Preprint ANL/MCS-P873-0201
Mathematics and Computer Science Division
Argonne National Laboratory

E. Michael Gertz

A Quasi-Newton Trust-Region Method

February 1, 2001 — Revised: January 30, 2003

Abstract. The classical trust-region method for unconstrained minimization can be augmented with a line search that finds a point that satisfies the Wolfe conditions. One can use this new method to define an algorithm that simultaneously satisfies the quasi-Newton condition at each iteration and maintains a positive-definite approximation to the Hessian of the objective function. This new algorithm has strong global convergence properties and is robust and efficient in practice.

Key words. Unconstrained Optimization, Quasi-Newton Methods, Trust-Region Methods, Line-Search Methods.

1. Introduction

Many important problems may be expressed in terms of nonlinear multivariate unconstrained optimization. The basic unconstrained optimization problem is to minimize a real-valued objective function $f(x)$ over all vectors $x \in \mathbb{R}^n$. Many techniques are available for solving unconstrained minimization problems when $f(x)$ is twice continuously differentiable. Quasi-Newton techniques for minimizing $f(x)$ are popular, particularly whenever the matrix-valued second derivative of $f(x)$, called the Hessian and written $\nabla^2 f(x)$, is not known analytically or is prohibitively expensive to compute or store.

As their name suggests, quasi-Newton methods are closely related to Newton's method. Given the current iterate x_k, an unmodified Newton method computes a step s_k by the formula

$$s_k = -\nabla^2 f(x_k)^{-1} \nabla f(x_k)$$

and then takes $x_{k+1} = x_k + s_k$ as its next iterate. Similarly, an unmodified quasi-Newton method takes

$$s_k = -B_k^{-1} \nabla f(x_k) \tag{1}$$

as the step to the next iterate, where $\{B_k\}$ is a sequence of matrix approximation to the Hessian. Like Newton's method, quasi-Newton methods must be modified to enforce convergence and to encourage convergence to a minimizer, rather

---

E. Michael Gertz: Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL 60439, USA (gertz@mcs.anl.gov).

than a maximizer or other point at which $\nabla f(x) = 0$. Furthermore, the simple quasi-Newton iteration given by (1) is not even defined if B_k is singular, and a practical algorithm must handle this case.

Two broad classes of modified quasi-Newton algorithms are trust-region methods and line-search methods. Trust-region methods define each iterate as the approximate minimizer of a relatively simple model function within a region in which the algorithm "trusts" that the model function behaves like f. In their simplest form, line-search methods produce each iterate by searching for an acceptable value of x along a line passing through the previous iterate. The fact that many different line-search methods exist makes it difficult to make a general statement about how the search direction is chosen. Normally, the step will be related to the quasi-Newton step (1), with some modification that preserves convergence if B_k becomes nearly singular. A line-search method often requires more iterations to find a minimizer of f than does a trust-region method. On the other hand, a line-search method tends to compute each iterate more quickly than does a comparable trust-region method.

In this paper, we propose and analyze a new algorithm that combines elements of trust-region methods with elements of line-search methods. The new algorithm retains the quick convergence and stability of trust-region methods, while significantly decreasing the average cost per iteration of the method. The new method, like most trust-region methods, also puts few restrictions on the sequence $\{B_k\}$. In particular, the method will behave correctly if some of the matrices are indefinite or singular. Thus, one can to choose B_k to be $\nabla^2 f(x_k)$, the exact Hessian of the objective function, or one can use a quasi-Newton method such as the symmetric rank-one (SR1) update that does not produce a positive definite sequence $\{B_k\}$.

When B_k is not the exact Hessian, however, there is some advantage to maintaining positive definiteness. If B_k is not positive definite, then the exact solution to the trust-region subproblem will be at a point at which the trust-region radius will be active. Therefore, when B_k is indefinite, a method for solving the trust-region subproblem approximately will not, in general, choose an unmodified quasi-Newton step (1).

Line-search quasi-Newton methods have traditionally also had another advantage over their trust-region counterparts. By employing an appropriate line-search strategy and an appropriate updating strategy for B_k, one can generate a positive definite sequence of approximations to the Hessian of $f(x)$. In contrast, trust-region methods have traditionally had to modify B_k in ways that make it a less accurate model of the Hessian of $f(x)$ in order to maintain a positive definite sequence of approximate Hessians. We describe some of these modifications, and the sense in which they make the approximate Hessian less accurate, in Section 3.

The algorithm proposed in this paper, on the other hand, provides a natural way of modifying the approximate Hessian to produce a positive definite sequence of matrices. We provide a general convergence theory for the algorithm and show how the algorithm, in conjunction with the BFGS update (5), may be used to produce an efficient quasi-Newton method.

2. Trust-Region Methods

Trust-region methods produce a trial step by minimizing a quadratic model of the objective function subject to a constraint on the length of the trial step. Because of this restriction, trust-region methods are sometimes known as restricted-step methods. In this section, we summarize some properties of trust-region methods. For an in-depth overview of trust-region methods see Conn, Gould, and Toint [2].

A quadratic model of $f(x_k + s) - f(x_k)$ takes the form

$$\mathcal{Q}_k(s) = g_k^T s + \tfrac{1}{2} s^T B_k s, \tag{2}$$

where $s = x - x_k$. Typically, g_k is chosen to be $\nabla f(x_k)$ or a close approximation to the gradient. In this paper, we always choose $g_k = \nabla f(x_k)$, and we write $g(x)$ for $\nabla f(x)$. Trust-region methods put few restrictions on the choice of B_k. In Section 3 we describe how to choose B_k to be a quasi-Newton approximation to the Hessian. The trust-region subproblem may be formally stated as follows:

$$\begin{array}{l} \text{minimize} \quad \mathcal{Q}_k(s) \\ \text{subject to } \|N_k s\| \leq \delta_k, \end{array} \tag{3}$$

where δ_k is a scalar known as the trust-region radius, $\|\cdot\|$ is any vector norm, and N_k is a nonsingular matrix used to scale the problem.

No consensus exists on what choice of N_k is appropriate. Convergence theory allows the choice $N_k = I$ for all k, and this choice seems to be acceptable for well-scaled problems. Choosing N_k to be a constant matrix in order to improve the scaling of the problem is a trivial modification to a trust-region algorithm. Some methods for solving problem (3), however, must vary N_k at each iteration in order to compute the trial step efficiently (see, for instance, Steihaug [15] and Toint [18])

Our new quasi-Newton method always chooses B_k to be positive definite. We also choose $\|\cdot\|$ to be $\|\cdot\|_\infty$ and N_k to be diagonal. With these choices, the trust-region subproblem (3) is a convex quadratic program (QP) with only simple bounds on the variables. The minimizer is unique, and the subproblem can be solved by off-the-shelf software. Possibly the first method to use a quadratic model of the objective function with the trust-region based on $\|\cdot\|_\infty$ was described by Wilson [19]. More modern methods were studied by Fletcher [3], who maintains a positive definite B_k, and by Friedlander et al. [5], who define a method that finds a direction of sufficient decrease by solving a simplified QP.

3. Quasi-Newton Methods

We use a quasi-Newton updating scheme to define the matrices B_k in our quadratic model (2). Quasi-Newton methods use the curvature information from the current iteration, and possibly the matrix B_k to define B_{k+1}. A true quasi-Newton method will choose B_{k+1} so that

$$g_{k+1} - g_k = B_{k+1}(x_{k+1} - x_k). \tag{4}$$

In this way, $B_{k+1}(x_{k+1}-x_k)$ is a finite difference approximation to the derivative of $g(x)$ in the direction of $x_{k+1} - x_k$. For a practical quasi-Newton method, computing B_{k+1} should be considerably less expensive than computing $\nabla^2 f(x)$. Popular quasi-Newton methods choose $B_{k+1} = B_k + E$, where E is a matrix of low rank, usually one or two.

By using a rank-two update, we may also arrange that B_k is always a positive-definite, symmetric matrix. Many rank-two formulas may be used, but probably the most famous is the BFGS update,

$$B_{k+1} = B_k - \frac{(B_k p_k)(B_k p_k)^T}{p_k^T B_k p_k} + \frac{y_k y_k^T}{y_k^T p_k}, \tag{5}$$

where $p_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$. It is well known (see, for instance, Fletcher [4]) that if B_k is positive definite and $y_k^T p_k > 0$ and B_{k+1} is chosen using the BFGS update, then B_{k+1} is also positive definite.

We note that for any update satisfying the quasi-Newton condition (4), the matrix B_{k+1} cannot be positive definite if $y_k^T p_k < 0$, because $y_k^T p_k = p_k^T B_{k+1} p_k$. Typically, quasi-Newton methods that use the BFGS update employ a line search to locate a point for which $y_k^T p_k > 0$. In their simplest form, these methods will generate a search direction s_k for which $g_k^T s_k \leq 0$ and then search for an positive α_k that satisfies the well-known strong Wolfe conditions,

$$f(x_k + \alpha_k s_k) - f(x_k) \leq \alpha_k \eta_1 g_k^T s_k \tag{6}$$

and

$$|g(x_k + \alpha_k s_k)^T s_k| \leq -\omega g_k^T s_k, \tag{7}$$

where $0 < \eta_1 < \omega < 1$. Usually, one also requires that $\eta_1 < 1/2$ so that the Wolfe condition (6) is met by the exact minimizer of a quadratic function. One then takes $x_{k+1} = x_k + \alpha_k s_k$. We observe that s_k, α_k, and p_k are related by the rule

$$x_{k+1} - x_k = p_k = \alpha_k s_k.$$

If x_{k+1} satisfies the Wolfe condition on the gradient (7), then

$$(g_{k+1} - g_k)^T p_k \geq -(1-\omega) g_k^T p_k, \tag{8}$$

and therefore $y_k^T p_k > 0$. Thus, when paired with the BFGS update, a line search using the Wolfe conditions will produce a positive-definite sequence of matrices B_k, and the quadratic terms may be dropped. Traditional trust-region methods, which do not employ the Wolfe conditions, cannot ensure that $y_k^T p_k > 0$ at each iteration. Thus, traditional trust-region algorithms, typically involve iterations for which no symmetric positive definite B_{k+1} satisfies the quasi-Newton condition (4).

When the Wolfe conditions are not used to define x_{k+1}, the BFGS update may still be used to define B_{k+1} when $y_k^T s_k > 0$. Nocedal and Yuan [11] suggest simply setting $B_{k+1} = B_k$ when $y_k^T s_k < 0$. Obviously, their method will not satisfy the quasi-Newton condition (4) at each iteration, but it will keep B_k positive definite. Some care must be taken in applying the update, because when

the Wolfe conditions are not used, $y_k^T s_k$ may equal zero or may be arbitrarily close to zero. Fletcher [3] develops a trust-region method that employs a rank-two formula suggested by Powell [12] when $y_k^T s_k \leq 0$. As we noted above, his method cannot satisfy the quasi-Newton condition.

Nocedal and Yuan [11] also experiment with the symmetric rank-one update

$$B_{k+1} = B_k + \frac{(y_k - B_k p_k)(y_k - B_k p_k)^T}{(y_k - B_k p_k)^T p_k}, \tag{9}$$

which is the unique rank-one update that maintains a symmetric sequence of matrices. The matrices generated by this quasi-Newton scheme are not necessarily positive definite. To avoid numerical problems, the authors skip the update if $|(y_k - B_k p_k)^T p_k|$ is less than some tolerance. They allow B_k to become indefinite and use a method similar to Moré and Sorensen's [10] to compute the approximate global minimizer. We are not aware of anyone using the BFGS update to define B_{k+1} when $y_k^T s_k < 0$. We see no technical reason why this could not be done in the context of trust-region methods, provided that proper care is taken when $s_k^T B_k s_k$ or $y_k^T s_k$ is nearly zero. On the other hand, we see no compelling reason to use the BFGS update unless one wishes to generate a sequence of positive definite matrices.

With low-rank quasi-Newton methods, it is not necessary to factor an $n \times n$ matrix at each iteration to solve the trust-region subproblem. Gill et al. [7] show how to update a Cholesky factorization of B_k to find a Cholesky factorization of B_{k+1} when B_{k+1} is a low-rank modification of B_k. If B_0 is chosen to be a diagonal matrix, no factorization of B_k is ever required. The process of updating a factorization of B_k requires $O(n^2)$ multiplications for a rank-one modification and $O(2n^2)$ multiplications for a rank-two modification. This requirement is clearly preferable to the $O(\frac{1}{6}n^3)$ multiplications needed to factor a dense, symmetric B_{k+1}. In our numerical experiments, we used the LSSOL algorithm of Gill et al. [8], which is based on an algorithm by Stoer [16]. This code is specifically designed for positive semidefinite B_k and can efficiently use a factorization of B_k, rather than B_k itself, to find a minimizer.

4. Combination Trust-Region Line-Search Methods

The trust-region subproblem, repeated here for convenience, is

$$\begin{array}{l} \text{minimize} \quad \mathcal{Q}_k(s) \\ \text{subject to } \|N_k s\| \leq \delta_k. \end{array} \tag{10}$$

We observe that the trial step s_k produced by solving the trust-region subproblem is often a suitable search direction for a line-search technique. Furthermore, these line-search techniques may use a variant of the strong Wolfe conditions,

$$f(x_k + \alpha_k s_k) - f(x_k) \leq \eta_1 \left(\alpha_k g_k^T s_k + \tfrac{1}{2}\alpha_k^2 \min(0, s_k^T B_k s_k)\right) \tag{11}$$

and

$$|g(x_k + \alpha_k s_k)^T s_k| \leq -\omega(g_k^T s_k + \alpha_k \min(0, s_k^T B_k s_k)), \tag{12}$$

as their termination criteria. When B_k is positive semidefinite, these conditions reduce to the traditional strong Wolfe conditions (6) and (7), which allow combination algorithms to fully exploit the properties of the BFGS update. Furthermore, combination methods may be applied even when B_k is not positive definite. Gertz [6] shows that when B_k is chosen to be the exact Hessian, the quadratic terms in (11) and (12) can be used to ensure second-order convergence of these methods.

To simplify notation, we introduce the function

$$q_k(s) = g_k^T s + \tfrac{1}{2}\min(0, s^T B_k s). \tag{13}$$

In terms of this function, the first Wolfe condition (11) may be written more succinctly as

$$f(x_k) - f(x_k + \alpha s_k) \geq -\eta_1 q_k(\alpha s_k). \tag{14}$$

We also define the set

$$\mathcal{W}(\eta_1, \omega) = \{\alpha > 0 \mid \alpha \text{ satisfies conditions (11) and (12)}\}. \tag{15}$$

This notation makes clear the dependence of the set of α's that satisfy the Wolfe conditions on the parameters η_1 and ω.

This paper develops new algorithms, first introduced by Gertz [6], that apply a line search to the trust-region trial step. These new methods apply rules defined to favor iterations of the form $x_{k+1} = x_k + s_k$, because the trust-region step often produces a very good reduction in f. When $f(x_k + s_k)$ does not meet a sufficient decrease condition, the method performs a line search to find an α_k so that $f(x_k + \alpha_k s_k) - f(x_k)$ is sufficiently small when compared with $\mathcal{Q}_k(\alpha_k s_k)$. In this manner, every time a trial step is computed, the new algorithms produce a new value of x that meets a sufficient decrease requirement. Classical trust-region methods, by contrast, may require several solutions of the relatively expensive trust-region subproblem before they find an acceptable new value of x. The length $\alpha_k \|N_k s_k\|$ of the accepted step provides useful information for controlling the size of the trust-region radius. We have used this length extensively and effectively in adjusting δ_k.

Toint [18] describes a method that also adds a line search to a trust-region method. His method is based on the observation that one may substitute $\hat{x}_k$ for x_k where $f(\hat{x}_k) < f(x_k)$ at any point in a trust-region method without adversely affecting the proof that $\liminf_{k\to\infty} \|g_k\| = 0$. Toint therefore performs a line search at every iterate to attempt to find a lower value of the objective function. His method does not impose a sufficient decrease condition on the line search. Nocedal and Yuan [11] describe a method that performs a backtracking line search when $f(x_k + s_k) \geq f(x_k)$. They also do not impose a sufficient decrease condition on this line search, and moreover they accept $f(x_k + s_k)$ whenever $f(x_k + s_k) \leq f(x_k)$. Thus, they do not impose a sufficient decrease requirement on any step. Gertz [6] also introduces and develops a trust-region method with a backtracking line search. The backtracking method employs a sufficient decrease criterion at every iteration and has strong convergence properties. However, this method is less useful in developing quasi-Newton algorithms than are the

methods described here because it does not generate steps that satisfy the Wolfe conditions (11) and (12). For a method that adds a backtracking line search to a trust-region method for equality-constrained optimization, see El Hallabi [9].

The convergence theorems presented in Gertz [6] are stronger than those presented in the papers discussed in the preceding paragraph. Since these authors do not impose a sufficient decrease requirement on each step, the techniques used to prove the stronger convergence theorems are not available for their methods. We doubt that the stronger theorems hold. Nocedal and Yuan do make limited use of the line search to control the trust-region radius, in the sense that when $\alpha_k s_k$ is determined by the backtracking line search and $\alpha_k \|N_k s_k\|$ is small enough, they set $\delta_k = \alpha_k \|N_k s_k\|$. Our use of the step length to adjust the trust-region radius is more sophisticated.

The theory developed by Nocedal and Yuan is limited to the case in which $\|\cdot\| = \|\cdot\|_2$, whereas the theory developed here is not. Furthermore, they base their algorithm on a method of solving the trust-region subproblem developed by Moré and Sorensen [10] but omit a key step in this method. The omitted step, which computes an approximate null vector of a certain matrix, often allows Moré and Sorensen's method to find a solution to the trust-region subproblem far more quickly than their simplified method. Nocedal and Yuan, in fact, use Moré and Sorensen's method in their numerical experiments but are vague about the theoretical basis of their using the method. The omission of the step from Moré and Sorensen's algorithm is not trivial. It affects the theoretical properties of s_k significantly.

Neither the method of Toint nor the method of Nocedal and Yuan necessarily finds steps for which $(g_{k+1} - g_k)^T s_k > 0$. Thus, they are limited in their ability to fully exploit the properties of the BFGS update in their methods.

5. The Biased Wolfe Trust-Region Algorithm

Line-search methods based on the Wolfe conditions have a desirable property not found in backtracking methods: it is possible to find acceptable values of α that are greater than one. If the method chooses an $\alpha > 1$, then either the quadratic model was unduly pessimistic, or the trust-region bound was active and the trust-region radius was too small. In either case, the step $\alpha_k s_k$ still satisfies a sufficient decrease condition, and generally $f(x_k + \alpha_k s_k) < f(x_k + s_k)$. Since it is possible, and sometimes necessary, that $\alpha > 1$, we may define a trust-region method that relies exclusively on $\alpha_k \|N_k s_k\|$ to adjust the trust-region radius.

Algorithm 5.1. *Wolfe Trust-Region Algorithm*
Let constants $k_{\max}$, $\nu \geq 1$ and $0 < \eta_1 < \omega < 1$ be given.
$k \leftarrow 1$; $\delta_k \leftarrow 1$; $x_k \leftarrow x_0$
while $k \leq k_{\max}$ and not converged
  Let s_k be an approximate solution to the trust-region subproblem (3).
  Choose $\alpha_k \in \mathcal{W}(\eta_1, \omega)$.
  $x_{k+1} \leftarrow x_k + \alpha_k s_k$

Choose $\delta_{k+1} \in [\alpha_k \|N_k s_k\|, \alpha_k \nu \|N_k s_k\|]$.
$k \leftarrow k + 1$
end

The introduction of the constant ν merits some explanation. The trial step s_k is not computed exactly. Thus, even when all minimizers of the trust-region subproblem occur on the boundary, s_k may be computed only such that $\|N_k s_k\|$ is within some tolerance of δ_k. The value of this tolerance depends on the method used solve the subproblem but is often significantly greater than machine precision. We may always arrange so that $\|N_k s_k\| \leq \delta_k$, but we cannot practically require that $\|N_k s_k\| = \delta_k$. The constant ν is used to compensate for this effect. Assume that $\|N_k s_k\| \geq (1/\nu)\delta_k$ unless $s_k = -B_k^{-1} g$. An acceptable rule for choosing $\delta_{k+1} \in [\alpha_k \|N_k s_k\|, \alpha_k \nu \|N_k s_k\|]$ is

$$\delta_{k+1} = \begin{cases} \alpha_k \|N_k s_k\| & \text{if } s_k = -B_k^{-1} g_k \\ \alpha_k \delta_k & \text{otherwise.} \end{cases} \tag{16}$$

Other reasonable choices depend on the algorithm used to solve the trust-region subproblem.

Gertz [6] has shown that Algorithm 5.1 is theoretically sound and possesses a fast asymptotic convergence rate. Practical problems arise, however, when this algorithm is paired with usual methods of performing the line search. The rule Algorithm 5.1 used to adjust the trust-region radius can lead to unnecessarily small choices of δ_{k+1}. Moreover, the constants η_1 and ω used in the Wolfe conditions are often chosen to produce a rather relaxed search. Because $\alpha_k = 1$ is usually the first value tried by line search algorithm, Algorithm 5.1 is not aggressive in increasing the trust-region radius, even when an increase is permissible. Away from a solution, unnecessarily small values of δ_k can cause the algorithm to take unnecessarily small steps. Numerical experiments show an increase in the number of iterations for some problems. More significant, however, is the fact that some methods for solving the trust-region subproblem (3) appear to be adversely affected by an unnecessarily small choice of δ_k. Since the solution of (3) can be the computational bottleneck of a trust-region algorithm, it is important to solve the trust-region subproblem as efficiently as possible.

Algorithm 5.1 is theoretically interesting because of its simplicity, but for practical use it must be modified to "bias" it against decreasing the trust-region radius. Rather than modifying the line search to cause it to search more aggressively for $\alpha > 1$, we have chosen the simple scheme outlined below.

Algorithm 5.2. *Biased Wolfe Trust-Region Algorithm*
Let constants $k_{\max}$, $0 < \alpha_{\min} < 1$, $0 < \eta_1 < \eta_2 < 1$ and $\gamma_3 > \nu \geq 1$ be given.
$k \leftarrow 1$; $\delta_k \leftarrow 1$; $x_k \leftarrow x_0$
while $k \leq k_{\max}$ and not converged
Let s_k be an approximate solution to the trust-region subproblem (3).
$\rho_k \leftarrow (f(x_k + s_k) - f(x_k)) / q_k(s_k)$
Choose $\alpha_k \in \mathcal{W}(\eta_1, \omega)$.
$x_{k+1} \leftarrow x_k + \alpha_k s_k$

if $\rho_k \geq \eta_2$ **and** $\alpha_k \geq \alpha_{\min}$ **then**
Choose $\hat{\nu}_k \in [1, \nu]$.
$\delta_{k+1} = \max(\delta_k, \alpha_k \hat{\nu}_k \|N_k s_k\|, \gamma_3 \|N_k s_k\|)$
else
Choose $\delta_{k+1} \in [\alpha_k \|N_k s_k\|, \alpha_k \nu \|N_k s_k\|]$.
end if
$k \leftarrow k + 1$
end

The scalar ν_k may be chosen implicitly at each iteration by a rule such as (16). One would first use an appropriate rule to choose a trust-region radius $\hat{\delta}_k$ in the interval $[\alpha_k \|N_k s_k\|, \alpha_k \nu \|N_k s_k\|]$ and then take $\hat{\nu}_k = \hat{\delta}_k / \|N_k s_k\|$. It is significantly more notationally convenient to refer to the sequence of scalars $\{\hat{\nu}_k\}$ than to the sequence $\{\hat{\delta}_k\}$. Thus the notation $\hat{\delta}_k$ will not be used in the sequel.

This pseudocode has two subtle details. The first is that if $\rho_k \geq \eta_2$ and $\alpha_k \geq \alpha_{\min}$, then $\delta_{k+1} \geq \delta_k$, but this is not the only condition under which the trust-region radius may increase. Even if $\rho_k \leq \eta_2$, we may still find $\alpha_k > 1$ and increase the trust-region radius. The test $\rho_k \geq \eta_2$ exists only to bias the algorithm toward choosing a larger trust-region radius. The second detail is that Algorithm 5.2 can take small steps without reducing the trust-region radius. Specifically, if $\rho_k \geq \eta_2$ and $\alpha_k \geq \alpha_{\min}$, then the trust-region radius will not decrease, but $\alpha_k \|N_k s_k\|$ can be significantly smaller than δ_k.

Without further conditions, it appears impossible to prove that Algorithm 5.2 does not converge to a point that is not stationary. One need also require that the algorithm for performing the line search find an α_k that satisfies

$$f(x_k + \alpha_k s_k) - f(x_k) - \eta_1 q_k(\alpha_k s_k) \leq f(x_k + s_k) - f(x_k) - \eta_1 q_k(s_k). \quad (17)$$

A common method of finding an α_k that satisfies conditions (11) and (12) is to apply a safeguarded univariate minimization routine to the function

$$\phi_k(\alpha) = f(x_k + \alpha s_k) - f(x_k) - \eta_1 q_k(\alpha s_k). \quad (18)$$

One can safely assume that this routine uses $\alpha = 1$ as its first iterate and produces a nonincreasing sequence of objective values. Our assumption (17) is merely the statement that $\phi_k(\alpha_k) \leq \phi_k(1)$, which is simple to achieve in practice.

It is not immediately clear that a line search based on minimizing $\phi_k(\alpha)$ will find a value of α_k satisfying both (11) and (12). In fact, without some further conditions on B_k there may be no such α_k. In Lemma 2 we will give conditions under which α_k may be computed. Before we state and prove Lemma 2, however, we must discuss conditions on s_k that a reasonable algorithm will meet.

6. First-Order Convergence of Trust-Region Methods

Under mild assumptions, one can prove that trust-region algorithms produce a sequence of iterates $\{x_k\}$ for which $\lim_{k\to\infty} \|g_k\| = 0$. Since the approximation to the first derivative of f converges to zero, such proofs are often called proofs of

first-order convergence. These proofs are global convergence proofs in the sense that it is not necessary to assume that some x_k is sufficiently close to a stationary point, but only to assume that all iterates lie in a region in which $f(x)$ is bounded below. Unfortunately, it is not usually possible to produce global proofs that the sequence of iterates converges. Proofs that $\{x_k\}$ converges (explicitly or implicitly) generally assume that some iterate is sufficiently close to an isolated local minimizer.

In the following sections, we prove first-order convergence results for Algorithm 5.2. These convergence theorems do not require that B_k be positive definite. We present the more general theory because Algorithm 5.2 is not restricted to methods that use the BFGS update; because the general results are not significantly more difficult to prove; and because the general results demonstrate that the algorithm remains robust in the presence of near-singularity or indefiniteness resulting from numerical error.

The earliest proofs of first-order convergence are due to Powell [13,14], who proved that $\liminf_{k\to\infty} \|g_k\| = 0$. Thomas [17] extended this result, proving that under additional conditions, $\lim_{k\to\infty} \|g_k\| = 0$. Thomas's proof, however, relies heavily on Powell's result. Powell's theorem is remarkable, not only in that it requires weak assumptions on f, but also in that his proof provides a general framework for proving the convergence of trust-region algorithms. To fit within this framework, an algorithm must have the following two properties:

P1. If $f(x_k)$ is bounded below and $\|g_k\|$ is bounded away from 0, then $\delta_k \to 0$ and $\{x_k\}$ converges.
P2. If $\|g_k\|$ is bounded away from 0 and $\{x_k\}$ converges, then $\delta_k \not\to 0$.

It follows immediately that for any algorithm satisfying P1 and P2, either $f(x_k)$ is unbounded below or $\liminf_{k\to\infty} \|g_k\| = 0$.

Proofs of convergence for trust-region methods require that we specify conditions on the step s_k. The first condition specifies how accurate s_k must be as a solution to the trust-region subproblem. It is wholly unrealistic to require that s_k be an exact solution. Possibly the weakest requirement is proposed by Powell, who requires that for some constant $\tau > 0$,

$$-\mathcal{Q}_k(s_k) \geq \tau \|g_k\| \min(\delta_k/\|N_k\|, \|g_k\|/\|B_k\|) \quad \text{and} \quad \|N_k s_k\| \leq \delta_k. \tag{19}$$

This criterion was motivated by the following lemma.

Lemma 1 (Powell). *Let κ be a constant such that for all s, $\|s\|_2 \geq \kappa\|s\|$. Then*

$$\min_{\alpha}\{\mathcal{Q}_k(-\alpha g_k) \mid \|N_k(\alpha g_k)\| \leq \delta_k\} \leq -\tfrac{1}{2}\kappa^2 \|g_k\| \min\left(\frac{\delta_k}{\|N_k\|}, \frac{\|g_k\|}{\|B_k\|_2}\right). \tag{20}$$

Proof. See Powell [13,14]. □

Since the minimum value of $\mathcal{Q}_k(s)$ within the trust region is at least as small as the constrained minimum value along the steepest descent direction, the lemma provides an upper bound on the constrained minimum value of $\mathcal{Q}_k(s)$. Notice, however, that the lemma also yields an upper bound for the choice of τ in (19). Because $\|B_k\|_\infty \geq \|B_k\|_2$, a comparison of inequalities (19) and (20)

shows that τ must be chosen less than $\frac{1}{2}\kappa^2$, where κ depends on the choice of norm. For the infinity and Euclidean norms, $\kappa = 1$; for the one-norm, $\kappa = 1/\sqrt{n}$. We assume henceforth without comment that τ is chosen sufficiently small.

The second condition that must be placed on s_k is more of a theoretical annoyance than a practical problem. Powell's criterion (19) is not sufficient to ensure that $g_k^T s_k \leq 0$, a condition that is needed to ensure that the line search will be successful. Clearly, if $g_k^T s_k > 0$, then $\mathcal{Q}(-s_k) < \mathcal{Q}(s_k)$, and thus $-s_k$ is a better approximation to a minimizer of $\mathcal{Q}(s)$ than is s_k. Therefore, even if our algorithm for solving the trust-region subproblem (10) produced an s_k such that $g_k^T s_k > 0$, we could simply negate the step. Such an action, however, is usually not necessary, because methods for solving this subproblem tend to be defined in such a way that they always generate steps for which $g_k^T s_k \leq 0$. See Conn, Gould, and Toint [2] for a survey of methods for finding an approximate solution of (10).

It is not, however, possible to prove in general that $g_k^T s_k$ is not zero. In particular, if B_k is indefinite and $\|\cdot\|$ is the infinity norm, then there is the unlikely possibility that the global minimizer, $s_k^\star$, of the trust-region subproblem might occur at a point for which $g_k^T s_k^\star = 0$ even if $g_k \neq 0$. Thus, for the line search to be successful, we must assume that the second-order information B_k is a sufficiently accurate representation of $\nabla^2 f(x_k)$ whenever $g_k^T s_k = 0$.

We note that this is not an issue for quasi-Newton methods that maintain a positive definite approximation to the Hessian. When B_k is positive definite, $0 > \mathcal{Q}_k(s_k) > g_k^T s_k$. Nonetheless, we present the following lemma giving general conditions under which the line search will be successful.

Lemma 2. *Let $f(x)$ be twice continuously differentiable and bounded below. Let $g(x) = \nabla f(x)$, and denote $g_k = g(x_k)$ for all indices k. If $g_k^T s_k \leq 0$ and furthermore*

$$s_k^T \nabla^2 f(x_k) s_k < \eta_1 s_k^T B_k s_k < 0 \text{ whenever } g_k^T s_k = 0, \tag{21}$$

then there are constants $0 < \alpha_l < \alpha_h$ such that α satisfies the quadratic Wolfe conditions (11) and (12) whenever $\alpha_l \leq \alpha \leq \alpha_h$.

Proof. Consider the function

$$\phi_k(\alpha) = f(x_k + \alpha s_k) - f(x_k) - \eta_1 \alpha g_k^T s_k - \tfrac{1}{2}\eta_1 \alpha^2 \min(0, s_k^T B_k s_k),$$

first introduced in (18). By assumption, $f(x_k + \alpha_k s_k)$ is bounded below, and $g_k^T s_k \leq 0$ with $s_k^T B_k s_k$ negative when $g_k^T s_k = 0$. Therefore $\lim_{\alpha\to\infty} \phi_k(\alpha) = \infty$. The first and second derivatives of ϕ_k at zero are

$$\phi_k'(0) = (1 - \eta_1) g_k^T s_k \quad \text{and} \quad \phi_k''(0) = s_k^T \nabla^2 f(x_k) s_k - \eta_1 \min(0, s_k^T B_k s_k).$$

If $g_k^T s_k$ is negative, then so is $\phi_k'(0)$. If, on the other hand, $g_k^T s_k = 0$, then $s_k^T B_k s_k$ is negative, and thus $\phi_k'(0) = 0$ and $\phi_k''(0) < 0$. In either case, because $\phi_k(0)$ equals zero, $\phi_k(\alpha)$ is negative for all sufficiently small positive α. Because there are $\alpha > 0$ with $\phi_k(\alpha) < 0$ but $\phi_k(\alpha)$ is continuous and $\lim_{\alpha\to\infty} \phi_k(\alpha) = \infty$, there is a $\beta > 0$ with $\phi_k(\beta) = 0$. Let α^* be the global minimizer of $\phi_k(\alpha)$ in $[0, \beta]$. The

minimum value cannot occur at the endpoints because $\phi_k(0) = \phi_k(\beta) = 0$, but there are α in $[0, \beta]$ with $\phi_k(\alpha) < 0$. Thus α^* is also a local minimizer of $\phi_k(\alpha)$, and $\phi_k(\alpha^*) < 0$.

Let $\bar{\alpha}$ be any local minimizer of $\phi_k(\alpha)$ with $\phi_k(\bar{\alpha}) < 0$. Then

$$f(x_k + \bar{\alpha}s_k) - f(x_k) - \eta_1(\bar{\alpha}g_k^T s_k + \tfrac{1}{2}\bar{\alpha}^2 \min(0, s_k^T B_k s_k)) < 0,$$

so

$$f(x_k + \bar{\alpha}s_k) - f(x_k) < \eta_1(\bar{\alpha}g_k^T s_k + \tfrac{1}{2}\bar{\alpha}^2 \min(0, s_k^T B_k s_k)).$$

Therefore $\bar{\alpha}$ satisfies the first Wolfe condition (11). Because $\bar{\alpha}$ is a local unconstrained minimizer, $\phi_k'(\bar{\alpha}) = 0$. In other words

$$g(x_k + \bar{\alpha}s_k)^T s_k = \eta_1(g_k^T s_k + \bar{\alpha} \min(0, s_k^T B_k s_k)).$$

The right-hand side of the preceding equation is clearly negative, and thus $g(x_k + \bar{\alpha}s_k)^T s_k < 0$. Because $\omega > \eta_1$, we find

$$-\omega(g_k^T s_k + \bar{\alpha} \min(0, s_k^T B_k s_k)) > -g(x_k + \bar{\alpha}s_k)^T s_k = |g(x_k + \bar{\alpha}s_k)^T s_k|,$$

which shows that $\bar{\alpha}$ satisfies the second Wolfe condition (12). The existence of an appropriate interval then follows immediately from the existence of a local minimizer and the continuity of $\phi_k(\alpha)$. □

A well-defined univariate minimization routine will therefore find an appropriate α_k. We assume that, in any implementation of Algorithm 5.2, the method used to solve the trust-region subproblem will find an s_k at each iteration that satisfies Powell's sufficient decrease criterion (19) and the conditions of Lemma 2. We discuss one such method in Section 8.

We now show that Algorithm 5.2 has the property P2.

Lemma 3. *Let $\{x_k\}$ be a sequence of iterates produced by Algorithm 5.2, let $\Omega \subset \mathbb{R}^n$ be a region containing $\{x_k\}$ in its interior, and let $f(x)$ be twice continuously differentiable and bounded below in Ω. Assume further that $\|B_k\|$, $\|N_k\|$ and $\|N_k^{-1}\|$ are all bounded above. Then if $\{\|g_k\|\}$ is bounded away from zero and $\{x_k\}$ converges, $\delta_k \not\to 0$.*

Proof. Because $x_{k+1} = x_k + \alpha_k s_k$, we may denote $g(x_k + \alpha_k s_k)$ as g_{k+1}. The second Wolfe condition (12) requires that

$$-|g_{k+1}^T s_k| \geq \omega(g_k^T s_k + \alpha_k \min(0, s_k^T B_k s_k)).$$

The value $-|g_{k+1}^T s_k|$ is bounded above by $g_{k+1}^T s_k$, and $\min(0, s_k^T B_k s_k)$ is bounded below by $-|s_k^T B_k s_k|$. Therefore

$$g_{k+1}^T s_k + \alpha_k \omega |s_k^T B_k s_k| \geq \omega g_k^T s_k.$$

When $-g_k^T s_k$ is subtracted from both sides of this inequality, it becomes

$$(g_{k+1} - g_k)^T s_k + \alpha_k \omega |s_k^T B_k s_k| \geq -(1 - \omega) g_k^T s_k. \tag{22}$$

If $\delta_k \to 0$ but $\{\|g_k\|\}$ is bounded away from zero, then $\delta_k/\|N_k\| \leq \|g_k\|/\|B_k\|$ for sufficiently large values of k. We subtract $\frac{1}{2}(1-\omega)s_k^T B_k s_k$ from the right side of (22), add its absolute value to the left side, and apply the sufficient decrease condition (19) to obtain

$$(g_{k+1} - g_k)^T s_k + (\alpha_k\omega + \tfrac{1}{2}(1-\omega))|s_k^T B_k s_k| \geq \tau(1-\omega)\frac{\|g_k\|}{\|N_k\|}\delta_k.$$

Let $\|\cdot\|_T$ denote the norm dual to $\|\cdot\|$. In other words $\|x\|_T = \max_{\|y\|=1} |x^T y|$. Let $\kappa_1 > 0$ be a constant such that for all x, $\|x\|_T \leq \kappa_1\|x\|$. It then follows from norm inequalities that

$$\|g_{k+1} - g_k\|_T\|s_k\| + \kappa_1(\alpha_k\omega + \tfrac{1}{2}(1-\omega))\|s_k\|^2\|B_k\| \geq \tau(1-\omega)\frac{\|g_k\|}{\|N_k\|}\delta_k.$$

Using the relationship $\|s_k\| \leq \|N_k^{-1}\|\delta_k$ and canceling wherever possible, we find

$$\|g_{k+1} - g_k\|_T\|N_k^{-1}\| + (\kappa_2\alpha_k + \kappa_3)\|N_k s_k\| \geq \tau(1-\omega)\frac{\|g_k\|}{\|N_k\|}, \tag{23}$$

where κ_2 and κ_3 are constants such that

$$\kappa_2 \geq \kappa_1\omega\|B_k\|\|N_k^{-1}\|^2 \text{ and } \kappa_3 \geq \tfrac{1}{2}\kappa_1(1-\omega)\|B_k\|\|N_k^{-1}\|^2.$$

Now, if $\delta_k \to 0$, then both $\|N_k s_k\|$ and $\alpha_k\|N_k s_k\| \leq \delta_{k+1} \to 0$. Since $\{x_k\}$ converges, the iterates eventually lie in a compact region, and within this region $g(x)$ is uniformly continuous. Thus, it is clear that the left-hand side of (23) converges to zero. On the other hand, the right-hand side of (23) is bounded below by a positive constant, which yields a contradiction. Therefore, it cannot be that $\delta_k \to 0$. □

We will show that under suitable conditions, and particularly under the false assumption that $\|g(x_k)\|$ is bounded away from zero, $\sum_{k=1}^{\infty}\delta_k$ is finite. It will then follow that $\delta_k \to 0$ and that $\{x_k\}$ is Cauchy and thus converges. In other words, we will show that Algorithm 5.2 has the property P1.

To prove that the sum of the entire sequence δ_k is finite, we proceed in stages, breaking the sequence into more manageable subsequences. Let $\bar{\alpha}$ be a constant such that $0 < \bar{\alpha}\nu < 1$. Define the set

$$\mathcal{S}(\bar{\alpha}) = \{k \mid \alpha_k \geq \bar{\alpha}\}. \tag{24}$$

The set $\mathcal{S}(\bar{\alpha})$ can be thought of as the set of indices for which α_k is not "too small." Let the set $\mathcal{T}$ be defined by the rule

$$\mathcal{T} = \{k \mid f(x_k) - f(x_k + s_k) \geq -\eta_2 q_k(s_k) \text{ and } \alpha_k \geq \alpha_{\min}\}. \tag{25}$$

We also define a set to represent the union of $\mathcal{S}(\bar{\alpha})$ and $\mathcal{T}$, specifically

$$\mathcal{U}(\bar{\alpha}) = \mathcal{S}(\bar{\alpha}) \cup \mathcal{T}. \tag{26}$$

The set $\mathcal{U}(\bar{\alpha})$ is important because it contains as a (typically proper) subset all indices at which the trust region radius increases. Those indices at which the

trust region increases because $\alpha_k > 1$ are in $\mathcal{S}(\bar{\alpha})$, and those indices at which the trust-region increases because $\rho \geq \eta_2$ and $\alpha_k \geq \alpha_{\min}$ are in $\mathcal{T}$.

Because $\alpha_k \nu \|N_k s_k\| \leq \delta_{k+1}$, our primary interest is to obtain bounds involving $\|x_{k+1} - x_k\| = \alpha_k \|s_k\|$ indirectly by obtaining bounds involving δ_{k+1}. We first establish bounds δ_{k+1} for all indices $k \in \mathcal{S}(\bar{\alpha})$.

Lemma 4. *Let $\{x_k\}$, $\{s_k\}$, and $\{\alpha_k\}$ be the sequences of iterates, search directions, and step lengths generated by Algorithm 5.2, respectively. Let $\bar{\alpha}$ be any number in the interval $(0, 1/\nu)$, and define $\mathcal{S}(\bar{\alpha})$ by (24). It holds that*

$$f(x_k) - f(x_k + \alpha_k s_k) \geq \frac{\eta_1 \tau \bar{\alpha}^2}{\gamma_3} \|g(x_k)\| \min\left(\frac{\delta_{k+1}}{\nu \|N_k\|}, \gamma_3 \frac{\|g_k\|}{\|B_k\|}\right) \tag{27}$$

for any $k \in \mathcal{S}(\bar{\alpha})$.

Proof. A combination of the inequality $-q_k(s_k) \geq -\mathcal{Q}(s_k)$ and Powell's sufficient decrease requirement (19) yields

$$-q(s_k) \geq \tau \|g_k\| \min(\delta_k / \|N_k\|, \|g_k\| / \|B_k\|).$$

The inequality $\bar{\alpha} < 1/\nu \leq 1$ may be used to obtain an expression for $q_k(\alpha_k s_k)$ in terms of $q_k(s_k)$ for all $k \in \mathcal{S}(\bar{\alpha})$. If $s_k^T B_k s_k \geq 0$, then $-q_k(\alpha_k s_k) = -\alpha_k g_k^T s_k \geq -\bar{\alpha}\alpha_k q_k(s_k)$. Alternatively, if $s_k^T B_k s_k < 0$, then

$$\begin{aligned} -q_k(\alpha_k s_k) &= -\alpha_k g_k s_k^T - \frac{1}{2}\alpha_k^2 s_k^T B_k s_k \\ &\geq -\bar{\alpha}\alpha_k g_k^T s_k - \tfrac{1}{2}\bar{\alpha}\alpha_k s_k^T B_k s_k = -\bar{\alpha}\alpha_k q_k(s_k). \end{aligned}$$

In either case, $\alpha_k s_k$ satisfies the Wolfe condition on the objective (11) and

$$f(x_k) - f(x_k + \alpha_k s_k) \geq \eta_1 \tau \bar{\alpha} \|g_k\| \min\left(\alpha_k \delta_k / \|N_k\|, \alpha_k \|g_k\| / \|B_k\|\right) \tag{28}$$

for all $k \in \mathcal{S}(\bar{\alpha})$.

By the updating rules for Algorithm 5.2, $\delta_{k+1} \leq \max(\alpha_k \nu \delta_k, \gamma_3 \delta_k)$. For $k \in \mathcal{S}(\bar{\alpha})$, both $\alpha_k \nu / \bar{\alpha} \geq 1$ and $\gamma_3 / \bar{\alpha} > 1$, and thus

$$\delta_{k+1} \leq \frac{\gamma_3 \alpha_k}{\bar{\alpha}} \nu \delta_k.$$

Substituting this upper bound for δ_{k+1} into inequality (28) and noting that $\alpha_k \geq \bar{\alpha}$ for $k \in \mathcal{S}$, we obtain the bound (27). □

Lemma 5. *Let $\{x_k\}$, $\{s_k\}$, and $\{\alpha_k\}$ be the sequences of iterates, search directions, and step lengths generated by Algorithm 5.2, respectively. Let $\bar{\alpha}$ be a constant such that $0 < \bar{\alpha} < 1/\nu \leq 1$, and define the set $\mathcal{U}(\bar{\alpha})$ by (26). Then*

$$f(x_k) - f(x_{k+1}) \geq \hat{\eta} \tau \|g_k\| \min\left(\delta_{k+1} / (\nu \|N_k\|), \gamma_3 \|g_k\| / \|B_k\|\right), \tag{29}$$

where

$$\hat{\eta} = \min\left(\frac{\eta_1 \bar{\alpha}^2}{\gamma_3}, \frac{\eta_2 - \eta_1}{\gamma_3}\right),$$

for all $k \in \mathcal{U}(\bar{\alpha})$.

Proof. The step lengths generated by Algorithm 5.2 satisfy the Wolfe conditions (11) and (12) and the condition (17). Condition (17), repeated here for clarity, is that

$$f(x_k + \alpha_k s_k) - f(x_k) - \eta_1 q_k(\alpha_k s_k) \leq f(x_k + s_k) - f(x_k) - \eta_1 q_k(s_k).$$

We combine this condition with the first Wolfe condition (11) to conclude

$$f(x_k) - f(x_k + \alpha_k s_k) \geq f(x_k) - f(x_k + s_k) + \eta_1 q_k(s_k). \tag{30}$$

Consider, now, $k \in \mathcal{T}$. By definition, for indices in $\mathcal{T}$ it holds that $f(x_k) - f(x_k + s_k) \geq -\eta_2 q_k(s_k)$. Thus because of the bound (30), it also holds that

$$f(x_k) - f(x_k + \alpha_k s_k) \geq -(\eta_2 - \eta_1) q_k(s_k).$$

But then, because s_k meets Powell's sufficient decrease condition (19),

$$f(x_k) - f(x_k + \alpha_k s_k) \geq (\eta_2 - \eta_1)\tau \|g_k\| \min\left(\delta_k / \|N_k\|, \|g_k\| / \|B_k\|\right),$$

for all $k \in \mathcal{T}$.

If $k \notin \mathcal{S}(\bar{\alpha})$, then by definition of $\mathcal{S}(\bar{\alpha})$, $\alpha_k < \bar{\alpha} < \gamma_3$. Whenever $k \in \mathcal{T}$ and $\alpha_k < \gamma_3$, the radius $\delta_{k+1} \leq \gamma_3 \delta_k \leq \gamma_3 \nu \delta_k$, and thus

$$f(x_k) - f(x_k + \alpha_k s_k) \geq \frac{\eta_2 - \eta_1}{\gamma_3} \tau \|g_k\| \min\left(\frac{\delta_{k+1}}{\nu \|N_k\|}, \gamma_3 \frac{\|g_k\|}{\|B_k\|}\right), \tag{31}$$

for all $k \in \mathcal{T} \setminus \mathcal{S}(\bar{\alpha})$. We may therefore combine Equations (27) and (31) to obtain the desired bound (29) for $k \in \mathcal{U}(\bar{\alpha}) = \mathcal{S}(\bar{\alpha}) \cup \mathcal{T}$. □

As we noted above, the set $\mathcal{U}(\bar{\alpha})$ contains all the indices for which δ_k increases. We now explore the behavior of iterates in the complement of $\mathcal{U}(\bar{\alpha})$.

Lemma 6. *Let $0 < \bar{\alpha}\nu < 1$, let $\mathcal{U}(\bar{\alpha})$ be defined by (26) and let $\{k \mid \ell < k \leq m\}$ be any unbroken sequence of iteration indices for Algorithm 5.2 with no members in $\mathcal{U}(\bar{\alpha})$. Then*

$$\sum_{k=\ell}^{m} \delta_{k+1} < \frac{1}{1 - \nu\bar{\alpha}} \delta_{\ell+1}. \tag{32}$$

Proof. When $k \notin \mathcal{U}(\bar{\alpha})$, the trust region radius decreases by at least a constant factor. Specifically, $\delta_{k+1} \leq \bar{\alpha}\nu\delta_k$, where $\bar{\alpha}\nu < 1$. Let $\{k \mid \ell < k \leq m\}$ be any unbroken sequence of iterates with no members in $\mathcal{U}(\bar{\alpha})$. Then

$$\sum_{k=\ell}^{m} \delta_{k+1} \leq \sum_{k=\ell}^{m} (\bar{\alpha}\nu)^{(k-\ell)} \delta_{\ell+1} < \sum_{j=0}^{\infty} (\bar{\alpha}\nu)^j \delta_{\ell+1} = \frac{1}{1 - \nu\bar{\alpha}} \delta_{\ell+1},$$

which is the desired result. □

It is significant that ℓ may be in $\mathcal{U}(\bar{\alpha})$ and that the sum in the bound (32) includes δ_{m+1} even if $m + 1 \in \mathcal{U}(\bar{\alpha})$.

Corollary 1. *Let $\{k \mid \ell < k < \infty\}$ be any unbroken sequence of iteration indices for Algorithm 5.2 with no members in $\mathcal{U}(\bar{\alpha})$. Then*

$$\sum_{k=\ell}^{\infty} \delta_{k+1} \leq \frac{1}{1-\nu\bar{\alpha}} \delta_{\ell+1}.$$

Proof. Because $\delta_k > 0$ for all k, it holds that $\sum_{k=\ell}^{\infty} \delta_{k+1} = \sup_{m>\ell} \sum_{k=\ell}^{m} \delta_{k+1}$. Lemma 6 then provides an upper bound on this supremum. □

We now use Powell's criterion (19) to show that Algorithm 5.2 has the property P1.

Lemma 7. *Let $\{x_k\}$ be a sequence of iterates produced by Algorithm 5.2, let $\Omega \subset \mathbb{R}^n$ be a region containing $\{x_k\}$ in its interior, and let $f(x)$ be twice continuously differentiable and bounded below in Ω. Assume further that $\|B_k\|$, $\|N_k\|$ and $\|N_k^{-1}\|$ are all bounded above. Then if $\{\|g_k\|\}$ is bounded away from zero, $\delta_k \to 0$ and $\{x_k\}$ converges.*

Proof. If $\mathcal{U}(\bar{\alpha})$ is finite, then clearly $\sum_{k\in\mathcal{U}(\bar{\alpha})} \delta_{k+1} < \infty$. Suppose, on the other hand, that $\mathcal{U}(\bar{\alpha})$ is infinite. Algorithm 5.2 generates a nonincreasing sequence $\{f(x_k)\}$ of function values that is, by assumption, bounded below. Thus $\sum_{k=0}^{\infty} (f(x_k) - f(x_{k+1}))$ is finite, and $f(x_k) - f(x_{k+1})$ converges to zero. Also by assumption, the sequence $\{\|g_k\|/\|B_k\|\}$ is bounded away from zero, and hence the bound (29) reduces to

$$f(x_k) - f(x_{k+1}) \geq \frac{\hat{\eta}\tau\|g_k\|}{\nu\|N_k\|} \delta_{k+1},$$

for sufficiently large $k \in \mathcal{U}(\bar{\alpha})$.

Therefore, from the false premise that $\{\|g_k\|\}$ is bounded away from zero, it follows that $\sum_{k\in\mathcal{U}(\bar{\alpha})} \delta_{k+1}$ is finite. We now show that, based on the same premise, $\sum_{k=0}^{\infty} \delta_{k+1}$ is finite.

If $\mathcal{U}(\alpha)$ is empty, then Corollary 1 implies that

$$\sum_{k=0}^{\infty} \delta_{k+1} \leq \frac{1}{1-\nu\bar{\alpha}} \delta_1 < \infty.$$

Similarly, if $\mathcal{U}(\alpha)$ is finite and $K = \max(\mathcal{U}(\bar{\alpha}))$ then

$$\sum_{k=0}^{\infty} \delta_{k+1} = \sum_{k=0}^{K} \delta_{k+1} + \sum_{k=K}^{\infty} \delta_{k+1} \leq \sum_{k=0}^{K} \delta_{k+1} + \frac{1}{1-\nu\bar{\alpha}} \delta_{K+1} < \infty.$$

Suppose, then, that $\mathcal{U}(\bar{\alpha})$ is infinite. We denote the members of $\mathcal{U}(\bar{\alpha})$, taken as a subsequence of all iterates, using the notation $\{k_j\}_{j=1}^{\infty}$. The sequence of all iterates may be partitioned into nonoverlapping subsequences as follows. Let $\mathcal{P}_0 = \{k \mid k < k_1\}$ and $\mathcal{P}_j = \{k \mid k_j \leq k < k_{j+1}\}$ for indices $1 \leq j < \infty$.

If $\mathcal{P}_0$ is not empty, then its first index is zero, and none of its indices are in $\mathcal{U}(\alpha)$. Thus, Lemma 6 applies, and we may use the bound (32) to conclude that

$$\sum_{k\in\mathcal{P}_0} \delta_{k+1} < \frac{1}{1-\nu\bar{\alpha}}\delta_1 < \infty.$$

Similarly, for $1 \le j < \infty$, the first index in the subsequence $\mathcal{P}_j$ is k_j, and none of the other indices are in $\mathcal{U}(\bar{\alpha})$. Thus

$$\sum_{j=1}^{\infty}\sum_{k\in\mathcal{P}_j} \delta_{k+1} \le \frac{1}{1-\nu\bar{\alpha}}\sum_{j=1}^{\infty}\delta_{k_j+1} = \frac{1}{1-\nu\bar{\alpha}}\sum_{k\in\mathcal{U}(\bar{\alpha})}\delta_{k+1} < \infty.$$

But then $\sum_{k=0}^{\infty}\delta_{k+1} = \sum_{k\in\mathcal{P}_0}\delta_{k+1} + \sum_{j=1}^{\infty}\sum_{k\in\mathcal{P}_j}\delta_{k+1} < \infty$. It follows immediately that $\delta_k \to 0$. Moreover, because $\|x_{k+1}-x_k\| \le \|N_k^{-1}\|\delta_{k+1}$ and $\|N_k^{-1}\|$ is bounded above, $\sum_{k=0}^{\infty}\|x_{k+1}-x_k\|$ is finite. Thus the sequence $\{x_k\}$ is Cauchy and must converge. □

We have proved the following theorem. For clarity, we explicitly state the conditions we have imposed on the sequences $\{\alpha_k\}$ and $\{s_k\}$.

Theorem 1. *Let $\{x_k\}$ be a sequence of iterates produced by Algorithm 5.2, let $\Omega \subset \mathbb{R}^n$ be a region containing $\{x_k\}$ in its interior, and let $f(x)$ be twice continuously differentiable in Ω. Suppose that at each iteration the approximate solution, s_k, to the trust-region subproblem (10) satisfies Powell's sufficient decrease criterion (19) and the conditions of Lemma 2. At each iteration, let the line search be performed so that α_k satisfies Condition (17). Assume further that $\|B_k\|$, $\|N_k\|$ and $\|N_k^{-1}\|$ are all bounded above. If f is bounded below in Ω, then* $\liminf_{k\to\infty}\|g_k\| = 0$.

Proof. Lemmas 3 and 7 are clearly contradictory, and so $\|g_k\|$ cannot be bounded away from zero. □

7. Strengthening the Convergence Results

One may show, under additional assumptions, that $\lim_{k\to\infty}\|g(x_k)\| = 0$. The first such proof is due to Thomas [17], who showed that $\lim_{k\to\infty}\|g(x_k)\| = 0$ for classical trust-region methods, in other words for those methods for which $x_{k+1} = x_k$ whenever $\rho_k \le \eta_1$. Our proof borrows from Thomas but has been extensively modified to allow for a line search.

Lemma 8. *Assume that the sequence $\{x_k\}$ generated by Algorithm 5.2 lies in a compact region and that $\{\|N_k\|\}$ is bounded above. Then the sequence $\{\delta_k\}$ is also bounded above.*

Proof. Since $\{x_k\}$ lies in a compact and thus bounded region, the sequence $\{\|x_{k+1}-x_k\|\}$ has a finite upper bound. But $x_{k+1}-x_k = \alpha_k s_k$, and by assumption $\{\|N_k\|\}$ is bounded above. Therefore, the sequence $\{\alpha_k\|N_k s_k\|\}$ is

also bounded above. For each iteration, there are two possible rules for choosing δ_{k+1}. The first rule, applied when $\rho_k < \eta_2$ or $\alpha_k < \alpha_{\min}$, is to choose

$$\delta_{k+1} \in [\alpha_k \|N_k s_k\|, \alpha_k \nu \|N_k s_k\|].$$

If, on the other hand, $\rho_k \geq \eta_2$ and $\alpha_k \geq \alpha_{\min}$, we choose δ_{k+1} by the rule

$$\delta_{k+1} = \max\left(\delta_k, \alpha_k \hat{\nu}_k \|N_k s_k\|, \gamma_3 \|N_k s_k\|\right),$$

where, by definition, the sequence of scalars $\{\hat{\nu}_k\}$ is bounded above by ν and below by one.

Let Γ be any number greater than $\alpha_k \nu \|N_k s_k\|$ for all k. The existence of an upper bound on $\{\alpha_k \|N_k s_k\|\}$ implies that a suitable Γ exists. Assume further, without loss of generality, that Γ is greater than $\gamma_3 \|N_k s_k\|$ for those k for which $\alpha_k \geq \alpha_{\min}$. Then a straightforward inductive argument shows that $\delta_k \leq \max\{\delta_0, \Gamma\}$ for all k. □

The bound on $\{\delta_k\}$ may be used to obtain a strengthened form of Lemma 6.

Lemma 9. *Let $\{k \mid \ell \leq k \leq m\}$ be any unbroken sequence of iterates with no members in $\mathcal{U}(\bar{\alpha})$. Let Δ be an upper bound on $\{\delta_k\}$, and let ξ be a constant scalar such that $\xi \geq \|N_k^{-1}\|$ for all k. Then*

$$\sum_{k=\ell}^{m} \alpha_k \|s_k\| < \frac{\bar{\alpha}\xi}{1 - \nu\bar{\alpha}} \Delta. \tag{33}$$

Proof. Lemma 6 implies that $\sum_{k=\ell}^{m} \delta_{k+1} < \delta_{\ell+1}/(1 - \nu\bar{\alpha})$. But, because ℓ is not in $\mathcal{U}(\bar{\alpha})$, it holds that $\delta_{\ell+1} \leq \bar{\alpha}\delta_\ell$. Furthermore $\delta_\ell \leq \Delta$ by definition of Δ, and so

$$\sum_{k=\ell}^{m} \delta_{k+1} < \frac{\bar{\alpha}}{1 - \nu\bar{\alpha}} \Delta.$$

But $\alpha_k \|s_k\| \leq \alpha_k \xi \|N_k s_k\| \leq \xi \delta_{k+1}$ for any index k. The bound (33) immediately follows. □

We observe that, in contrast to Lemma 6, the conditions of Lemma 9 require that ℓ not be in $\mathcal{U}(\bar{\alpha})$.

Lemma 10. *Assume that the sequence $\{x_k\}$ generated by Algorithm 5.2 lies in a compact region and that the sequences $\{\|B_k\|\}$, $\{\|N_k\|\}$, and $\{\|N_k^{-1}\|\}$ remain bounded. Choose $\epsilon_1 > \epsilon_2 > 0$. For every $\epsilon_3 > 0$ there is an integer L sufficiently large that $\sum_{k=\ell}^{m-1} \alpha_k \|s_k\| < \epsilon_3$ for all indices $m > \ell \geq L$ for which $\|g(x_\ell)\| > \epsilon_1$, $\|g(x_k)\| \geq \epsilon_2$ for consecutive indices $k = \ell, \ell+1, \ldots, m-1$, and $\|g(x_m)\| < \epsilon_2$.*

Proof. Note that if $\|g(x_k)\| > \epsilon_1$ holds only finitely often, then the lemma is trivially true. Furthermore, because $\liminf_{k\to\infty} \|g(x_k)\| = 0$, it must hold that for every iteration ℓ for which $\|x_\ell\| > \epsilon_1$, there must exist a subsequent iteration m such that $\|g(x_m)\| < \epsilon_2$.

Let Δ be an upper bound on $\{\delta_k\}$ and ξ be an upper bound on $\{\|N_k^{-1}\|\}$. Let ϵ_3 be given. We choose the scalar $\bar{\alpha}$ so small that

$$\frac{\bar{\alpha}\xi}{1-\nu\bar{\alpha}}\Delta \leq \frac{\epsilon_3}{2}, \tag{34}$$

with the foresight that this bound will be used in the sequel to obtain a bound on (38).

Let $\mathcal{J}$ denote the sequence of iteration indices $\{\ell, \ell+1, \ldots, m-1\}$. Let $\{k_j\}_{j=1}^r$ denote the subsequence of $\mathcal{J}$ with indices in $\mathcal{U}(\bar{\alpha})$. We partition the set $\mathcal{J}$ into r nonoverlapping subsequences as follows. If $\mathcal{U}(\bar{\alpha}) \cap \mathcal{J}$ is empty, then let $\mathcal{P}_0 = \mathcal{J}$. Otherwise let, $\mathcal{P}_0 = \{\ell, \ell+1, \ldots, k_1-1\}$, $\mathcal{P}_j = \{k_j, k_j+1, \ldots, k_{j+1}-1\}$, $j = 1, 2, \ldots, r-1$ and $\mathcal{P}_r = \{k_r, k_r+1, \ldots, m-1\}$. These definitions allow the quantity to be bounded to be written as

$$\|x_p - x_q\| \leq \sum_{k=\ell}^{m-1} \alpha_k \|s_k\| = \sum_{k\in P_0} \alpha_k \|s_k\| + \sum_{j=1}^{r} \sum_{k\in\mathcal{P}_j} \alpha_k \|s_k\|. \tag{35}$$

First we estimate the quantity $\sum_{k=\mathcal{P}_0} \alpha_k \|s_k\|$. If $p \in \mathcal{U}(\bar{\alpha})$, then $\mathcal{P}_0$ is empty, and the sum is zero. Suppose, then, that $\mathcal{P}_0$ is nonempty. Because no indices in $\mathcal{P}_0$ are in $\mathcal{U}(\bar{\alpha})$, the conditions of Lemma 9 are met, and thus

$$\sum_{k=\mathcal{P}_0} \alpha_k \|s_k\| < \frac{\bar{\alpha}\xi}{1-\nu\bar{\alpha}}\Delta. \tag{36}$$

Clearly, the bound (36) also holds if $\mathcal{P}_0$ is empty.

We now consider the sum

$$\sum_{j=1}^{r} \sum_{k\in\mathcal{P}_j} \alpha_k \|s_k\| \leq \xi \sum_{j=1}^{r} \sum_{k\in\mathcal{P}_j} \delta_{k+1}.$$

For $j = 1, \ldots, r$, the conditions of Lemma 6 are met, and thus

$$\sum_{j=1}^{r} \sum_{k\in\mathcal{P}_j} \alpha_k \|s_k\| < \frac{\xi}{1-\nu\bar{\alpha}} \sum_{j=1}^{r} \delta_{k_j+1} \tag{37}$$

The bounds (36) and (37) may be combined to conclude

$$\sum_{k=0}^{q-1} \alpha_k \|s_k\| \leq \frac{\xi}{1-\nu\bar{\alpha}} \sum_{j=1}^{r} \delta_{k_j+1} + \frac{\bar{\alpha}\xi}{1-\nu\bar{\alpha}}\Delta. \tag{38}$$

Lemma 4 implies that for every index in the set $\mathcal{U}(\bar{\alpha})$,

$$f(x_k) - f(x_{k+1}) \geq \hat{\eta}\tau\|g_k\| \min\left(\delta_{k+1}/(\nu\|N_k\|), \gamma_3\|g_k\|/\|B_k\|\right),$$

where $\hat{\eta}$ is a positive scalar constant. Let $\mathcal{G}(L) = \{k \mid k \geq L \text{ and } \|g(x_k)\| > \epsilon_2\}$. Because $\sum_{k=1}^{\infty} f(x_k) - f(x_{k+1}) < \infty$, because the sequences $\{\|B_k\|\}$ and $\{\|N_k\|\}$

are bounded by assumption, and because $\|g(x_k)\|$ is bounded away from zero for $k \in \mathcal{G}(L)$, it follows that

$$\sum_{k \in \mathcal{U}(\bar{\alpha}) \cap \mathcal{G}(L)} \delta_{k+1} < \infty.$$

Thus, there is a L so large that

$$\sum_{k \in \mathcal{U}(\bar{\alpha}) \cap \mathcal{G}(L)} \delta_{k+1} < \frac{\epsilon_3}{2}. \tag{39}$$

We use the bounds (34) and (39) to bound the right-hand side of (38). We conclude that if $\ell \geq L$, then $\sum_{k=\ell}^{m-1} \alpha_k \|s_k\| < \epsilon_3$. □

Theorem 2. *Let all the conditions of Theorem 1 hold. Assume further that all iterates lie in a compact region. Then* $\lim_{k\to\infty} \|g(x_k)\| = 0$.

Proof. If $\{\|g(x_k)\|\}$ does not converge to zero, then there is an $\epsilon_1 > 0$ such that $\|g(x_k)\| > \epsilon_1$ infinitely often. But, by Theorem 1, $\|g(x_k)\| < \epsilon_1/2$ infinitely often.

Because $g(x)$ is uniformly continuous within a compact set, there is an ϵ_3 sufficiently small that $\|g(x_p) - g(x_q)\| < \epsilon_1/2$ whenever $\|x_p - x_q\| < \epsilon_3$. Taking $\epsilon_2 = \epsilon_1/2$ and applying Lemma 10, we find that there is an integer L sufficiently large that $\sum_{k=\ell}^{m-1} \alpha_k \|s_k\| < \epsilon_3$ for all indices $m > \ell \geq L$ for which $\|g(x_\ell)\| > \epsilon_1$, $\|g(x_k)\| \geq \epsilon_1/2$ for consecutive indices $k = \ell, \ell+1, \ldots, m-1$, and $\|g(x_m)\| < \epsilon_1/2$. But then for any such indices ℓ and m

$$\|x_\ell - x_m\| \leq \sum_{k=\ell}^{m-1} \alpha_k \|s_k\| < \epsilon_3,$$

and so $\|g(x_\ell) - g(x_m)\| < \epsilon_1/2$. This is a contradiction, because $\|g(x_\ell)\| > \epsilon_1$ and $\|g(x_m)\| < \epsilon_1/2$. Thus, there can be no $\epsilon_1 > 0$ such that $\|g(x_k)\| > \epsilon_1$ infinitely often, and therefore $\lim_{k\to\infty} \|g(x_k)\| = 0$. □

8. Numerical Results

We conducted a series of numerical tests on all unconstrained problems in the CUTE [1] test suite with fewer than 100 variables. All algorithms use the Wolfe conditions and the BFGS update to produce a sequence of positive definite approximate Hessians. One of the algorithms uses only a line search on the direction $s_k = -B_k^{-1} g_k$. The other two algorithms are based on Algorithms 5.1 and 5.2 and employ an infinity norm trust region. The iteration is considered to have converged if $\|g(x_k)\|_2 < 10^{-6}(1 + \|g(x_0)\|_2)$. The Euclidean norm is used in the convergence test to make these results comparable with the results from second-order methods. If the exact Hessian at the computed solution is not positive semidefinite, the solution is rejected.

If the solution is not found after 300 iterations, the algorithm is considered to have failed. Likewise, the algorithm will stop if the line search fails to discover a step satisfying the Wolfe conditions. Moreover, we also consider the algorithm to have failed if $\|g(x_k + \alpha_k s_k)\|_2 > 10^{-6}$ and $\alpha\|s_k\|/\|x_k\| < \epsilon$, where $\epsilon \approx 2 \times 10^{-16}$ is machine precision. We have found that when a step is that small, it is extraordinarily unlikely that the iteration will eventually succeed.

The scaling matrix is taken to be $N_k = I$, and the trust-region subproblem is solved by using LSSOL [8]. In all cases we use $\eta_1 = .05$, $\eta_2 = .25$, and $\omega = .9$. For Algorithm 5.2, we set $\alpha_{\min} = 10^{-6}$. For each iteration δ_{k+1} is taken to be some appropriate multiple of $\|s_k\|$. For instance, when the algorithm calls for choosing $\delta_{k+1} \in [\alpha_k\|Ns\|, \nu\alpha_k\|Ns\|]$, we choose $\delta_{k+1} = \alpha_k\|s\|$. For all problems, we use $\delta_0 = 1$ for our initial trust-region radius.

Tables 1 and 2 show the number of function evaluations required by each algorithm to solve each problem. In Table 3 we list the total number of problems solved by each algorithm. Thirty-three problems are solved by all three algorithms. Table 3 also lists the total number of function evaluations required to solve these thirty-three problems. Both the basic Wolfe condition based algorithm and the biased algorithm perform better than the basic line-search algorithm, with the biased algorithm clearly having the best overall performance.

9. Conclusion

We have developed the convergence theory of a method that performs a line search on the direction proposed by the solution of a trust-region subproblem. This method, combined with a BFGS update and a Wolfe condition–based line search, produces an effective quasi-Newton method.

When the Hessian of the function is available, this method has the same second-order convergence properties as traditional trust-region methods. Furthermore, under suitable conditions, convergence will be Q-quadratic. Proofs that the method has these properties are given in [6].

Acknowledgments. This research was performed in partial satisfaction of the requirements for the degree Doctor of Philosophy in Mathematics at the University of California, San Diego.

Final preparation of this paper was supported in part by the Mathematical, Information, and Computational Sciences Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38; and by the National Science Foundation, under grant CDA-9726385.

We are grateful to Philip Gill, Joshua Griffin and the anonymous referees for their careful reading of the manuscript and a number of helpful suggestions.

References

1. I. Bongartz, A. R. Conn, N. I. M. Gould, and Ph. L. Toint. CUTE: Constrained and unconstrained testing environment. Report 93/10, Département de Mathématique, Facultés Universitaires de Namur, 1993.
2. Andrew R. Conn, Nicholas I. M. Gould, and Philippe L. Toint. *Trust-Region Methods.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2000.

Table 1. Test problems A–H with tolerance 10^{-6}

Problem	Wolfe-LS	Wolfe-TR	Biased-TR
AKIVA	^{a}—	17	15
ALLINITU	34	27	24
BARD	43	27	22
BEALE	175	20	17
BIGGS6	^{b}—	^{b}—	^{b}—
BOX3	25	29	21
BRKMCC	23	8	8
BROWNBS	217	25	20
BROWNDEN	—	35	31
CLIFF	15	7	7
CUBE	—	53	44
DECONVU	—	^{b}—	—
DENSCHNA	14	2	2
DENSCHNB	17	11	14
DENSCHNC	225	21	16
DENSCHND	^{a}—	54	^{b}—
DENSCHNE	—	50	48
DENSCHNF	204	2	2
DJTL	^{a}—	^{a}—	^{a}—
ENGVAL2	—	33	30
EXPFIT	^{a}—	18	26
GROWTHLS	^{a}—	^{b}—	159
GULF	—	71	56
HAIRY	183	114	33
HATFLDD	62	50	13
HATFLDE	^{a}—	43	45
HEART6LS	—	—	—
HEART8LS	—	—	249
HELIX	178	31	33
HIMMELBB	^{b}—	2	2
HIMMELBF	—	57	28
HIMMELBG	22	19	19
HIMMELBH	15	2	2
HUMPS	^{a}—	252	^{a}—
HYDC20LS	—	—	—

— : Iteration limit reached
^{a}—: Line search failed
^{b}—: Stationary point not a minimizer.

3. R. Fletcher. An algorithm for solving linearly constrained optimization problems. *Mathematical Programming*, 2:133–165, 1972.
4. R. Fletcher. *Practical Methods of Optimization*. John Wiley and Sons, 1987.
5. A. Friedlander, J. M. Martínez, and S. A. Santos. A new trust region algorithm for bound constrained minimization. *Appl. Math. Optim.*, 30:235–266, 1994.
6. E. Michael Gertz. *Combination trust-region line-search methods for unconstrained optimization*. PhD thesis, University of California, San Diego, 1999.
7. P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders. Methods for modifying matrix factorizations. *Math. Comp.*, 28:505–535, 1974.
8. P. E. Gill, S. J. Hammarling, W. Murray, M. A. Saunders, and M. H. Wright. User's guide for LSSOL (Version 1.0): A Fortran package for constrained linear least-squares and convex quadratic programming. Report SOL 86-1, Department of Operations Research, Stanford University, Stanford, CA, 1986.
9. M. El Hallabi. Globally convergent multi-level inexact hybrid algorithm for equality constrained optimization. Technical Report RT11-98 (revised), Département Informatique et Optimisation, Institut National des Postes et Télécommunications, Rabat, Morocco, 1999.

Table 2. Test problems J–Z with tolerance 10^{-6}

Problem	Wolfe-LS	Wolfe-TR	Biased-TR
JENSMP	[a]—	34	31
KOWOSB	37	42	34
LOGHAIRY	[a]—	[a]—	[a]—
MARATOSB	[b]—	[b]—	[b]—
MEXHAT	92	21	15
MEYER3	[a]—	[a]—	[a]—
OSBORNEA	[a]—	83	73
OSBORNEB	[a]—	87	80
PALMER1C	89	46	37
PALMER1D	89	42	41
PALMER2C	83	49	23
PALMER3C	83	41	38
PALMER4C	109	36	30
PALMER5C	21	29	27
PALMER6C	79	35	28
PALMER7C	73	43	32
PALMER8C	81	61	35
PFIT1LS	[a]—	—	—
PFIT2LS	[a]—	—	—
PFIT3LS	[a]—	—	—
PFIT4LS	[a]—	—	—
ROSENBR	—	42	40
S308	126	15	12
SINEVAL	[a]—	111	94
SISSER	170	15	15
SNAIL	22	137	139
TOINTGOR	117	90	104
TOINTPSP	[a]—	78	77
TOINTQOR	49	93	91
VIBRBEAM	[a]—	81	92
YFITU	—	107	97
ZANGWIL2	3	2	2

— : Iteration limit reached.
[a]—: Line search failed.
[b]—: Stationary point not a minimizer.

Table 3. Summary of quasi-Newton results

	Wolfe-LS	Wolfe-TR	Biased-TR
Problems solved with	33	53	53
Sum of F-evals	2775	1192	956

10. Jorge J. Moré and D. C. Sorensen. Computing a trust region step. *SIAM J. Sci. Stat. Comput.*, 4(3):553–572, September 1983.
11. Jorge Nocedal and Ya Xiang Yuan. Combining trust-region and line-search techniques. Technical Report OTC 98/04, Optimization Technology Center, March 1998.
12. M. J. D. Powell. A new algorithm for unconstrained optimization. Technical Report T.P. 382, A.E.R.E. Harwell, 1970.
13. M. J. D. Powell. A new algorithm for unconstrained optimization. In J. B. Rosen, O. L. Mangasarian, and K. Ritter, editors, *Nonlinear Programming*. Academic Press, 1970.
14. M. J. D. Powell. Convergence properties of a class of minimization algorithms. In O. L. Mangasarian, R. R. Meyer, and S. M. Robinson, editors, *Nonlinear Programming 2*, pages 1–27. Academic Press, 1975.

15. Trond Steihaug. The conjugate gradient method and trust regions in large scale optimization. *SIAM J. Numer. Anal.*, 20(3):626–637, June 1983.
16. J. Stoer. On the numerical solution of constrained least squares problems. *SIAM J. Numer. Anal.*, 8:382–411, 1971.
17. S. W. Thomas. *Sequential estimation techniques for quasi-Newton algorithms.* PhD thesis, Cornell University, 1975.
18. Ph. L. Toint. Towards an efficient sparsity exploiting Newton method for minimization. In I. S. Duff, editor, *Sparse Matrices and Their Uses*, pages 57–87. Academic Press, New York, 1982.
19. P. B. Wilson. *A Simplicial Algorithm for Concave Programming.* PhD thesis, Harvard University, 1963.