) First derivatives were approximated using finite differences. The gradient stopping tolerance used was 10^{-5}, and the step tolerance was (machine epsilon)$^{1/2}$. The upper bound used on the number of iterations was 500. As done in Conn, Gould, and Toint [1988b], we skipped the SR1 update if either

$$|s_k^T(y_k - B_k s_k)| < r\|s_k\|\|y_k - B_k s_k\|,$$

where $r = 10^{-8}$, or if $\|B_{k+1} - B_k\| > 10^8$. The BFGS update was skipped if $s_k^T y_k <$ (machine epsilon)$^{1/2}$)$\|s_k\|\|y_k\|$. All experiments were run using double precision arithmetic on a Pyramid P90 computer that has a machine epsilon of order 10^{-16}.

For each test function, Tables A2 and A3 in the appendix report the performance of the SR1 and BFGS methods using line search and trust region respectively. The tables contain the number of the function as given in the original source (see Table A1), the dimension of the problem (n), the number of iterations required to solve the problem (itrn.), the number of function evaluations, (f-eval.), required to solve the problem (which includes n for each finite difference gradient evaluation), and the relative gradient at the solution (rgx). The last column (sp) indicates whether the starting point used is x_0, $10x_0$, or $100x_0$, where x_0 is the standard starting point.

In order to compare the performance of the two methods with respect to the number of iterations and the number of function evaluations required to solve these problems, we consider problems solved by both methods and calculate the ratio of the mean of the number of iterations (or function evaluations) required to solve these problems by the SR1 method to the corresponding mean for the BFGS method. Table 1 below reports the ratios of these means, using both arithmetic mean and geometric mean. These numbers indicate that on the set of test problems we used, the SR1 is 10% to 15% faster and cheaper than the BFGS method.

Table 2 gives the number of problems where the SR1 method requires at least 5, 10, 20, 30, 40, 50 iterations less than the BFGS method, and vice versa. This table, which is based on numbers from Table A2, also indicates the superiority of the SR1 on these problems.

Table 1: Ratio of SR1 Cost to BFGS Cost

Mean	Line Search		Trust Region	
	Itrn.	Function Evaluations	Itrn.	Function Evaluations
Arithmetic	0.82	0.83	0.84	0.88
Geometric	0.83	0.85	0.84	0.92

Table 2: Comparisons of Iterations

	Line Search						Trust Region					
Iterations Different	5	10	20	30	40	50	5	10	20	30	40	50
SR1 Better	26	20	13	10	7	3	27	20	11	9	5	1
BFGS Better	7	5	2	2	1	1	8	6	3	1	1	1

2.2. Error in the Hessian Approximation and Uniform Linear Independence

In an attempt to understand the difference between the SR1 and the BFGS, we tested how closely the final Hessian approximations produced by the line search and trust region SR1 and BFGS algorithms come to the exact Hessians at the final iterates. Recall that the Hessian error for the SR1 is analyzed by Conn, Gould, and Toint [1991] under the assumption of uniform linear independence which we redefine here.

Definition A sequence of vectors $\{s_k\}$ in R^n is said to be uniformly linearly independent if there exist $\zeta > 0$, k_0 and $m \geq n$ such that, for each $k \geq k_0$, one can choose n distinct indices $k \leq k_1 < \ldots < k_n \leq k + m$ such that the minimum singular value of the matrix $S_k = [s_{k_1}/||s_{k_1}||, \ldots, s_{k_n}/||s_{k_n}||]$ is $\geq \zeta$.

Using this definition, Theorem 2 of Conn, Gould, and Toint [1991] proves the following.

Theorem 2.1 (Conn, Gould, and Toint [1991]) Suppose that $f(x)$ is twice continuously differentiable everywhere, and that $\nabla^2 f(x)$ is bounded and Lipschitz continuous, that is, there exist constants $M > 0$ and $\gamma > 0$ such that for all $x, y \in R^n$,

$$||\nabla^2 f(x)|| \leq M \text{ and } ||\nabla^2 f(x) - \nabla^2 f(y)|| \leq \gamma ||x - y||.$$

Let $x_{k+1} = x_k + s_k$, where $\{s_k\}$ is a uniformly linearly independent sequence of steps, and suppose that $\lim_{k\to\infty} \{x_k\} = x_*$ for some $x_* \epsilon R^n$. Let $\{B_k\}$ be generated by the SR1 formula

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

where B_0 is symmetric, and suppose that $\forall k \geq 0$, y_k and s_k satisfy

$$|s_k^T(y_k - B_k s_k)| \geq r\|s_k\|\|y_k - B_k s_k\|, \tag{2.1}$$

for some fixed $r \in (0,1)$. Then $\lim_{k\to\infty} \|B_k - \nabla^2 f(x_*)\| = 0$.

We now present some computational tests to determine to what extent such Hessian convergence occurs in practice. For these tests we used analytic gradients and a gradient stopping tolerance of 10^{-10} and computed the quantity

$$\|B_l - \nabla^2 f(x_l)\| / \|\nabla^2 f(x_l)\|,$$

where x_l is the solution obtained by the algorithm, and B_l is the Hessian approximation at x_l. These results are reported in Tables A4 and A5 in the appendix and summarized in Tables 3 and 4 below. Tables 3 and 4 list for each method, the number of problems for which $\|B_l - \nabla^2 f(x_l)\| / \|\nabla^2 f(x_l)\|$ lies in a given range.

Table 3: Number of Problems with $\|B_l - \nabla^2 f(x_l)\| / \|\nabla^2 f(x_l)\|$ in Indicated Range (Line Search Methods)

	$\leq 10^{-4}$	$[10^{-4}, 10^{-3})$	$[10^{-3}, 10^{-2})$	$[10^{-2}, 10^{-1})$	$[10^{-1}, 1)$	≥ 1
SR1	4	3	2	8	3	8
BFGS	3	0	1	10	6	8

Table 4: Number of Problems with $\|B_l - \nabla^2 f(x_l)\| / \|\nabla^2 f(x_l)\|$ in Indicated Range (Trust Region Methods)

	$\leq 10^{-4}$	$[10^{-4}, 10^{-3})$	$[10^{-3}, 10^{-2})$	$[10^{-2}, 10^{-1})$	$[10^{-1}, 1)$	≥ 1
SR1	5	0	4	5	4	10
BFGS	0	0	5	7	7	9

While the SR1 seems to produce slightly better final approximations than the BFGS, there is no evidence from these tables that it significantly outperforms the BFGS with respect to convergence to the actual Hessian at the solution. Also, in a good number of cases, neither method comes close to the correct Hessian.

The lack of convergence of the SR1 Hessian approximations to the correct value in many of these problems may appear to conflict with the analysis of Conn, Gould and Toint [1991] given in Theorem 2.1. In fact, there are two possible explanations for this apparent conflict: either the algorithm has not converged closely enough for the final convergence of the matrices to be apparent (this is hard to test in finite precision arithmetic) or an assumption of Theorem 2.1 must be violated. The two assumptions of Theorem 2.1 that could possibly be invalid are that the denominator of the SR1 is not too small (2.1), and the uniform linear independence condition. In our experiments, (2.1) was violated at most once for each test problem, and so this assumption does not appear to be a problem in the SR1 method. Thus, we decided to test whether the uniform linear independence condition is satisfied for these problems.

Since the uniform linear independence condition would be very hard to test due to the freedom to choose m and τ, we have tested a weaker condition. For each value $\tau = 10^{-i}, i = 1, 2, \ldots, 8$, we computed the number of steps (say m) required so that the smallest singular value of the matrix, $\hat{S}_m$, composed of the final normalized m steps of the algorithm, is $> \tau$ ($\hat{S}_m = [s_l/\|s_l\|, s_{l-1}/\|s_{l-1}\|, \ldots, s_{l-(m-1)}/\|s_{l-(m-1)}\|]$, where $m \geq n$). Tables A6 and A7 contain the results of these experiments, which are summarized in Tables 5 and 6 below. A "*" entry in Tables A6 and A7 means that the smallest singular value is $< \tau$ even if all the iterates are used.

These results indicate that the uniform linear independence assumption does not seem to hold for all problems, especially those with large dimensions. Therefore in the next section we develop a convergence result for the SR1 method that does not make this assumption.

Table 5: Number of Problems where $\sigma_{min}(S_m) > \tau$ for m/n in Indicated Range. - Line Search SR1 Method

	m/n in					
τ	[1,2)	[2,3)	[3 – 4)	[4 – 5)	[5 – 10)	Never
10^{-1}	7	1	3	3	6	8
10^{-2}	12	1	0	3	5	7
10^{-8}	12	1	0	4	4	7

Table 6: Number of Problems where $\sigma_{min}(S_m) > \tau$ for m/n in Indicated Range. - Trust Region SR1 Method

	m/n in					
τ	[1,2)	[2,3)	[3 – 4)	[4 – 5)	[5 – 10)	Never
10^{-1}	4	3	0	3	6	12
10^{-2}	12	1	0	3	5	7
10^{-8}	13	0	0	3	5	7

3. Convergence Rate of the SR1 Without Uniform Linear Independence

As was indicated at the end of the previous section, the condition of uniform linear independence of the sequence $\{s_k\}$ under which Conn, Gould, and Toint [1991] analyze the performance of the SR1 method may be too strong in practice. Therefore in this section we consider the convergence rate of the SR1 method without this condition. We will show that if we drop the condition of uniform linear independence of $\{s_k\}$ but add instead the assumption that the sequence $\{B_k\}$ remains positive definite and bounded, then the line search Algorithm 2.1 generates at least p q-superlinear steps out of every $n+p$ steps. This will enable us to prove that convergence is $2n$-step q-quadratic.

The basic idea behind our proof is that, if any step falls close enough to a subspace spanned by $m \leq n$ recent steps, then the Hessian approximation must be quite accurate in this subspace. Thus, if in addition the step is the full secant step $-B_k^{-1}\nabla f(x_k)$, it should be a superlinear step. But in a line search method, for the step to be the full secant step, B_k must be positive definite, which accounts for the new assumption of positive definiteness of B_k at the good steps. In Section 4 we will show that empirically this assumption seems very sound, although counterexamples are possible.

Throughout this section the following assumptions will frequently be made:

Assumptions

3.1 The function f has a local minimizer at a point x_* such that $\nabla^2 f(x_*)$ is positive definite, and its Hessian $\nabla^2 f(x)$ is Lipschitz continuous near x_*, that is, there exists a constant $\gamma > 0$ such that for all x, y in some neighborhood of x_*,

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq \gamma \|x - y\|.$$

3.2 The sequence $\{x_k\}$ converges to the local minimizer x_*.

We first state the following result, due to Conn, Gould, and Toint [1991], which does not assume linear independence of the step directions and which will be used in the proof of the next lemma.

Lemma 3.1 Let $\{x_k\}$ be a sequence of iterates defined by $x_{k+1} = x_k + s_k$. Suppose that Assumptions 3.1 and 3.2 hold, that the sequence of matrices $\{B_k\}$ is generated from $\{x_k\}$ by the SR1 update, and that for each iteration

$$|s_k^T(y_k - B_k s_k)| \geq r\|s_k\|\|y_k - B_k s_k\| \tag{3.1}$$

where r is a constant $\in (0,1)$. Then, for each j, $\|y_j - B_{j+1}s_j\| = 0$, and

$$\|y_j - B_i s_j\| \leq \frac{\gamma}{r}\left(\frac{2}{r} + 1\right)^{i-j-2} \eta_{i,j}\|s_j\| \tag{3.2}$$

for all $i \geq j+2$, where $\eta_{i,j} = \max\{\|x_p - x_s\| \mid j \leq s \leq p \leq i\}$, and γ is the Lipschitz constant from Assumption 3.1.

Actually, it is apparent from the proof of Lemma 3.1 by Conn, Gould, and Toint that, if the update is skipped whenever (3.1) is violated, then (3.2) still holds for all j for which (3.1) is true.

In the lemma below, we show that if the sequence of steps generated by an iterative process using the SR1 update satisfies (3.1), and the sequence of matrices is bounded, then out of any set of $n+1$ steps, at least one is very good. As in the previous lemma, condition (3.1) actually needs only hold at this set of $n+1$ steps, as long as the update is not made when that condition fails.

Lemma 3.2 Suppose the assumptions of Lemma 3.1 are satisfied for the sequences $\{x_k\}$ and $\{B_k\}$ and that in addition there exists M for which $\|B_k\| \leq M$ for all k. Then there exists $K \geq 0$ such that for any set of $n+1$ steps $\mathcal{S} = \{s_{k_j} : K \leq k_1 \leq \cdots \leq k_{n+1}\}$ there exists an index k_m with $m \in \{2, 3, \ldots, n+1\}$ such that

$$\frac{\|(B_{k_m} - \nabla^2 f(x_*))s_{k_m}\|}{\|s_{k_m}\|} < \bar{c}\epsilon_{\mathcal{S}}^{1/n}$$

where

$$\epsilon_{\mathcal{S}} = \max_{1 \leq j \leq n+1} \{\|x_{k_j} - x_*\|\}$$

and

$$\bar{c} = 4\left[\gamma + \sqrt{n}\frac{\gamma}{r}\left(\frac{2}{r}+1\right)^{k_{n+1}-k_1-2} + M + \|\nabla^2 f(x_*)\|\right].$$

Proof. Given $\mathcal{S}$, for $j = 1, 2, \ldots, n+1$ define

$$S_j = \left[\frac{s_{k_1}}{\|s_{k_1}\|}, \frac{s_{k_2}}{\|s_{k_2}\|}, \ldots, \frac{s_{k_j}}{\|s_{k_j}\|}\right].$$

We will first show that $\exists m \in [2, n+1]$ such that $s_{k_m}/\|s_{k_m}\| = S_{m-1}u - w$, S_{m-1} has full column rank and is well conditioned, and $\|w\|$ is very small. (In essence either $m = n+1$, S_{m-1} spans n-space well, and $w = 0$, or $m < n+1$, S_{m-1} has full rank and is well conditioned, and s_{k_m} is nearly in the space spanned by S_{m-1}.) Then, using the fact that $(B_{k_m} - \nabla^2 f(x_*))S_{m-1}$ is small due to the Hessian approximating properties of the SR1 update given in Lemma 3.1 above, and that w is small by this construction, we will have the desired result.

For $j \in \{1, \ldots, n\}$, let τ_j be the smallest singular value of S_j and define $\tau_{n+1} = 0$. Note that

$$1 = \tau_1 \geq \tau_2 \ldots \geq \tau_{n+1} = 0$$

Let m be the smallest integer for which

$$\frac{\tau_m}{\tau_{m-1}} < \epsilon_{\mathcal{S}}^{1/n}. \tag{3.3}$$

Then since $m \leq n+1$ and $\tau_1 = 1$,

$$\begin{aligned} \tau_{m-1} &= \tau_1\left(\frac{\tau_2}{\tau_1}\right)\cdots\left(\frac{\tau_{m-1}}{\tau_{m-2}}\right) \\ &> \epsilon_{\mathcal{S}}^{(m-2)/n} \\ &> \epsilon_{\mathcal{S}}^{(n-1)/n}. \end{aligned} \tag{3.4}$$

Since $x_k \to x_*$, we may assume without loss of generality that $\epsilon_{\mathcal{S}} \in (0, (\frac{1}{4})^n)$ for all k. Now we choose $z \in R^m$ such that

$$\|S_m z\| = \tau_m \|z\|, \tag{3.5}$$

and

$$z = \begin{bmatrix} u \\ -1 \end{bmatrix}$$

where $u \in R^{m-1}$. (The last component of z is nonzero due to (3.3).) Let $w = S_m z$. Then, from the definition of S_m and z,

$$\frac{s_{k_m}}{\|s_{k_m}\|} = S_{m-1}u - w. \tag{3.6}$$

Since τ_{m-1} is the smallest singular value of S_{m-1} we have that

$$\begin{aligned} \|u\| &\leq \frac{1}{\tau_{m-1}}\|S_{m-1}u\| \\ &= \frac{1}{\tau_{m-1}}\|w + \frac{s_{k_m}}{\|s_{k_m}\|}\| \\ &\leq \frac{\|w\|+1}{\tau_{m-1}}. \end{aligned} \tag{3.7}$$

By (3.4) this implies that

$$\|u\| < \frac{\|w\|+1}{\epsilon_S^{(n-1)/n}}. \tag{3.8}$$

Also, using (3.5) and (3.7), we have that

$$\begin{aligned} \|w\|^2 &= \|S_m z\|^2 \\ &= \tau_m^2\|z\|^2 \\ &= \tau_m^2(1+\|u\|^2) \\ &\leq \tau_m^2 + \left(\frac{\tau_m}{\tau_{m-1}}\right)^2(\|w\|+1)^2. \end{aligned}$$

Therefore, since (3.3) implies that $\tau_m < \epsilon_S^{1/n}$, using (3.3),

$$\begin{aligned} \|w\|^2 &< \epsilon_S^{2/n} + \epsilon_S^{2/n}(\|w\|+1)^2 \\ &< 4\epsilon_S^{2/n}(\|w\|+1)^2. \end{aligned} \tag{3.9}$$

This implies that

$$\|w\|(1-2\epsilon_S^{1/n}) < 2\epsilon_S^{1/n}$$

and hence, $\|w\| < 1$, since $\epsilon_S < (\frac{1}{4})^n$. Therefore, (3.8) and (3.9) imply that

$$\|u\| < \frac{2}{\epsilon_S^{(n-1)/n}} \tag{3.10}$$

$$\|w\| < 4\epsilon_S^{1/n}. \tag{3.11}$$

This gives the desired result that w is small, as well as a necessary bound on $\|u\|$.

Now we show that $\|(B_{k_j} - \nabla^2 f(x_*))S_{j-1}\|$, $j \in [2, n+1]$, is small. Note that this result is independent of the choice of j. By Lemma 3.1 we have that

$$\begin{aligned} \|y_i - B_{k_j}s_i\| &\leq \frac{\gamma}{r}\left(\frac{2}{r}+1\right)^{k_j-i-2}\eta_{k_j,i}\|s_i\| \\ &\leq 2\frac{\gamma}{r}\left(\frac{2}{r}+1\right)^{k_{n+1}-k_1-2}\epsilon_S\|s_i\| \end{aligned} \tag{3.12}$$

for all $i \in \{k_1, k_2, \ldots, k_{j-1}\}$. Also, letting

$$G_i = \int_0^1 \nabla^2 f(x_i + ts_i)dt,$$

we have

$$\begin{aligned} G_i s_i &= \int_0^1 \nabla^2 f(x_i + ts_i)s_i dt \\ &= \nabla f(x_{i+1}) - \nabla f(x_i) \\ &= y_i, \end{aligned}$$

and by the Lipschitz continuity of $\nabla^2 f(x)$,

$$\begin{aligned} \|y_i - \nabla^2 f(x_*)s_i\| &= \|(G_i - \nabla^2 f(x_*))s_i\| \\ &= \|\int_0^1 (\nabla^2 f(x_i + ts_i) - \nabla^2 f(x_*))s_i dt\| \\ &\le \|s_i\| \int_0^1 \|\nabla^2 f(x_i + ts_i) - \nabla^2 f(x_*)\| dt \\ &\le \gamma \|s_i\| \int_0^1 \|x_i + ts_i - x_*\| dt \\ &\le \gamma \|s_i\| \epsilon_S, \end{aligned} \tag{3.13}$$

where γ is the Lipschitz constant. Therefore, using the triangle inequality, and (3.12) and (3.13) we have

$$\begin{aligned} \|(B_{k_j} - \nabla^2 f(x_*))\frac{s_i}{\|s_i\|}\| &\le \|(y_i - B_{k_j})\frac{s_i}{\|s_i\|}\| + \|(y_i - \nabla f(x_*))\frac{s_i}{\|s_i\|}\| \\ &\le (2c + \gamma)\epsilon_S, \end{aligned}$$

where $c = \frac{\gamma}{r}\left(\frac{2}{r} + 1\right)^{k_{n+1} - k_1 - 2}$, and hence for any $j \in [2, n+1]$

$$\|(B_{k_j} - \nabla^2 f(x_*))S_{j-1}\| \le \sqrt{n}(2c + \gamma)\epsilon_S. \tag{3.14}$$

Finally, using (3.6), (3.14) with $j = m$, (3.11) and (3.10) we have that

$$\begin{aligned} \frac{\|(B_{k_m} - \nabla^2 f(x_*))s_{k_m}\|}{\|s_{k_m}\|} &= \|(B_{k_m} - \nabla^2 f(x_*))(S_{m-1}u - w)\| \\ &\le \|(B_{k_m} - \nabla^2 f(x_*))S_{m-1}\| \|u\| \\ &\qquad + \|B_{k_m} - \nabla^2 f(x_*)\| \|w\| \\ &\le \|u\| \sqrt{n}(2c + \gamma)\epsilon_S + \|w\|(\|B_{k_m}\| + \|\nabla^2 f(x_*)\|) \\ &< \left(\frac{2}{\epsilon_S^{(n-1)/n}}\right) \sqrt{n}(2c + \gamma)\epsilon_S \\ &\qquad + 4\epsilon_S^{1/n}(M + \|\nabla^2 f(x_*)\|) \\ &< 4\left[\sqrt{n}(c + \gamma) + M + \|\nabla^2 f(x_*)\|\right]\epsilon_S^{1/n} \\ &= \bar{c}\epsilon_S^{1/n}. \square \end{aligned}$$

In order to use this lemma to establish a rate of convergence we need the following result which is closely related to the well-known superlinear convergence characterization of Dennis and Moré [1974].

Lemma 3.3 Suppose the function f satisfies Assumption 3.1. If the quantities $e_k = \|x_k - x_*\|$ and $\dfrac{\|(B_k - \nabla^2 f(x_*))s_k\|}{\|s_k\|}$ are sufficiently small, and if $B_k s_k = -\nabla f(x_k)$, then

$$\|x_k + s_k - x_*\| \leq \|\nabla^2 f(x_*)^{-1}\| \left[2\frac{\|(B_k - \nabla^2 f(x_*))s_k\|}{\|s_k\|}e_k + \frac{\gamma}{2}e_k^2\right].$$

Proof. By the definition of s_k

$$\nabla^2 f(x_*)s_k = (\nabla^2 f(x_*) - B_k)s_k - \nabla f(x_k)$$

so that

$$s_k = -(x_k - x*) + \nabla^2 f(x_*)^{-1}\left[(\nabla^2 f(x_*) - B_k)s_k - \nabla f(x_k) + \nabla^2 f(x_*)(x_k - x_*)\right]. \quad (3.15)$$

Therefore, using Taylor's theorem and Assumption 3.1,

$$\|x_k - x_* + s_k\| \leq \|\nabla^2 f(x_*)^{-1}\| \left[\|(\nabla^2 f(x_*) - B_k)s_k\| + \frac{\gamma}{2}e_k^2\right]. \quad (3.16)$$

Now it follows from (3.15) that if $\|\nabla^2 f(x_*)^{-1}\|\|(B_k - \nabla^2 f(x_*))s_k\|/\|s_k\| \leq \frac{1}{3}$ then by Taylor's theorem,

$$\|s_k\| \leq \frac{3}{2}[\|x_k - x_*\| + \|\nabla^2 f(x_*)^{-1}\|\frac{\gamma}{2}\|x_k - x_*\|^2] \leq 2\|x_k - x_*\|,$$

if e_k is sufficiently small. Using this inequality together with (3.16) gives the result. □

Using these two lemmas one can show that for any $p > n$, Algorithm 2.1 will generate at least $p - n$ superlinear steps every p iterations provided that B_k is safely positive definite, which implies that B_k is not perturbed in Step 2 and $\mu_k = 0$. In the following theorem, this is proved and used to establish a rate of convergence for Algorithm 2.1 under the assumption that the sequence $\{B_k\}$ becomes, and stays, positive definite. In a corollary we show that this implies that the rate of convergence for Algorithm 2.1 is $2n$-step q-quadratic. As we will see in the next section, our test results show that the positive definiteness condition is generally satisfied in practice. We are assuming here that if B_k is positive definite, then it is not perturbed in Step 2, i.e., we are assuming that "safely positive definite" just means positive definite.

Theorem 3.1 Consider Algorithm 2.1 and suppose that Assumptions 3.1 and 3.2 hold. Assume also that for all $k \geq 0$,

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

for a fixed $r \in (0, 1)$, and that $\exists M$ for which $\|B_k\| \leq M$ $\forall k$. Then, if $\exists K_0$ such that B_k is positive definite for all $k \geq K_0$, then for any $p \geq n + 1$ there exists K_1 such that for all $k \geq K_1$,

$$e_{k+p} \leq \alpha e_k^{p/n} \quad (3.17)$$

where α is a constant and e_j is defined as $\|x_j - x_*\|$.

Proof. Since $\nabla^2 f(x_*)$ is positive definite, there exists K_1, $\beta_1 > 0$ and $\beta_2 > 0$ such that

$$\beta_1[f(x_k) - f(x_*)]^{\frac{1}{2}} \leq \|x_k - x_*\| \leq \beta_2[f(x_k) - f(x_*)]^{\frac{1}{2}} \tag{3.18}$$

for all $k \geq K_1$. Therefore, since we have a descent method, for all $l > k > K_1$, $\|x_l - x_*\| \leq \frac{\beta_2}{\beta_1}\|x_k - x_*\|$. Now, given $k > K_1$ we apply Lemma 3.2 to the set $\{s_k, s_{k+1}, \ldots, s_{k+n}\}$. Thus, there exists $l_1 \in \{k+1, \ldots, k+n\}$ such that

$$\frac{\|(B_{l_1} - \nabla^2 f(x_*))s_{l_1}\|}{\|s_{l_1}\|} < \bar{c}\left(\frac{\beta_2}{\beta_1}e_k\right)^{1/n}. \tag{3.19}$$

(If there is more than one such index l_1, we choose the smallest.) Equation (3.19) implies that for $\|x_{l_1} - x_*\|$ sufficiently small, by Theorem 6.4 of Dennis and Moré [1977], Algorithm 2.1 will choose $\lambda_{l_1} = 1$ so that $x_{l_1+1} = x_{l_1} + s_{l_1}$. This fact, together with Lemma 3.3 and (3.19), implies that if e_k is sufficiently small then

$$e_{l_1+1} \leq \hat{\alpha} e_k^{1/n} e_{l_1} \tag{3.20}$$

for some constant $\hat{\alpha}$. Now we can apply Lemma 3.2 to the set

$$\{s_k, s_{k+1}, \ldots, s_{k+n}, s_{k+n+1}\} - \{s_{l_1}\}$$

to get l_2. Repeating this $n - p$ times we get a set of integers $l_1 < l_2 < \ldots < l_{p-n}$, with $l_1 > k$ and $l_{p-n} < k + p$ such that

$$e_{l_i+1} \leq \hat{\alpha} e_k^{1/n} e_{l_i} \tag{3.21}$$

for each l_i. Now letting $h_j = [f(x_j) - f(x_*)]^{\frac{1}{2}}$, since we have a descent method,

$$h_{j+1} \leq h_j, \tag{3.22}$$

and using (3.18) we have that for our arbitrary $k \geq K_1$,

$$\begin{aligned} h_{l_i+1} &\leq \frac{1}{\beta_1} e_{l_i+1} \\ &\leq \frac{\hat{\alpha}}{\beta_1} e_k^{1/n} e_{l_i} \\ &\leq \frac{\hat{\alpha}\beta_2}{\beta_1} e_k^{1/n} h_{l_i} \end{aligned} \tag{3.23}$$

for $i = 1, 2, \ldots, p - n$. Therefore using (3.22) and (3.23) we have that

$$h_{k+p} \leq \left(\frac{\hat{\alpha}\beta_2}{\beta_1} e_k^{1/n}\right)^{p-n} h_k$$

which, by (3.18) implies that

$$e_{k+p} \leq \frac{\beta_2}{\beta_1}\left(\frac{\hat{\alpha}\beta_2}{\beta_1} e_k^{1/n}\right)^{p-n} e_k.$$

Therefore,

$$e_{k+p} \leq \hat{\alpha}^{p-n}\left(\frac{\beta_2}{\beta_1}\right)^{p-n+1} e_k^{p/n},$$

and 3.17 follows. □

Corollary 3.1 Under the assumptions of Theorem 3.1 the sequence $\{x_k\}$ generated by Algorithm 2.1 is $n+1$-step q-superlinear, i.e.,

$$\frac{e_{k+n+1}}{e_k} \to 0,$$

and is $2n$-step q-quadratic, i.e.,

$$\limsup_{k\to\infty} \frac{e_{k+2n}}{e_k^2} \leq \infty.$$

Proof. Let $p = n+1$, and $p = 2n$ in Theorem 3.1. $\square$

Note that a $2n$-step q quadratically convergent sequence has an r order of $\left(\sqrt{2}\right)^{\frac{1}{n}}$. Since the integer p in the theorem is arbitrary, an interesting, purely theoretical question is what value of p will prove the highest r-convergence order for the sequence. It is not hard to show that, by choosing p to be an integer close to en, the r order approaches $e^{\frac{1}{en}} \approx 1.44^{\frac{1}{n}}$ for n sufficiently large, and that this value is optimal for this technique of analysis.

4. Positive Definiteness of the SR1 Update

One of the requirements in Theorem 3.1 for the rate of convergence to be p-step q-superlinear, is that the sequence $\{B_k\}$ generated by the SR1 method be positive definite. Actually, the proof of Theorem 3.1 only requires positive definiteness of B_k at the $p-n$ out of p "good iterations." In this section, we present computational results to confirm that in practice, the SR1 method generally satisfies this requirement.

In Table A8 in the appendix, in the fourth column, we report for each iteration whether B_k is positive definite or not. The 5th column reports the percentage of iterates at which the SR1 update is positive definite, and the 6th column contains the largest number j for which all of $B_{l-(j-1)}, \ldots, B_l$ are positive definite, where B_l is the Hessian approximation at the final iterate. The results of Table A8 are summarized in Table 4, which indicates that the SR1 formula was positive definite at least 70% of the time on every one of our test problems. In light of this, and since Theorem 3.1 really only requires positive definiteness at the "good steps" (at other steps all that is needed is that f be reduced), chances that superlinear steps will be taken at least every n steps by the algorithm seem good. Another way of viewing this is that, we know from Theorem 3.1 that out of every $2n$ steps, at least n will be "good steps" so long as B_k is positive definite at these iterations. Thus, if for example B_k is positive definite at 80% of these $2n$ steps, at least 30% of the $2n$ iterates must be "good steps."

Table 7: Percentage of Iterations with B_k Positive Definite

%	≤ 70	$[70, 90)$	$[80, 90)$	$[90, 100)$	100
problems	0	5	12	6	5

We also tested the denominator condition that

$$|s_k^T(y_k - B_k s_k)| \geq r\|s_k\|\|y_k - B_k s_k\| \tag{4.1}$$

where $r = 10^{-8}$ using standard initial points. The last column in Table A8, which reports the number of times this condition was violated, indicates that this condition rarely is violated in practice. This finding is consistent with the results of Conn, Gould, and Toint [1988b].

Finally we present an example that shows that it is possible for a line search SR1 algorithm to fail to have B_k positive definite at all iterations, and to converge linearly to the minimizer x_*. This shows that the assumptions of Theorem 3.1 cannot be guaranteed to hold. We then consider the same example in a trust region SR1 algorithm, and show that it does not suffer from the same problems. This leads us to feel that it may not be necessary to assume $\{B_k\}$ positive definite in order to prove superlinear convergence for a trust region SR1 method.

Example 4.1 Let

$$f(x) = \frac{1}{2}x^T x, \; x_0 = \begin{bmatrix} \alpha \\ 0 \end{bmatrix}, \text{ and } B_0 = \begin{bmatrix} 1 & 0 \\ 0 & \sigma \end{bmatrix}$$

where $\sigma < 0$. At the first iteration, the algorithm will compute

$$x_1 = x_0 - \begin{bmatrix} 1+\delta_0 & 0 \\ 0 & \sigma+\delta_0 \end{bmatrix}^{-1} \nabla f(x_0) = \frac{\delta_0}{1+\delta_0} x_0$$

for some $\delta_0 > -\sigma$, and accept this point as the next iterate. The SR1 update will produce $y_0 - B_0 s_0 = 0$, so that $B_1 = B_0$. The remaining iterates proceed analogously, so that for each k, $B_k = B_0$ and

$$x_{k+1} = \frac{\delta_k}{1+\delta_k} x_k$$

for some $\delta_k > -\sigma$, meaning that the rate of convergence is no better than linear with constant $\dfrac{|\sigma|}{1+|\sigma|}$.

It is interesting to consider the behavior on the same problem of a trust region SR1 algorithm that exactly solves the problem

$$\min_{s \in R^n} \nabla f(x_k)^T s + \frac{1}{2} s^T B_k s \text{ subject to } \|s\| \le \Delta_k \tag{4.2}$$

at each iteration. If there exists μ_0 such that $B_0 + \mu_0 I$ is positive definite and $\|(B_0 + \mu_0 I)^{-1} \nabla f(x_0)\| = \Delta_0$, then as in the line search method, $x_1 = \dfrac{\mu_0}{1+\mu_0} x_0$ and $B_1 = B_0$. Since ared$_0$ =pred$_0$, the trust region radius is not decreased. Thus eventually at some iterate k, we must have $\|(B_k + \mu_k I)^{-1} \nabla f(x_k)\| < \Delta_k$ for all $\mu_k > -\lambda_k$, where $\lambda_k < 0$ is the smallest eigenvalue of B_k. In this case the solution to (4.2) is the step

$$\begin{aligned} x_{k+1} &= x_k - (B_k - \lambda_k I)^+ \nabla f(x_k) - \nu e_2 \\ &= x_k - \left(\frac{1}{1-\sigma}\right) x_k - \nu e_2 \end{aligned}$$

for a $\nu \neq 0$ that makes $\|s_k\| = \Delta_k$. (Here $e_2 = (0,1)^T$ is the eigenvector of B_k corresponding to the negative eigenvalue.) It is then straightforward to verify that $y_k - B_k s_k = \nu(\sigma - 1)e_2$, $B_{k+1} = I = \nabla^2 f(x)$ and $x_{k+2} = x_*$.

A practical trust region algorithm will not solve (4.2) exactly, but any algorithm that deals with the "hard case" (when $\|(B_k - \lambda_k I)^+ \nabla f(x_k)\| < \Delta_k$) well, such as algorithms of Moré and Sorenson [1982], will have the same effect. That is, at some point it will set

$$x_{k+1} = x_k - (B_k + \mu_k I)^{-1} \nabla f(x_k) - v_k$$

where v_k is a negative curvature direction for B_k. This implies that $v_k^T e_2 \neq 0$, which in turn leads to $B_{k+1} = I$ and $x_{k+2} = x_*$. Thus the trust region method has the ability, for this example, to correct negative eigenvalues in the Hessian approximation. This indicates that it may be possible to establish superlinear convergence of a trust region SR1 algorithm without assuming a priori either strong linear independence of the iterates or positive definiteness of $\{B_k\}$. This issue is currently under investigation.

5. Conclusions and Future Research

We have attempted, in this paper, to investigate theoretical and numerical aspects of quasi-Newton methods that are based on the SR1 formula for the Hessian approximation. We considered both line search and trust region algorithms.

We tested the SR1 method on a fairly large number of standard test problems from Moré, Garbow, and Hillstorm [1981], and Conn, Gould, and Toint [1988b]. Our test results show that on the set of problems we tried, the SR1 method, on the average, requires somewhat fewer iterations and function evaluations than the BFGS method in both line search and trust region algorithms. Although there is no result for the BFGS method concerning the convergence of the sequence of approximating matrices to the correct Hessian like the one given by Conn, Gould, and Toint [1991] for the SR1, numerical tests do not show that the SR1 method is more accurate than the BFGS method in this regard. One reason for this, as indicated by our numerical experiments, is that the requirement of uniform linear independence that is needed by the theory of Conn, Gould, and Toint [1991] often fails to be satisfied in practice.

Under conditions that do not assume uniform linear independence of the generated steps, but do assume positive definiteness and boundedness of the Hessian approximations, we were able to prove $n+1$-step q-superlinear convergence, and $2n$ step quadratic convergence, of a line search SR1 method. We also gave numerical evidence that the SR1 update is positive definite most of the time, and that one of the potential problems of the formula, that of the denominator being zero, is rarely encountered in practice.

An interesting topic for future research that was mentioned in Section 4 is the convergence analysis of a trust region SR1 method, again without the assumption of uniform linear independence of steps. It is possible that the assumption of the positive definiteness of the Hessian approximations, which we showed is necessary and sufficient to prove superlinear convergence in the line search SR1 method, may not be necessary to prove superlinear convergence for a properly chosen trust region SR1 algorithm.

References

C. G. Broyden, J. E. Dennis, Jr. and J. J. Moré, (1973), *On the local and superlinear convergence of quasi-Newton methods,* Journal of the Institute of Mathematics and its Applications, 12, pp. 223-246.

A. R. Conn, N.I.M. Gould, and Ph. Toint, (1988a), *Global convergence of a class of trust region algorithms for optimization with simple bounds,* SIAM J. Num. Anal. Vol. 25, No. 2, pp. 433-460, (also Vol. 26, No. 3 (1989), pp. 764-767).

A. R. Conn, N.I.M. Gould, and Ph. Toint, (1988b), *Testing a class of methods for solving minimization problems with simple bounds on the variables,* Math. Comp. Vol. 50, No. 182, pp. 399-430.

A. R. Conn, N.I.M. Gould, and Ph. Toint, (1991), *Convergence of quasi-Newton matrices generated by the symmetric rank one update,* Math. Programming (to appear).

J. E. Dennis, Jr. and J. J. Moré, (1974), *A characterization of superlinear convergence and its application to quasi-Newton methods,* Mathematics of Computation, 28, pp. 549-560.

J. E. Dennis, Jr. and J. J. Moré, (1977), *Quasi-Newton methods, motivation and theory,* SIAM Review, 19, pp. 46-89.

J. E. Dennis, Jr. and R. B. Schnabel, (1983), *Numerical Methods for Nonlinear Equations and Unconstrained Optimization,* Prentice-Hall, Englewood Cliffs, New Jersey.

A. V. Fiacco and G. P. McCormick, (1968), *Nonlinear Programming,* John Wiley & Sons, New York.

R. Fletcher, (1980), *Practical Methods of Optimization,* John Wiley & Sons, New York.

P. E. Gill, W. Murray, and M. H. Wright, (1981), *Practical Optimization,* Academic Press, London.

D. Goldfarb, (1970), *A family of variable metric methods derived by variational means,* Mathematics of Computation, 24, pp. 23-26.

M. D. Hebden, (1973), *An algorithm for minimization using exact second derivatives,* Rept. TP515, A.E.R.E, Harwell, England.

J. J. Moré, (1977), *The Levenberg-Marquardt algorithm: implementation and theory,* in Numerical Analysis, G. A. Watson, ed., Lecture Notes in Math. 630, Springer Verlag, Berlin, pp. 105-116.

J. J. Moré, B. S. Garbow, and K. E. Hillstrom, (1981), *Testing unconstrained optimization software,* TOMS, 7, pp. 17-41.

M. R. Osborne and L. P. Sun, (1988), *A new approach to the symmetric rank-one updating algorithm,* Technical Report NMO/01, School of Mathematics, Australian National University.

S. S. Oren and E. Spedicato, (1976), *Optimal conditioning of self-scaling variable metric algorithms,* Mathematical Programming, 10, pp. 70-90.

M. J. D. Powell, (1975), *Convergence properties of a class of minimization algorithms,* in Nonlinear Programming 2, O. Mangasarian, R. Meyer, and S. Robinson, eds., Academic Press, New York, pp. 1-27.

R. B. Schnabel, B. E. Weiss, and J. E. Koontz, (1982), *A modular system of algorithms for unconstrained minimization,* University of Colorado, Department of Computer Science, TR CU-CS-240-82.

Appendix

Table A1: List of Test Functions Numbers and Names.

Number	Dimension	Name
MGH05	2	Beale Function.
MGH07	2	Helical Valley Function.
MGH09	3	Gaussian Function.
MGH12	3	Box 3-Dimensional Function.
MGH14	3	Wood Function.
MGH16	4	Brown and Dennis Function.
MGH18	4	Biggs Exp6 Function.
MGH20	6	Watson Function.
MGH21	9	Extended Rosenbrock Function.
MGH22	10	Extended Powell Singular Function.
MGH23	10	Penalty Function I.
MGH24	10	Penalty Function II.
MGH25	10	Variably Dimensioned Function.
MGH26	10	Trigonometric Function.
MGH35	9	Chebyquad Function.
CGT01	8	Generalized Rosenbrock Function.
CGT02	25	Chained Rosenbrock Function.
CGT04	20	Generalized Singular Function.
CGT05	20	Chained Singular Function.
CGT07	8	Generalized Wood Function.
CGT08	8	Chained Wood Function.
CGT10	30	A Generalized Broyden Tridiagonal Function.
CGT11	30	Another Generalized Broyden Tridiagonal Function.
CGT12	30	Generalized Broyden Banded Function.
CGT14	30	Toint's 7-diagonal generalization of Broyden Tridiagonal Function (see Toint 1978).
CGT16	30	Trigonometric Function (Toint, 1978).
CGT17	8	A Generalized Cragg and Levy Function.
CGT21	30	A Generalized Brown Function.

MGH: problems from Moré, Garbow, and Hillstrom [1981].
CGT: problems from Conn, Gould, and Toint [1987].

Table A2: Iterations and Function Evaluations – Line Search

Function	n	BFGS			SR1			sp
		itrn.	f-eval	rgx	itrn.	f-eval	rgx	
MGH05	2	16	58	0.7E-06	14	52	0.1E-05	1
MGH07	3	26	141	0.4E-05	30	142	0.4E-06	1
MGH09	3	5	34	0.3E-05	3	26	0.2E-07	1
MGH12	3	35	157	0.5E-06	21	99	0.6E-06	1
MGH14	4	32	186	0.7E-05	26	160	0.5E-05	1
MGH16	4	31	183	0.1E-05	21	133	0.3E-07	1
MGH18	6	43	336	0.2E-05	37	302	0.6E-06	1
MGH20	9	95	1020	0.2E-05	46	532	0.8E-05	1
MGH21	10	34	461	0.9E-05	34	462	0.3E-05	1
MGH22	8	45	464	0.7E-05	36	382	0.4E-05	1
MGH23	10	135	1604	0.9E-05	204	2377	0.6E-05	1
MGH24	10	25	358	0.7E-05	25	362	0.8E-05	1
MGH25	10	16	259	0.7E-06	16	259	0.7E-06	1
MGH26	10	27	374	0.3E-05	27	375	0.2E-05	1
MGH35	9	25	320	0.2E-05	25	320	0.3E-06	1
MGH05	2	47	154	0.3E-07	41	139	0.1E-06	10
MGH07	3	29	136	0.6E-06	38	175	0.4E-07	10
MGH09	3	20	98	0.1E-05	17	102	0.3E-06	10
MGH12	3	66	286	0.5E-05	55	259	0.5E-05	10
MGH14	4	58	316	0.6E-05	69	379	0.1E-06	10
MGH16	4	59	322	0.3E-05	37	212	0.1E-05	10
MGH18	6	45	361	0.3E-05	46	369	0.1E-05	10
MGH20	9	95	1020	0.2E-05	46	532	0.8E-05	10
MGH21	10	57	775	0.3E-05	60	813	0.4E-07	10
MGH22	8	88	977	0.9E-05	67	793	0.3E-05	10

Table A2 (continued)

Function	n	BFGS			SR1			sp
		itrn.	f-eval	rgx	itrn.	f-eval	rgx	
MGH23	10	177	2080	0.9E-05	192	2235	0.9E-05	10
MGH25	10	41	535	0.3E-05	23	337	0.3E-05	10
MGH26	10	72	876	0.7E-05	43	560	0.9E-06	10
MGH07	3	31	174	0.4E-06	23	113	0.6E-07	100
MGH14	4	118	625	0.5E-06	104	567	0.5E-05	100
MGH16	4	89	472	0.2E-05	55	303	0.3E-06	100
MGH20	9	95	1020	0.2E-05	46	532	0.8E-05	100
MGH21	10	158	2185	0.8E-05	154	1906	0.5E-06	100
MGH22	8	129	1227	0.4E-05	90	875	0.9E-05	100
MGH25	10	472	5276	0.1E-04	335	3769	0.1E-04	100
CGT01	8	71	707	0.5E-05	81	843	0.4E-06	1
CGT02	25	36	1315	0.7E-05	43	1505	0.6E-05	1
CGT04	20	85	2049	0.9E-05	49	1291	0.5E-05	1
CGT05	20	311	6797	0.8E-05	180	4055	0.9E-05	1
CGT07	8	129	1273	0.3E-05	116	1132	0.4E-06	1
CGT08	8	141	1348	0.5E-05	140	1347	0.1E-05	1
CGT10	30	58	2328	0.9E-05	40	1770	0.7E-05	1
CGT11	30	37	1686	0.3E-05	32	1526	0.8E-05	1
CGT12	30	264	8734	0.6E-05	199	6734	0.5E-05	1
CGT14	30	70	2699	0.5E-05	100	3640	0.9E-05	1
CGT16	10	11	203	0.4E-05	11	204	0.2E-05	1
CGT17	8	134	1269	0.8E-05	92	892	0.3E-05	1
CGT21	20	12	504	0.2E-05	11	483	0.3E-09	1

Table A3: Iterations and Function Evaluations – Trust Region

Function	n	BFGS			SR1			sp
		itrn.	f-eval	rgx	itrn.	f-eval	rgx	
MGH05	2	15	57	0.3E-06	16	68	0.5E-05	1
MGH07	3	27	133	0.1E-05	29	150	0.4E-06	1
MGH09	3	5	38	0.3E-05	3	31	0.2E-07	1
MGH12	3	32	150	0.3E-05	26	146	0.8E-05	1
MGH14	4	46	265	0.4E-07	34	247	0.5E-05	1
MGH16	4	33	188	0.1E-05	20	138	0.7E-05	1
MGH18	6	43	341	0.9E-05	40	344	0.8E-05	1
MGH20	9	88	957	0.3E-05	46	584	0.3E-05	1
MGH21	10	42	555	0.2E-05	49	671	0.2E-06	1
MGH22	8	41	428	0.6E-05	26	294	0.8E-05	1
MGH24	10	24	344	0.2E-05	24	357	0.8E-05	1
MGH25	10	14	236	0.6E-05	14	236	0.6E-05	1
MGH26	10	27	373	0.2E-05	24	349	0.1E-05	1
MGH35	9	24	308	0.4E-05	21	285	0.3E-05	1
MGH05	2	45	160	0.9E-05	36	147	0.9E-06	10
MGH07	3	29	141	0.1E-05	33	171	0.4E-05	10
MGH09	3	21	112	0.8E-05	15	84	0.9E-05	10
MGH12	3	62	292	0.9E-06	19	122	0.7E-05	10
MGH14	4	82	443	0.6E-06	74	467	0.8E-06	10
MGH16	4	59	324	0.5E-06	35	222	0.8E-07	10
MGH18	6	39	323	0.5E-05	51	437	0.6E-07	10
MGH20	9	88	957	0.3E-05	46	584	0.3E-05	10
MGH21	10	63	788	0.3E-05	58	800	0.2E-05	10
MGH22	8	94	913	0.5E-05	56	575	0.8E-05	10

Table A3 (continued)

Function	n	BFGS			SR1			sp
		itrn.	f-eval	rgx	itrn.	f-eval	rgx	
MGH23	10	22	337	0.4E-05	113	1335	0.8E-05	10
MGH24	10	224	2609	0.1E-04	253	3140	0.1E-04	10
MGH25	10	36	488	0.7E-05	25	371	0.3E-05	10
MGH26	10	87	1040	0.7E-05	48	650	0.1E-05	10
MGH07	3	34	158	0.2E-05	22	118	0.2E-05	100
MGH14	4	85	471	0.1E-05	69	426	0.3E-05	100
MGH16	4	89	472	0.4E-06	52	311	0.1E-04	100
MGH20	9	88	957	0.3E-05	46	584	0.3E-05	100
MGH21	10	165	1941	0.2E-05	149	2139	0.3E-06	100
MGH22	8	116	1127	0.8E-05	80	840	0.2E-05	100
CGT01	8	58	584	0.7E-05	80	848	0.8E-05	1
CGT02	25	45	1550	0.4E-05	46	1597	0.2E-05	1
CGT04	20	110	2579	0.3E-05	89	2195	0.5E-05	1
CGT05	20	323	7048	0.5E-05	156	3645	0.8E-05	1
CGT07	8	123	1190	0.4E-05	139	1429	0.3E-06	1
CGT08	8	130	1255	0.9E-05	146	1524	0.5E-05	1
CGT10	30	58	2326	0.9E-05	42	1832	0.7E-05	1
CGT11	30	35	1619	0.3E-05	31	1493	0.5E-05	1
CGT12	30	62	2454	0.8E-05	44	1916	0.5E-05	1
CGT14	30	34	1582	0.8E-05	29	1452	0.5E-05	1
CGT16	10	11	204	0.4E-05	11	206	0.3E-05	1
CGT17	8	83	818	0.9E-05	74	802	0.8E-05	1
CGT21	20	12	504	0.2E-05	11	485	0.3E-09	1

Table A4: Testing Convergence of $\{B_k\}$ to $\nabla^2 f(x_*)$ – Line Search

Function	n	BFGS		SR1	
		itr	$\|\|H_l - B_l\|\|/\|\|H_l\|\|$	itr	$\|\|H_l - B_l\|\|/\|\|H_l\|\|$
MGH05	2	19	0.458E-04	16	0.686E-05
MGH07	3	28	0.274E-04	33	0.175E-06
MGH09	3	9	0.918E+00	4	0.918E+00
MGH12	3	38	0.545E-04	24	0.147E-03
MGH14	4	35	0.830E-02	29	0.154E-04
MGH16	4	34	0.928E-01	23	0.348E-04
MGH18	6	47	0.234E+01	40	0.234E+01
MGH20	9	175	0.105E+00	100	0.264E-02
MGH21	10	35	0.804E-01	34	0.645E-01
MGH22	8	74	0.161E+01	49	0.160E+01
MGH23	10	178	0.167E+04	215	0.167E+04
MGH24	10	348	0.177E-01	330	0.140E-03
MGH25	10	16	0.748E+04	16	0.748E+04
MGH26	10	31	0.689E-01	31	0.468E-01
MGH35	9	28	0.834E+00	26	0.833E+00
CGT01	8	73	0.393E-01	83	0.144E-01
CGT02	25	43	0.570E-01	50	0.317E-01
CGT04	20	500	0.133E+04	500	0.133E+04
CGT05	20	500	0.582E+03	500	0.503E+03
CGT07	8	138	0.691E-01	124	0.111E-01
CGT08	8	147	0.425E-01	146	0.492E-02
CGT10	30	150	0.134E+03	84	0.185E+03
CGT11	30	44	0.781E-01	37	0.448E-01
CGT12	30	273	0.384E+00	210	0.691E-01
CGT14	30	86	0.279E+00	107	0.303E+00
CGT16	10	18	0.466E-01	16	0.385E-03
CGT17	8	216	0.462E+00	125	0.566E-01
CGT21	20	16	0.124E+01	12	0.120E+01

Table A5: Testing Convergence of $\{B_k\}$ to $\nabla^2 f(x_*)$ – Trust Region

Function	n	BFGS		SR1	
		itr	$\|\|H_l - B_l\|\|/\|\|H_l\|\|$	itr	$\|\|H_l - B_l\|\|/\|\|H_l\|\|$
MGH05	2	17	0.235E-02	18	0.102E-05
MGH07	3	30	0.400E-02	31	0.172E-05
MGH09	3	9	0.918E+00	4	0.918E+00
MGH12	3	36	0.396E-02	30	0.473E-02
MGH14	4	47	0.216E-02	41	0.290E-05
MGH16	4	36	0.809E-01	22	0.369E-04
MGH18	6	47	0.234E+01	40	0.234E+01
MGH20	9	157	0.261E-01	99	0.176E-02
MGH21	10	47	0.999E+00	51	0.999E+00
MGH22	8	77	0.277E+01	43	0.276E+01
MGH23	10	500	0.154E+04	149	0.218E+04
MGH24	10	287	0.391E-02	202	0.173E+02
MGH25	10	15	0.103E+05	15	0.103E+05
MGH26	10	31	0.906E-01	28	0.234E-01
MGH35	9	28	0.880E+00	23	0.880E+00
CGT01	8	61	0.110E+00	81	0.275E-01
CGT02	25	51	0.228E+00	50	0.107E+00
CGT04	20	500	0.314E+04	500	0.248E+04
CGT05	20	500	0.104E+04	500	0.671E+03
CGT07	8	122	0.354E-01	138	0.579E-02
CGT08	8	138	0.532E-01	139	0.405E-04
CGT10	30	115	0.109E+03	82	0.112E+03
CGT11	30	40	0.982E-01	34	0.690E-01
CGT12	30	97	0.770E+03	66	0.756E+03
CGT14	30	46	0.220E+00	40	0.160E-01
CGT16	10	16	0.523E-01	15	0.298E-02
CGT17	8	200	0.250E+00	123	0.117E-01
CGT21	20	16	0.124E+01	12	0.120E+01

Table A6: Testing Uniform Linear Independence of $\{s_k\}$ – Line Search

f(x)	n	Itr.	No. of steps so that $\sigma_{min}(\hat{S}_m)* >$							
			10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
MGH05	2	16	3	2	2	2	2	2	2	2
MGH07	3	33	4	3	3	3	3	3	3	3
MGH09	3	4	*	*	*	*	*	*	*	*
MGH12	3	24	14	5	3	3	3	3	3	3
MGH14	4	29	10	5	5	4	4	4	4	4
MGH16	4	23	6	4	4	4	4	4	4	4
MGH18	6	40	*	*	*	*	*	*	*	*
MGH20	9	100	74	70	67	64	63	62	61	60
MGH21	10	34	*	*	*	*	*	*	*	*
MGH22	8	49	*	*	*	*	*	*	*	*
MGH23	10	215	77	77	77	77	77	77	77	77
MGH24	10	330	79	79	79	79	79	79	79	79
MGH25	10	16	*	*	*	*	*	*	*	*
MGH26	10	31	30	16	10	10	10	10	10	10
MGH35	9	26	*	*	*	*	*	*	*	*
CGT01	8	83	26	15	13	13	13	13	13	13
CGT02	25	50	47	28	25	25	25	25	25	25
CGT04	20	500	87	87	87	87	87	87	87	87
CGT05	20	500	87	87	87	87	87	87	87	87
CGT07	8	124	76	76	76	42	34	34	34	34
CGT08	8	146	45	45	45	45	45	45	45	45
CGT10	30	84	*	*	60	34	30	30	30	30
CGT11	30	37	35	33	30	30	30	30	30	30
CGT12	30	210	98	98	88	88	88	88	88	88
CGT14	30	107	59	36	36	36	36	36	36	36
CGT16	10	16	11	10	10	10	10	10	10	10
CGT17	8	125	67	45	42	34	34	34	34	34
CGT21	20	12	*	*	*	*	*	*	*	*

* $\hat{S}_m = [s_l/||s_l||, s_{l-1}/||s_{l-1}||, \ldots, s_{l-m}/||s_{l-m}||]$, where $m \geq n$.

Table A7: Testing Uniform Linear Independence of $\{s_k\}$ – Trust Region

f(x)	n	Itr.	No. of steps so that $\sigma_{min}(\hat{S}_m)* >$							
			10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}
MGH05	2	18	3	2	2	2	2	2	2	2
MGH07	3	31	5	3	3	3	3	3	3	3
MGH09	3	4	*	*	*	*	*	*	*	*
MGH12	3	30	7	6	5	3	3	3	3	3
MGH14	4	41	8	5	4	4	4	4	4	4
MGH16	4	22	5	4	4	4	4	4	4	4
MGH18	6	40	*	*	*	*	*	*	*	*
MGH20	9	99	75	64	63	62	62	61	61	61
MGH21	10	51	*	*	*	*	*	*	*	*
MGH22	8	43	*	*	*	*	*	*	*	*
MGH23	10	149	77	77	77	77	77	77	77	77
MGH24	10	202	79	79	79	74	74	74	74	74
MGH25	10	15	*	*	*	*	*	*	*	*
MGH26	10	28	26	18	10	10	10	10	10	10
MGH35	9	23	*	*	*	*	*	*	*	*
CGT01	8	81	32	17	13	12	12	12	12	12
CGT02	25	50	*	29	26	25	25	25	25	25
CGT04	20	500	88	88	88	88	88	88	88	88
CGT05	20	500	88	87	87	87	87	87	87	87
CGT07	8	138	76	76	50	43	41	41	41	41
CGT08	8	139	41	41	41	41	41	41	41	41
CGT10	30	82	*	*	59	36	32	30	30	30
CGT11	30	34	*	31	30	30	30	30	30	30
CGT12	30	66	*	*	*	60	40	31	30	30
CGT14	30	40	*	33	30	30	30	30	30	30
CGT16	10	15	12	10	10	10	10	10	10	10
CGT17	8	123	73	49	39	34	33	33	33	33
CGT21	20	12	*	*	*	*	*	*	*	*

* $\hat{S}_m = [s_l/\|s_l\|, s_{l-1}/\|s_{l-1}\|, \ldots, s_{l-m}/\|s_{l-m}\|]$, where $m \geq n$.

Table A8: Testing Positive Definiteness – Line Search

$f(x)$	n	Itr.	0: Indefinite ; 1: Positive Definite	%pd	1*	2*
MGH05	2	14	1111111111111	1.00	13	1
MGH07	3	30	11111101111011110111111111111	0.90	12	1
MGH09	3	3	11	1.00	2	1
MGH12	3	21	11111111111111111111	1.00	20	1
MGH14	4	26	1111111011111101111110111	0.88	3	1
MGH16	4	21	10111111111111111111	0.95	18	1
MGH18	6	37	111111100111111111111111011111111111	0.92	11	1
MGH20	9	46	1111011111111111011111011101101111110			
			11111011	0.84	2	1
MGH21	10	34	111011111110111101001111111111111	0.85	13	1
MGH22	8	36	11111101011111111111111111011111111	0.91	9	1
MGH23	10	204	1111111111111111111011111111111101111			
			1110111011011010011010011110111110111			
			1111110110100011111001111111101110011			
			1111010111111011110101001101011111110			
			1111011011111110100110111011111011001			
			11111011111101111110111	0.77	3	0
MGH24	10	25	111111101110111110111111	0.88	6	1
MGH25	10	16	111111111111111	1.00	15	0
MGH26	10	27	11101110111011101101110111	0.77	3	1
MGH35	9	25	111110110111110111111111	0.88	9	1
CGT01	8	81	1111111100110100111101011011111110100			
			1101111110110111011001101110111111011			
			11111111	0.75	10	1
CGT02	25	43	1111111100111111110011011011011011111			
			111111	0.81	11	1
CGT04	20	49	1111111111011111110111111101111111111			
			111111111111	0.94	22	1

1*: Number of consecutive iterations where B_k was positive definite immediately prior to the termination of the algorithm.

2*: Number of iterations where the SR1 update is skipped because condition (4.1) was violated.

Table A8 (continued)

$f(x)$	n	Itr	0: Indefinite ; 1: Positive Definite	%pd	1*	2*
CGT05	20	180	111111111011111011111111111101110111			
			111111111111111010111101111111110111			
			111111110111011010001110111111101111			
			111111111010111111011011111001110111			
			11111111111110111111111111111111111	0.87	21	1
CGT07	8	116	111111111111111110111111101000011011			
			010010011111101011010011011101111011			
			011111111111111110111101111011011111			
			1111111	0.78	13	1
CGT08	8	140	111111101101111101110111111101101101			
			111110011011111101101110011011110100			
			110110000000011110111111001110100111			
			1110110011010011011111101011111	0.70	6	1
CGT10	30	40	111111111111111111111111101111111111			
			001	0.92	1	1
CGT11	30	32	1111011101111111111011101111111	0.87	8	1
CGT12	30	199	111111111110111111110110111011111111			
			111110110111110111011101110111110111			
			011111111110111011111101100111111010			
			110011111111111101010110111111101011			
			101111110011111011111110110011011111			
			110101011101111101	0.80	1	1
CGT14	30	100	111010111110111011101110011110110111			
			111111011101111011011111111010101111			
			111111111111101111111111111	0.83	12	1
CGT16	10	11	1111111111	1.00	10	1
CGT17	8	92	111110111111111101111110111101101111			
			011111100111111111101111101111111101			
			1111111110111111111	0.87	9	1
CGT21	20	11	1110101111	0.80	4	1

1*: Number of consecutive iterations where B_k was positive definite immediately prior to the termination of the algorithm.
2*: Number of iterations where the SR1 update is skipped because condition (4.1) was violated.