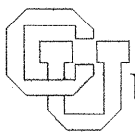
Approximate Solution of The Trust Region Problem by Minimization Over Two-Dimensional Subspaces *

Richard H. Byrd
Robert B. Schnabel
Gerald A. Schultz

CU-CS-346-86



University of Colorado at Boulder
DEPARTMENT OF COMPUTER SCIENCE

* Research supported by ARO contract DAAG 29-84-K-0140, NSF grant DCR-8403483, and NSF cooperative agreement DCR-8420944.

Abstract

The trust region problem, minimization of a quadratic function subject to a spherical trust region constraint, occurs in many optimization algorithms. In a previous paper, the authors introduced an inexpensive approximate solution technique for this problem that involves the solution of a two-dimensional trust region problem. They showed that using this approximation in an unconstrained optimization algorithm leads to the same theoretical global and local convergence properties as are obtained using the exact solution to the trust region problem. This paper reports computational results showing that the two-dimensional minimization approach gives nearly optimal reductions in the n-dimension quadratic model over a wide range of test cases. We also show that there is very little difference, in efficiency and reliability, between using the approximate or exact trust region step in solving standard test problems for unconstrained optimization. These results may encourage the application of similar approximate trust region techniques in other contexts.

1. Introduction.

In this paper we consider the problem

$$\text{minimize } \{g^T d + \tfrac{1}{2} d^T B d : \| d \| \leq \Delta\}, \tag{1.1}$$

where $g \in R^n$, $B \in R^{n \times n}$ is symmetric, and $\Delta > 0$. Problems of this type typically arise in trust region algorithms for unconstrained optimization. We report theoretical and computational results comparing approximate and exact solution techniques for (1.1). Our results show that an inexpensive approximate solution technique of Shultz, Schnabel, and Byrd [1985] appears to perform almost as well as the more expensive exact method in practice. These results appear to have interesting ramifications for the solution of trust region problems in several contexts.

In the context of unconstrained optimization, the quadratic function being minimized in (1.1) is an approximation to the Taylor series of the objective function at the current iterate, where g is the gradient of the objective function at the current iterate and B is some approximation to the Hessian at the current iterate. In view of this, we will refer to a function $M(d) = g^T d + \tfrac{1}{2} d^T B d$ as a quadratic model and speak of reducing the value of the quadratic model. Instead of just minimizing this approximation to the Taylor series, a trust region algorithm constrains the length of the step to be less than some adjustable parameter Δ, recognizing that the approximation is only accurate in some neighborhood of the current iterate. Then, the solution or approximate solution to (1.1) is used as the trial step to move to the next iterate. Thus, in a trust region algorithm, the main source of computational effort, apart from the function evaluations required, is the work on a problem of the form of (1.1) to determine the step from the current iterate.

Trust region algorithms differ in their strategies for approximately solving (1.1). An early trust region algorithm is the single dogleg algorithm of Powell [1970]. This algorithm takes as

its approximate solution to (1.1) the step s, with $\|s\| \leq \Delta$, on the piecewise linear curve passing through the origin, the Cauchy point $d = -\frac{\|g\|^2}{g^T B g} g$, which gives the lowest value of the quadratic model $g^T s + \frac{1}{2} s^T B s$ in the steepest descent direction, and the Newton point $d = -B^{-1} g$, which gives the lowest value of the quadratic model overall. Dennis and Mei [1979] suggest a similar strategy, but with a modified double dogleg curve that is biased toward the Newton point. Trust region algorithms of the dogleg type have the disadvantage that they are not intended to deal with the case that B is not positive definite. Note that they constrain d to the two-dimensional subspace spanned by the Newton and steepest descent directions.

The exact solution to (1.1) satisfies $(B+\alpha I)s = -g$ (see Theorem 1). Hebden [1973] and Moré [1977] suggest approximately solving (1.1) by iterating on α to obtain $\alpha > 0$ such that $\|(B+\alpha I)^{-1} g\|$ is approximately equal to Δ, and then taking the step $-(B+\alpha I)^{-1} g$. This approach requires a new factorization of $B+\alpha I$ for each new value of α, and thus may require increased algebraic effort in comparison to the dogleg algorithms.

More recently, Gay [1981], Sorensen [1982], and Moré and Sorensen [1983] have suggested methods that produce steps that attain at least a fixed fraction $\tau < 1$ of the minimal value for (1.1). We call such a method " τ-optimal." For most problems, these τ-optimal methods compute the approximate solution to (1.1) in the same way as the Moré-Hebden methods. In the so-called "hard case," when there is no α such that $B+\alpha I$ is positive semi-definite and $\|(B+\alpha I)^{-1} g\| \geq \Delta$, a τ-optimal method requires a direction of negative curvature of B to compute its approximate solution to (1.1). These τ-optimal methods have strong theoretical properties, while the computational work required is comparable to that required for the Moré-Hebden approach.

Shultz, Schnabel, and Byrd [1985] present an indefinite dogleg algorithm that achieves the strong theoretical properties of a τ-optimal algorithm while retaining the computational efficiency of a dogleg algorithm. The indefinite dogleg algorithm computes an approximate solution to (1.1) by performing a two-dimensional quadratic minimization,

$$\text{minimize } \{g^T d + \tfrac{1}{2} d^T B d : \| d \| \leq \Delta,\ d \in [u,v]\},$$

where u and v are $-B^{-1}g$ and $-g$ if B is positive definite and are chosen from among $-g$, $-(B+\alpha I)^{-1}g$, and a negative curvature direction when B is indefinite. Note that this approach, by exact minimization in a two-dimensional subspace, will always produce a step that reduces the quadratic model by at least as much as other dogleg type algorithms when B is positive definite. The additional cost of the two-dimensional minimization over a dogleg approach is simply the work required to exactly solve (1.1) where $n = 2$, and is thus negligible.

The main purpose of this paper is to report some perhaps surprising computational evidence that minimization over a subspace spanned by two reasonably chosen directions tends to produce a high percentage of the value given by minimizing exactly over all of R^n.

In Section 2 we briefly compare the theory of τ-optimal algorithms and algorithms of the type considered by Shultz, Schnabel, and Byrd [1985]. In Section 3 we briefly describe the algorithm tested. Section 4 describes our tests and and reports on their results. Finally, in Section 5 we comment on implications of these results.

2. Theoretical Comparison of Exact and Approximate Trust Region Algorithms.

This section discusses the percentage of the optimal value of (1.1) that is achieved by any approximate method in a class proposed by Shultz, Schnabel, and Byrd [1985].

First, we give some definitions.

$||\ ||$ is the Euclidean norm on R^n.

For a symmetric $B \in R^{n \times n}$, let $\lambda_1(B) \in R^n$ be the smallest eigenvalue of B, and let $v_1(B) \in R^n$ be an eigenvector of B corresponding to $\lambda_1(B)$. For notational convenience, we will sometimes suppress the dependence on B.

For any $A \in R^{n \times n}$, let A^+ denote the generalized inverse of A.

For any $u_1, u_2, \ldots, u_m \in R^n$, let $[\ u_1, u_2, \ldots, u_m\ ]$ denote the subspace of R^n spanned by $u_1, u_2, \ldots, u_m$.

A function $s : R^n \times R^{n \times n} \times (0,\infty) \rightarrow R^n$ is called a step computing function, typically denoted by $s(g,B,\Delta)$.

For any $g \in R^n$, symmetric $B \in R^{n \times n}$, and $\Delta > 0$, let $s_*(g,B,\Delta)$ be a solution to (1.1). Such an s_* is referred to as an optimal step computing function.

For $s \in R^n$, let $pred(s,g,B) = -s^T g - \frac{1}{2} s^T B s$.

For $\tau \in (0,1]$, a step computing function s is τ-optimal if for any $g \in R^n$, symmetric $B \in R^{n \times n}$, and $\Delta > 0$,

$$pred(s(g,B,\Delta),g,B) \geq \tau\ pred(s_*(g,B,\Delta),g,B)\ .$$

We will now state a theorem characterizing the solution to (1.1). See, for example, Sorensen [1982] for a proof of this result. This result will provide the theoretical basis for our step computing function as well as for τ-optimal step computing functions.

Theorem 1.

Consider any symmetric $B \in R^{n \times n}$, $g \in R^n$, and $\Delta > 0$. Let $s_* \in R^n$ be a solution to (1.1). If B is positive definite and $||B^{-1}g|| \leq \Delta$, then $s_* = -B^{-1}g$.

If B is positive definite and $\|B^{-1}g\| > \Delta$, then for the unique $\alpha > 0$ such that $\|(B+\alpha I)^{-1}g\| = \Delta$, $s_* = -(B+\alpha I)^{-1}g$.

If $\lambda_1 \leq 0$ and there is an $\alpha \geq -\lambda_1$ such that $\|(B+\alpha I)^{-1}g\| = \Delta$, then $s_* = -(B+\alpha I)^{-1}g$.

Otherwise, $s_* = -(B-\lambda_1 I)^{+}g + \xi v_1$, where $\xi \epsilon R$ is such that $\|-(B-\lambda_1 I)^{+}g + \xi v_1\| = \Delta$.

It is clear from Theorem 1 that any method that attempts to closely approximate the solution to (1.1) must approximate $\lambda_1(B)$ reasonably well in the case that B is not positive definite, and must also produce some direction of negative curvature in the final case in Theorem 1, which Moré and Sorensen designate the "hard case." This case can only arise if g is orthogonal to every eigenvector of B corresponding to the eigenvalue λ_1.

We now discuss the conditions on a step computing function presented in Shultz, Schnabel, and Byrd [1985]. They show that a trust region algorithm for minimization that uses a step computing function satisfying these conditions has the same global and local convergence properties as an algorithm using an optimal or τ-optimal step computing function. But, as we will show shortly, these conditions are slightly weaker than the τ-optimality condition in another sense.

The conditions on a step computing function $s(g,B,\Delta)$ are given below. The first condition, originally due to Powell [1970], roughly says that the step provides at least a fixed fraction of the decrease in the quadratic model that would be obtained from the best permissible step in the steepest descent direction. The second condition roughly says that when B is indefinite, the step provides at least a fixed fraction of the decrease in the quadratic model that would be obtained from the best permissible step in the direction of greatest negative curvature. The third condition simply says that if B is positive definite and the Newton step is permissible, then it is chosen.

Conditions on a Step Computing Function.

Condition #1

There are c_1, $\bar{c}_1 > 0$ such that for all $g \epsilon R^n$, for all symmetric $B \epsilon R^{n \times n}$, and for all $\Delta > 0$,

$$pred(\ s(g,B,\Delta),\ g,\ B\ ) \geq c_1 \|g\| \min\{\Delta, \bar{c}_1 \frac{\|g\|}{\|B\|}\}.$$

Condition #2

There is a $c_2 > 0$ such that for all $g \epsilon R^n$, for all symmetric $B \epsilon R^{n \times n}$, and for all $\Delta > 0$,

$$pred(\ s(g,B,\Delta),\ g,\ B\ ) \geq c_2(-\lambda_1(B))\Delta^2 .$$

Condition #3

If $B \epsilon R^{n \times n}$ is positive definite and $\|-B^{-1}g\| \leq \Delta$, then $s(g,B,\Delta) = -B^{-1}g$.

The step computing function for which we present test results in Section 4 satisfies these three conditions. Example 1 below shows that when B is positive definite, a step satisfying these three conditions may not be τ-optimal, for any $\tau > 0$. The step computing function used in this example is exactly the one used in our computational tests in Section 4, when B is positive definite. The second example given below shows that when B is indefinite, a step computing function satisfying Conditions #1 and #2 also may not be τ-optimal, for any $\tau > 0$. The step computing function used in this example, minimization over the subspace spanned by the gradient and a negative curvature direction, is the simplest function that satisfies Conditions #1 and #2 in the indefinite case.

Example 1 (positive definite case)

Let $\epsilon > 0$, $B = diag(1,\epsilon^2,\epsilon^4)$, $g = (\epsilon^2,\epsilon^2,\epsilon^3)^T$, $\alpha = \epsilon^2$, and $\Delta = \|(B+\alpha I)^{-1}g\|$. Then the solution to (1.1) is

$$s_* = -(B+\alpha I)^{-1}g = (\frac{-\epsilon^2}{1+\epsilon^2}, \frac{-\epsilon^2}{\epsilon^2+\epsilon^2}, \frac{-\epsilon^3}{\epsilon^4+\epsilon^2})^T = (\frac{-\epsilon^2}{1+\epsilon^2}, -\tfrac{1}{2}, \frac{-\epsilon}{1+\epsilon^2})^T ,$$

and $pred(s_*,g,B) = O(\epsilon^2)$. Define a step computing function by taking $s(g,B,\Delta)$ to be the solution to

$$\text{minimize } \{g^T d + \tfrac{1}{2} d^T B d : \|d\| \leq \Delta,\ d \in [\, -g\ ,\ -B^{-1}g\ ]\},$$

which is the step to the minimum of the quadratic model subject to the trust region in the two-dimensional subspace spanned by the gradient direction and the Newton direction. Since this step will do at least as well as the best gradient step, it is easy to see that $s(g,B,\Delta)$ satisfies Condition #1. Condition #2 is vacuously satisfied, since $\lambda_1(B) > 0$. Note that $\Delta = O(1)$, $\|g\| = O(\epsilon^2)$, $g^T B g = O(\epsilon^4)$, $g^T B^{-1} g = O(\epsilon^2)$, and $\|B^{-1}g\| = O(\frac{1}{\epsilon})$.

For given g, B, and Δ, $s(g,B,\Delta) = \mu g + \nu B^{-1} g$ for some $\mu, \nu \in \boldsymbol{R}$. Then

$$pred(\, s(g,B,\Delta),\, g,B) = -\mu g^T g - \nu g^T B^{-1} g - \tfrac{1}{2}\mu^2 g^T B g - \mu\nu g^T B B^{-1} g - \tfrac{1}{2}\nu^2 g^T B^{-1} B B^{-1} g$$

$$\leq (-\mu - \mu\nu) g^T g - \tfrac{1}{2}\mu^2 g^T B g - \nu g^T B^{-1} g\ .$$

Now, since $\dfrac{g^T B^{-1} g}{\|g\|\,\|B^{-1}g\|} = O(\epsilon)$, it follows that for all small enough $\epsilon > 0$, g and $B^{-1}g$ are nearly orthogonal and so $\|\nu B^{-1} g\| \leq O(\Delta) = O(1)$, and $\|\mu g\| \leq O(\Delta) = O(1)$. Thus, $|\nu| \leq O(\epsilon)$, $|\mu| \leq O(\frac{1}{\epsilon^2})$, and

$$pred(s(g,B,\Delta),g,B) \leq -\mu(1+\nu) O(\epsilon^4) - \tfrac{1}{2}\mu^2 O(\epsilon^4) + O(\epsilon^3)$$

$$\leq (-\mu - \tfrac{1}{2}\mu^2) O(\epsilon^4) - \mu O(\epsilon^5) + O(\epsilon^3)$$

$$\leq (-\mu - \tfrac{1}{2}\mu^2) O(\epsilon^4) + O(\epsilon^3)\ .$$

Finally, since $\max_{\mu \in \boldsymbol{R}} (-\mu - \tfrac{1}{2}\mu^2) = \tfrac{1}{2}$ we have that

$$pred(s(g,B,\Delta),g,B) \leq O(\epsilon^3)\ .$$

Thus,

$$\frac{pred(s(g,B,\Delta),g,B)}{pred(s_*,g,B)} = O(\epsilon)\ .$$

Example 2 (indefinite case)

Let $\epsilon > 0$, $B = diag(-\epsilon^2,\epsilon,1)$, $g = (0,\epsilon,\epsilon)^T$, $q = (1,0,0)^T$, $\alpha = 2\epsilon^2$, and $\Delta = \| (B+\alpha I)^{-1} g \|$. Then the solution to (1.1) is

$$s_* = -(B+\alpha I)^{-1} g = (0, \frac{-\epsilon}{\epsilon + 2\epsilon^2}, \frac{-\epsilon}{1 + 2\epsilon^2})^T = (0, \frac{-1}{1 + 2\epsilon}, \frac{-\epsilon}{1 + 2\epsilon^2})^T ,$$

and $pred(s_*,g,B) = O(\epsilon)$. Now, define a step computing function by taking $s(g,B,\Delta)$ to be the solution to

$$\text{minimize } \{ g^T d + \tfrac{1}{2} d^T B d : \| d \| \leq \Delta, d \epsilon [\, g \, , \, q \,] \}.$$

Again, this step satisfies Condition #1 since it does at least as well as the best gradient step, and it can easily be shown to satisfy Condition #2 since q is nearly the direction of greatest negative curvature of B. The best possible reduction for a step in the direction q is $\epsilon^2 \Delta^2 = O(\epsilon^2)$, since $\Delta = O(1)$. Since $\frac{g^T g}{g^T B g} \leq \frac{\Delta}{\| g \|}$, it follows easily that the best reduction in the direction g is given by the Cauchy step and is $\frac{1}{2} \frac{(g^T g)^2}{g^T B g} = O(\epsilon^2)$. Thus, since $g^T q = g^T B q = 0$, it is clear that $pred(s(g,B,\Delta),g,B) = O(\epsilon^2)$, so $\frac{pred(s(g,B,\Delta),g,B)}{pred(s_*,g,B)} = O(\epsilon)$.

In both Examples 1 and 2, the condition number of B approaches infinity. Theorem 2 gives a condition on the matrix B that implies the τ-optimality of a step computing function satisfying Conditions #1 and #2. This condition is, roughly, that if B is positive definite, the condition number of B is bounded, and if B is indefinite, the amount of negative curvature is not negligible.

Theorem 2. Suppose s is a step computing function that satisfies Conditions #1 and #2, with constants c_1, $\bar{c}_1$, $c_2 > 0$. Let $\kappa > 0$, and consider any g, B, and Δ, with $\frac{\| B \|}{|\lambda_1(B)|} \leq \kappa$.

Then

$$\frac{pred(s(g,B,\Delta),g,B)}{pred(s_*,g,B)} \geq \frac{1}{\kappa} \min\{2c_1\bar{c}_1, \frac{2\bar{c}_1 c_2}{2+\bar{c}_1}, \tfrac{1}{2}\kappa c_1, \tfrac{1}{2}\kappa c_2\},$$

where s_* is a solution to (1.1) for g, B, and Δ.

Proof. For notational convenience, let $s = s(g,B,\Delta)$, $pred = pred(s,g,B)$, and $pred_* = pred(s_*,g,B)$. First, note that

$$\min_{\|w\| \leq \Delta} g^T w + \tfrac{1}{2} w^T B w \geq \min_{\|w\| \leq \Delta} g^T w + \min_{\|w\| \leq \Delta} \tfrac{1}{2} w^T B w$$

$$\geq -\|g\|\Delta + \min\{0, \tfrac{1}{2}\lambda_1(B)\Delta^2\} .$$

Thus,

$$pred_* \leq \|g\|\Delta + \min\{0, \tfrac{1}{2}(-\lambda_1(B))\Delta^2\} . \tag{2.1}$$

Now, if $\lambda_1(B) > 0$, then by (2.1)

$$pred_* \leq \|g\|\Delta ,$$

and since the Newton step is the global minimum of $g^T w + \tfrac{1}{2} w^T B w$ in this case,

$$pred_* \leq pred(-B^{-1}g,g,B) = \tfrac{1}{2} g^T B^{-1} g \leq \tfrac{1}{2}\|g\|^2 \|B^{-1}\|$$

$$= \tfrac{1}{2}\|B^{-1}\| \|B\| \frac{\|g\|^2}{\|B\|} \leq \tfrac{1}{2}\frac{\|B\|}{\lambda_1(B)} \frac{\|g\|^2}{\|B\|} \leq \tfrac{1}{2}\kappa \frac{\|g\|^2}{\|B\|} .$$

Thus, by Condition #1,

$$\frac{pred}{pred_*} \geq \frac{c_1 \min\{\|g\|\Delta, \bar{c}_1 \frac{\|g\|^2}{\|B\|}\}}{\min\{\|g\|\Delta, \tfrac{1}{2}\kappa \frac{\|g\|^2}{\|B\|}\}} \geq c_1 \min\{1, \frac{2\bar{c}_1}{\kappa}\}. \tag{2.2}$$

Otherwise, $\lambda_1(B) \leq 0$. Suppose first that $\Delta \geq \bar{c}_1 \frac{\|g\|}{\|B\|}$. Then by (2.1),

$$pred_* \leq \|g\|\Delta + \tfrac{1}{2}\|B\|\Delta^2 \leq \frac{1}{\bar{c}_1}\|B\|\Delta^2 + \tfrac{1}{2}\|B\|\Delta^2 = (\tfrac{1}{2} + \frac{1}{\bar{c}_1})\|B\|\Delta^2 .$$

Thus, by Condition #2,

$$\frac{pred}{pred_*} \geq \frac{c_2|\lambda_1(B)|\Delta^2}{(\tfrac{1}{2}+\frac{1}{\bar{c}_1})\,||B||\Delta^2} \geq \frac{c_2}{\tfrac{1}{2}+\frac{1}{\bar{c}_1}}\,\frac{1}{\kappa} \geq \frac{2\bar{c}_1 c_2}{2+\bar{c}_1}\,\frac{1}{\kappa}. \tag{2.3}$$

Finally, if $\Delta < \bar{c}_1 \frac{||g||}{||B||}$, then by Conditions #1 and #2, $pred \geq c_1 ||g|| \Delta$, and $pred \geq c_2|\lambda_1(B)|\Delta^2$, so by (2.1)

$$\frac{pred}{pred_*} \geq \tfrac{1}{2}\frac{c_1||g||\Delta + c_2|\lambda_1(B)|\Delta^2)}{||g||\Delta + \tfrac{1}{2}|\lambda_1(B)|\Delta^2} \geq \tfrac{1}{2}\min\{c_1, c_2\}. \tag{2.4}$$

The conclusion now follows from (2.2), (2.3), and (2.4). □

3. The Step Computing Function.

We now briefly describe the step computing function used for the tests reported in Section 4. It is described in more detail in Shultz, Schnabel, and Byrd [1985]. They show that it satisfies the conditions on a step computing function given in Section 2. As shown by Example 1, however, it is not a τ-optimal step computing function. We will refer to this function as the indefinite dogleg step computing function.

When B is positive definite, our step computing function chooses d to minimize the quadratic model over the subspace spanned by the steepest descent and Newton directions, subject to the trust region constraint.

In the indefinite case, our step computing function computes an $\alpha \epsilon (-\lambda_1, -2\lambda_1]$, and a negative curvature direction v with $\frac{v^T B v}{v^T v} \epsilon [\lambda_1, \tfrac{1}{2}\lambda_1]$. These α and v are obtained by using the Lanczos method (see, e.g. Parlett [1980]) to approximate an eigenvalue of B, hopefully λ_1, to

within a relative error of 0.1. We then set α equal to a constant (2 in the implementation tested) multiplied by the approximation from the Lanczos method, and use the Cholesky factorization to factor $B+\alpha I$ and determine whether it is positive definite. If the Cholesky factorization fails, then the eigenvalue approximated by the Lanczos method was not λ_1; in this case a better direction of negative curvature is obtained from the Cholesky decomposition and the Lanczos method is restarted using this direction.

The step computing method always starts by attempting to do the Cholesky factorization of B. Thus, if B is positive definite, we immediately obtain the factorization of B needed to compute $-B^{-1}g$. The results in Section 4 show that an average of roughly 1.1 matrix factorizations per iteration are required by this method.

The Indefinite Dogleg Step Computing Function.

Let $\rho\epsilon(0,1)$ be given. Given $g\epsilon R^n$, symmetric $B\epsilon R^{n\times n}$, and $\Delta > 0$, if B is not positive definite, compute $\alpha\epsilon(-\lambda_1, -2\lambda_1]$ and $v\epsilon R^n$ such that $\dfrac{v^T Bv}{v^T v} < -\rho\lambda_1$.

If B is positive definite then $s(g,B,\Delta)$ is the solution to

$$\text{minimize } \{g^T d + \tfrac{1}{2}d^T Bd : \|d\| \leq \Delta,\ d\epsilon[-g, -B^{-1}g]\}. \tag{3.1}$$

If B is not positive definite and $\|(B+\alpha I)^{-1}g\| > \Delta$ then $s(g,B,\Delta)$ is the solution to

$$\text{minimize } \{g^T d + \tfrac{1}{2}d^T Bd : \|d\| \leq \Delta,\ d\epsilon[-g, -(B+\alpha I)^{-1}g]\}. \tag{3.2}$$

If B is not positive definite and $\|(B+\alpha I)^{-1}g\| \leq \Delta$ then $s(g,B,\Delta)$ is

$$-(B+\alpha I)^{-1}g+\xi v, \tag{3.3}$$

where ξ is selected so that $\|s(g,B,\Delta)\| = \Delta$ and $\xi v^T(B+\alpha I)^{-1}g \leq 0$.

The three steps (3.1), (3.2), and (3.3) above are denoted by "P", "I", and "H" for "positive definite," "indefinite," and "hard case," respectively, in Table 1. An alternative step in the hard case, perhaps more in keeping with the two-dimensional minimization approach, would be to choose $s(g,B,\Delta)$ as the solution to

$$\text{minimize } \{g^T d + \tfrac{1}{2} d^T B d : \| d \| \leq \Delta,\ d \in [-(B+\alpha I)^{-1} g,\ v]\} .$$

The step (3.3) was chosen instead because its form is closer to the step taken by the optimal algorithm in the hard case.

When $\lambda_1(B)$ is close to 0, the indefinite dogleg step is sometimes computed by a slight variant of the above. The augmentation α is instead calculated by

$$\alpha_g = \frac{pred_g}{c_2 \Delta^2} , \tag{3.4}$$

where

$$pred_g = \text{minimize } \{g^T d + \tfrac{1}{2} d^T B d : \| d \| \leq \Delta,\ d \in [-g]\} .$$

Then $s(g,B,\Delta)$ is calculated by (3.2). This modification, which is needed in theory when $\lambda_1(B) = 0$ and in practice when $\lambda_1(B)$ is close to 0, is explained in detail in Shultz, Schnabel, and Byrd [1985]. We denote a step that uses it by S, for "(nearly) semi-definite."

4. Test Results.

We now report our test results for the indefinite dogleg step computing function described above. We tested the method both on randomly generated g, B, and Δ, and on g, B, and Δ arising in the context of a trust region algorithm for minimization. Our approach was to compare the decrease in the quadratic model of our step computing function to the optimal decrease. The results detailed below show, perhaps surprisingly, that on the average the

indefinite dogleg step computing function attained a high percentage of the optimal decrease. The tests were performed in double precision arithmetic on a DEC VAX 11/780 in the Department of Computer Science of the University of Colorado at Boulder.

We first present results on the behavior of our indefinite dogleg step computing function on a large number of randomly generated trust region problems.

First we will describe how the trust region problems were generated. Our goal was to generate reasonable problems, and yet to make the problems difficult enough to provide a realistic test of the difficult cases for the indefinite dogleg step computing function. We tested the algorithm on a variety of test sets. Each test set consisted of 25 problems, 5 problems each of dimensions 20, 40, 60, 80, and 100, all generated by the same scheme.

A trust region problem is defined by the symmetric matrix B, a vector g, and the trust radius Δ. For each test set we first generated B and g, then chose optimal steps of one of three forms, and finally set Δ to be the length of the optimal step. In this way we efficiently generated problems with known optimal solutions. For the first through the nineteenth test sets, the optimal step was selected by choosing an augmentation α and then taking the optimal step to be $-(B+\alpha I)^{-1}g$. For the twentieth test set, values of α and ξ were chosen, and the optimal step was taken to be $-(B+\alpha I)^{-1}g+\xi v_1$, where v_1 is a normalized eigenvector for $\lambda_1(B)$. For the last test set, g was taken to be 0, and the optimal step was taken to be v_1.

The basic scheme for generating B and g was to first choose the eigenvalues of B from a uniform distribution in some interval, using the random number generator of Schrage [1979]. Then, as in Moré and Sorensen [1983], the diagonal matrix consisting of these eigenvalues was pre- and post-multiplied by orthogonal matrices, and B was taken to be the resulting matrix. Next the components of g were randomly generated in some interval, and then pre-multiplied

by the same orthogonal matrices.

When an interval is given in the second and third columns in Table 1, it is the interval in which the eigenvalues of B and the unmodified components of g are uniformly distributed. When an interval is listed in the fourth column in Table 1, it is the interval in which the values of the optimal augmentation α are uniformly distributed.

For the test sets 1 through 6 given in Table 1, no further modifications to B or g were made. The optimal step s_* was simply taken to be $-(B+\alpha I)^{-1}g$, with $\Delta = \|(B+\alpha I)^{-1}g\|$.

The test sets 7 through 16 were generated in the same way as the first six test sets, but in addition changes were made to either B or g, in an attempt to make the problems more difficult. In order to generate problems with a more difficult eigenvalue distribution, the smallest eigenvalue from a uniform distribution over (0,2) was sometimes set to 0, and sometimes switched to the opposite sign. These modifications are designated in Table 1 by a "Z" (for "zero") or an "O" (for "opposite"), respectively, following the interval for the eigenvalue range. In order to generate problems for which the best gradient step tends to do poorly, sometimes the components of g corresponding to positive eigenvalues were chosen to be uniformly distributed in the interval (−1,1), while the components corresponding to negative eigenvalues were chosen to be uniformly distributed in the interval (−0.1,0.1). The intent of this was to make g likely to be a direction with large positive curvature, and hence make the best gradient step tend to give a small reduction of the quadratic model. These problems are designated by a "B" (for "biased") following the interval listed in the gradient component range column.

For the test sets 17 through 19, the eigenvalues were chosen to be normally distributed with mean 0. This was done in order to produce test problems with a non-uniform eigenvalue distribution and to make it more likely that $\lambda_1(B)$ is relatively isolated from any other nega-

tive eigenvalues of B. The optimal step was taken to be of the form $-(B+\alpha I)^{-1}g$, as for the first 16 test sets. These problems are designated by an "N" (for "normal") appearing in the eigenvalue range column. In order to make these problems more difficult, the gradient components were chosen in the biased way described above, as designated by a "B" appearing in the gradient component range column.

For the test set 20, the component of the gradient corresponding to the smallest eigenvalue of B was set to 0. Then a random value for ξ was chosen uniformly in the interval (0,1), and the optimal step was taken as $s_* = -(B+\alpha I)^{-1}g + \xi v_1$. Since Δ was chosen as the length of s_* and the eigenvalue range was such that $\lambda_1(B)<0$ always occurred, this test set consists of problems of the type called the "hard case" by Moré and Sorensen [1983]. Thus they are designated by an "H" (for "hard case") appearing in the gradient component range column.

Test set 21 consisted of saddle-point problems. The gradient was set to 0 and the optimal solution chosen to be the eigenvector of length 1 corresponding to the smallest eigenvalue. These problems are designated in Table 1 by an "S" (for "saddle-point") appearing in the gradient component range column.

Now we describe the remaining columns of Table 1. For each test set, the fifth column lists the type of steps taken by the indefinite dogleg step computing function, followed by the number of problems out of the 25 in that set that each step type was taken.

For each test set, the sixth and seventh columns report the average and minimum ratios of the decrease in the quadratic model obtained by the indefinite dogleg step divided by the decrease obtained by the optimal step. The eighth column reports the average ratio of the decrease obtained by the best gradient step to the optimal decrease. This is included in order to show that the gradient direction alone does not do particularly well on these test problems.

Table 1

Fraction of Optimal Reduction Obtained by the Indefinite Dogleg Step Computing Function on Randomly Generated Trust Region Problems

Test Set Num.	Eig. Range	Grad. Range	Aug. Range	Step Type	Ave. Redn.	Min. Redn.	Best Grad. Ave.
1	(0,2)	(-1,1)	(0,.01)	P:25	.96	.60	.37
2	(-.1,1)	(-1,1)	(0,.1)	H:6,I:19	.97	.79	.36
3	(-.1,1)	(-1,1)	(0,1)	H:1,I:24	.98	.95	.85
4	(-.01,1)	(-1,1)	(0,.01)	S:2,P:12, H:2,I:9	.96	.72	.28
5	(-.01,1)	(-1,1)	(0,.1)	S:6,P:12, I:7	.91	.72	.48
6	(-.01,1)	(-1,1)	(0,1)	S:6,P:12, I:7	.97	.86	.87
7	(-1,1)	B	(0,.01)	H:25	.97	.87	.26
8	(-.1,1)	B	(0,.01)	H:24,I:1	.99	.90	.42
9	(-1,1)	B	(0,.1)	H:25	.99	.96	.81
10	(0,2)O	(-1,1)	(0,.01)	S:3,H:14,I:8	.97	.84	.12
11	(0,2)O	B	(0,.01)	S:4,H:13,I:8	.97	.79	.37
12	(0,2)O	B	(0,.1)	S:4,H:1,I:20	.95	.68	.53
13	(0,2)O	B	(0,1)	S:4,I:21	.96	.76	.83
14	(0,2)Z	B	(0,.01)	S:25	.96	.83	.17
15	(0,2)Z	B	(0,.1)	S:25	.98	.87	.37
16	(0,2)Z	B	(0,1)	S:25	.99	.96	.78
17	N	B	(0,.01)	H:25	.98	.83	.08
18	N	B	(0,.1)	H:25	.99	.84	.63
19	N	B	(0,1)	H:15,I:10	.99	.99	.94
20	(-1,1)	(-1,1)H	(0,.01)	H:25	.97	.91	.34
21	(-1,1)	S	-	H:25	.97	.84	0

Notes

1) In eigenvalue range column, "O" means smallest eigenvalue is switched to the opposite sign; "Z" means smallest eigenvalue is set to 0; "N" means eigenvalues are normally distributed.
2) In gradient component range column, "B" means that the gradient components are taken in (−0.1,0.1) if corresponding eigenvalue is negative, taken in (−1,1) otherwise; "H" means that gradient component corresponding to most negative eigenvalue is set to 0; "S" means that $g=0$.
3) In step type column,
"P" means that step was of the form (3.1),
"I" means that step was of the form (3.2),
"H" means that step was of the form (3.3),
"S" means that step was of the form (3.2) with α given by (3.4).

We observe from Table 1 that the indefinite dogleg step computing function obtains a very high fraction of the optimal reduction in the quadratic model. In fact, in all the test sets in Table 1, the lowest average fraction of the optimal obtained is 0.91 in test set 5, and all of the other average fractions are bigger than 0.95. Note also that the Newton step is never the optimal step in these tests, so it is not the case that the fraction is high simply because Newton steps are weighting it toward 1. Indeed, the smallest fraction of the optimal decrease obtained by any indefinite dogleg step in any of these tests is 0.6 in test set 1. These results indicate that the indefinite dogleg step computing function does quite well on solving (1.1), including problems where B is fairly large and non-sparse, even though it only computes the step in a two-dimensional subspace and is not τ-optimal in theory.

We now present test results for a trust region algorithm for unconstrained minimization that uses the indefinite dogleg step computing function to generate its iterates. The code that was tested was a modified version of the code described in Schnabel, Koontz, and Weiss [1985]. The exact gradient and Hessian functions were used.

This algorithm was run on a number of minimization problems in a standard test set in Moré Garbow, and Hillstrom [1981]. Table 4 lists the test function numbers and names. The basic set obtained from these 18 problems consists of three tests for each function, given by starting the iteration at the standard starting point x_0, at $10x_0$, and at $100x_0$. Test problems 4, 5, and 10 are badly scaled and were omitted from our tests because our implementation made no attempt to deal with bad scaling. On some of the remaining test functions, some of the larger starting points led to overflows at the first iterate or first step, and therefore had to be discarded.

Table 2 contains the results for the tests on this standard test set. The first two columns contain the test function numbers and the number of variables. The third column contains the power of 10 by which the standard starting point is multiplied. The fourth and fifth columns contain the number of iterations and function evaluations, respectively, that were required by the minimization algorithm using the optimal step computing function. The sixth and seventh columns contain the number of iterations and function evaluations for the same minimization algorithm using the indefinite dogleg step computing function. The eighth and ninth columns contain the average and minimum ratios, respectively, of the reduction in the quadratic model obtained by the indefinite dogleg step compared to the reduction obtained by the optimal step computing function. These numbers were obtained as follows. For each step taken by the indefinite dogleg algorithm, the optimal step was computed for the same quadratic trust region problem, and the ratio of the reduction in the quadratic model by the indefinite dogleg step to the reduction by the optimal step was computed. Then the average and minimum of these ratios was calculated over all the iterations. Finally, the tenth column contains the total number of Cholesky factorizations attempted by the indefinite dogleg minimization algorithm.

Table 2

Performance of the Indefinite Dogleg Step Computing Function in a Trust Region Algorithm for Unconstrained Minimization on Standard Test Functions with Standard Starting Points

Fcn. Num.	Dim.	x(0)	Opt. Itn.	Opt. Fcn.	Dog. Itn.	Dog. Fcn.	Ave. Redn.	Min. Redn.	Num. Chol.
1	3	0	13	14	10	12	.98	.91	12
		1	14	17	15	19	.99	.98	16
		2	14	18	16	20	.99	.98	16
2	6	0	30	41	47	63	.95	.53	51
3	3	0	2	3	2	3	.99	.99	2
6	10	0	14	15	14	15	1.	1.	14
		1	17	18	17	18	.99	.99	17
		2	23	24	23	24	.99	.99	23
7	9	0	12	13	14	15	.85	.39	14
		1	49	63	42	52	.85	.14	45
		2	52	64	47	57	.92	.35	47
7	12	0	13	14	21	22	.59	.14	21
8	10	0	31	43	31	43	.99	.99	31
		1	36	48	36	48	.99	.99	36
		2	43	57	43	57	.99	.96	43
9	4	0	76	102	75	100	.99	.96	75
		1	85	115	85	115	.99	.96	85
		2	77	102	83	112	.99	.91	83
9	10	0	88	115	92	123	.97	.88	92
		1	93	122	97	129	.97	.87	97
		2	100	131	105	139	.97	.86	105
11	4	0	8	9	8	9	.99	.99	8
		1	14	15	14	15	.99	.99	14
		2	20	21	20	21	.99	.99	20
12	3	0	20	23	25	29	.93	.46	25
13	10	0	9	10	9	12	.99	.98	9
		1	17	23	17	22	.98	.79	18
		2	14	15	14	15	.99	.99	14

14	2	0	20	23	22	27	.99	.99	22
		1	44	56	43	55	.99	.99	43
		2	110	146	110	146	.99	.99	110
15	4	0	15	16	15	16	.99	.94	15
		1	20	21	20	21	.99	.98	20
		2	26	27	26	27	.99	.99	26
16	2	0	9	11	9	11	.99	.99	11
		1	55	75	57	74	.99	.80	60
17	4	0	41	51	40	51	.99	.95	42
		1	43	55	45	59	.99	.97	47
		2	50	67	53	67	.99	95	55
18	7	0	7	9	7	9	.98	.90	9
18	8	0	10	15	12	16	.97	.91	15
18	9	0	9	12	9	12	.97	.81	14
18	10	0	10	14	10	14	.98	.95	13

The results in columns 8 and 9 of Table 2 show that the indefinite dogleg step computing function does a good job of approximately solving (1.1). Note that the very high average fractions in column 8 are considerably influenced by the presence of a large number of iterates at which the Newton step was taken. Nonetheless, the minimum values in column 9 show that the dogleg step does surprisingly well, even in the worst cases. The lowest fraction of the optimal value obtained for any step of any of the trust region problems encountered is .14, and the minimum is greater than .80 in 37 of the 43 test problems.

It is also interesting to observe that the indefinite dogleg step minimization algorithm performs only slightly worse than the optimal step minimization algorithm in terms of iterations and function evaluations. Further, the average number of Cholesky factorizations performed by the algorithm on all the Hessian matrices in this test set is only 1.05, and the average number of Cholesky factorizations on indefinite Hessian matrices is just 1.14.

Since we were especially interested in how the indefinite dogleg step computing function would perform in the presence of indefinite Hessians, we formed a second test set by using

several different starting points for each of the problems 2, 12, and 18. These problems were chosen because they seemed to be fairly difficult, and yielded a considerable number of indefinite Hessian matrices when used in the tests for Table 2. The starting points for the tests reported in Table 3 were chosen to be scattered around the standard starting point. Table 3 follows the same format as Table 2, except that the column for the starting point has been removed.

In Table 3 we still observe that the indefinite dogleg step usually obtains a quite high fraction of the optimal reduction. There are some low minimum reductions, but the average reductions are still very high. The overall performances of the indefinite dogleg and optimal minimization algorithms in terms of iterations and function evaluations are similar.

Our results also indicate that there is very little difference in efficiency or reliability between using the the optimal step and using the indefinite dogleg step in an unconstrained minimization algorithm.

5. Conclusions.

Our computational results indicate that even though the indefinite dogleg step computing function is not τ-optimal, it usually obtains a quite high fraction of the optimal reduction in the quadratic model in practice.

These results may not have great significance in situations in which the algebraic work of factoring $(B+\alpha I)$ or calculating the eigenvalues of B is negligible. For problems with a small number of variables, or with a very expensive objective function, the best course of action might be to simply use an optimal step computing function. Not only does an optimal step

Table 3

Performance of the Indefinite Dogleg Step Computing Function in a Trust Region Algorithm for Unconstrained Minimization on Standard Test Functions with Standard Starting Points

Fcn. Num.	Dim.	Opt. Itn.	Opt. Fcn.	Dog. Itn.	Dog. Fcn.	Ave. Redn.	Min. Redn.	Num. Chol.
2	6							
		56	82	52	70	.94	.38	58
		52	66	90	119	.91	.23	97
		33	45	38	55	.79	.17	44
		94	125	95	125	.97	.48	100
		73	102	62	86	.98	.75	64
		97	130	43	61	.78	.03	48
		151	190	145	186	.96	.15	157
		32	42	46	63	.82	.02	53
		48	69	36	50	.94	.47	38
		115	147	120	156	.92	.03	127
		28	37	41	57	.84	.15	47
12	3							
		22	26	23	27	.98	.87	23
		21	26	23	28	.95	.55	23
		19	24	26	31	.94	.42	26
18	7							
		14	18	15	19	.98	.81	18
		19	26	22	29	.92	.53	24
		20	29	20	27	.98	.87	23
		23	30	22	31	.97	.86	26

Table 4

List of Standard Test Function Numbers and Names

Test Function Number	Test Function Name
1	Helical Valley Function
2	Biggs Exp6 Function
3	Gaussian Function
4	Powell Badly Scaled Function
5	Box 3-Dimensional Function
6	Variably Dimensioned Function
7	Watson Function
8	Penalty Function I
9	Penalty Function II
10	Brown Badly Scaled Function
11	Brown and Dennis Function
12	Gulf Research and Development Function
13	Trigonometric Function
14	Extended Rosenbrock Function
15	Extended Powell Singular Function
16	Beale Function
17	Wood Function
18	Chebyquad Function

computing function provide the best possible solution to (1.1), but, given the code to compute the eigenvalue information for B, the optimal step computing function may well be easier to implement than other step computing functions.

In other situations, however, the observations in this paper are of interest and may suggest new algorithms. For problems where n is large, the two-dimensional minimization idea may be useful. Also, there appear to be various situations where a trust region approach seems attractive but the calculation of the optimal step is impractical. Examples include the

constrained optimization algorithm of Celis, Dennis, and Tapia [1985], and the tensor methods for nonlinear equations and optimization of Schnabel and Frank [1984,1986]. In these cases there is a natural two-dimensional subspace but no obvious dogleg path, so two-dimensional minimization is the apparent choice. The results of this paper seem to encourage the use of such strategies.

References.

M. R. Celis, J. E. Dennis and R. A. Tapia [1984], "A trust region strategy for nonlinear equality constrained optimization", in *Numerical Optimization 1984,* P. T. Boggs, R. H. Byrd, and R. B. Schnabel, eds., SIAM, Philadelphia, pp. 71-82.

J. E. Dennis Jr. and H. H. W. Mei [1979], "Two new unconstrained optimization algorithms which use function and gradient values", *Journal of Optimization Theory and its Applications* 28, pp. 453-482.

D. M. Gay [1981], "Computing optimal locally constrained steps", *SIAM Journal on Scientific and Statistical Computing* 2, pp. 186-197.

M. D. Hebden [1973], "An algorithm for minimization using exact second derivatives", Rept. TP55, A.E.R.E., Harwell, England.

J. J. Moré [1977], "The Levenberg-Marquardt algorithm: implementation and theory", in *Numerical Analysis, Dundee 1977, Lecture Notes in Mathematics 630,* G. A. Watson, ed., Springer-Verlag, Berlin, pp. 105-116.

J. J. Moré, B. S. Garbow, and K. E. Hillstrom [1981], "Testing unconstrained optimization software", *ACM Transactions on Mathematical Software* 7, pp. 17-41.

J. J. Moré and D. C. Sorensen [1983], "Computing a trust region step", *SIAM Journal on Scientific and Statistical Computing* 4, pp. 553-572.

B. N. Parlett [1980], *The Symmetric Eigenvalue Problem,* Prentice-Hall, Englewood Cliffs, New Jersey.

M. J. D. Powell [1970], "A new algorithm for unconstrained optimization", in *Nonlinear Programming,* J. B. Rosen, O. L Mangasarian, and K. Ritter, eds., Academic Press, New York, pp. 31-65.

R. B. Schnabel and P. Frank [1984], "Tensor methods for nonlinear equations", *SIAM Journal on Numerical Analysis* 21, pp. 815-843.

R. B. Schnabel and P. Frank [1986], "Solving systems of nonlinear equations by tensor methods", Technical Report CU-CS-334-86, Department of Computer Science, University of Colorado at Boulder.

R. B. Schnabel, B. E. Weiss, and J. E. Koontz [1985], "A modular system of algorithms for unconstrained minimization", *ACM Transactions on Mathematical Software* 11, pp. 419-440.

L. Schrage [1979], "A more portable Fortran random number generator", *ACM Transactions on Mathematical Software* 5, pp. 132-138.

G. A. Shultz, R. B. Schnabel, and R. H. Byrd [1985], "A family of trust- region-based algorithms for unconstrained minimization with strong global convergence properties", *SIAM Journal on Numerical Analysis* 22, pp. 47-67.

D. C. Sorensen [1982], "Newton's method with a model trust region modification", *SIAM Journal on Numerical Analysis* 19, pp. 409-426.